HOPNET: A Hybrid ant colony OPtimization routing algorithm for Mobile ad hoc NETwork

A thesis presented

by

Jianping Wang

to

The Department of Computer Science

in partial fulfillment of the requirements

for the degree of

Master of Science

in the subject of

Computer Science

The University of Manitoba

Winnipeg, Manitoba

December 2007

Your file Votre référence
ISBN: 978-0-494-41487-3
Our file Notre référence
ISBN: 978-0-494-41487-3

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION

HOPNET: a Hybrid ant colony OPtimization routing algorithm for mobile ad hoc NETwork

BY

Jianping Wang

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirement of the degree

MASTER OF SCIENCE

Abstract

Mobile Ad hoc network (MANET) is a group of mobile nodes which communicates with each other without any supporting infrastructure. Routing in MANET is extremely challenging because of MANETs dynamic features, its limited bandwidth and power energy. Nature-inspired algorithms (Swarm Intelligence) such as ant colony optimization (ACO) algorithms have shown to be a good technique for developing routing algorithms for MANETs. Swarm intelligence is a computational intelligence technique that involves collective behavior of autonomous agents that locally interact with each other in a distributed environment to solve a given problem in the hope of finding a global solution to the problem. In this thesis, we propose a hybrid routing algorithm for MANETs based on ACO and zone routing framework of bordercasting. The algorithm, HOPNET, based on ants hopping from one zone to the next, consists of the local proactive route discovery within a node's neighborhood and reactive communication between the neighborhoods. The algorithm has features extracted from ZRP and DSR protocols and is simulated on GlomoSim and is compared to AODV routing protocol. The algorithm is also compared to the well known hybrid routing algorithm, AntHocNet, which is not based on zone routing framework. Results indicate that HOPNET is highly scalable for large networks compared to AntHocNet. The results also indicate that the selection of the zone radius has considerable impact on the delivery packet ratio and HOPNET performs significantly better than AntHocNet for high and low mobility.

Contents

List of Figures

Acknowledgments

I would like to express my deep and sincere gratitude to my supervisor, Dr. Parimala Thulsiraman. Her direction and guidance helped to make the completion of this thesis a success.

I also wish to thank the members of my committee, Dr. Ellen Liu and Dr. Ekram Hossain.

I owe my loving thanks to my wife, Mengwei Cai and son, Yunbo Wang. Without their encouragement and understanding it would have been impossible for me to finish this work.

My special gratitude is due to my parents Shuming Wang and Shanmao Ying, my brother Jiankun Wang, my sisters Shuixian Wang and Shuiqiong Wang, and their families for their loving support.

To my wife Mengwei Cai and son Yunbo Wang, who offered me unconditional love and support throughout the course of my masters program. Also to my parents, Shuming Wang and Shanmao Ying, for giving me life.

Chapter 1

Introduction

A mobile ad hoc network (MANET)[35, 43] is a decentralized group of mobile nodes which exchange information temporarily by means of point to point wireless transmission. The network topology is unstructured and nodes may enter or leave at their will. A node can communicate to other nodes which are within its transmission range. However, to communicate to nodes out of its range, a node requires help from other nodes which play a "bridge" role to receive and forward messages. Therefore, a node in a MANET acts as both a terminal and a router. There are many applications for MANETs. For example, in a military field or operations, search and rescue activities, or any remote geographical area where there is no base station for communication. In these applications, the number of mobile nodes considered is not large, but in the future, the size of the network will increase dramatically as technology improves and therefore scalability will be an important issue for MANETs.

Routing in a MANET [38] is extremely challenging because of MANET's dynamic features, its limited bandwidth and power constraints. Due to nodes constantly moving, the network topology changes frequently: a good route will probably be unavailable after a short while. This would result in having each node along the route update their routing table frequently, causing many control packets to be sent through the network, consuming precious network resources. Therefore, to discover and maintain routes in a MANET environment is difficult.

Routing algorithms for MANET can be classified into three categories: *proactive*,

reactive and *hybrid.* In a proactive routing protocol, each node periodically broadcasts its routing table(s) to its neighbors, allowing all nodes to have a consistent network view. The advantage of this protocol is the short response time in determining a good route from source to destination due to the up to date network topology information in each node. This short response time, however, is at the expense of consuming a large portion of network bandwidth for the nonproductive control packets required to maintain a network overview at each node. Moreover, most of the established routes are never used, wasting network resources. Protocols such as DSDV[33], Fisheye[31] and WRP[26] fall into this category. In contrast, in a reactive routing protocol, a node does not need to periodically broadcast the routing table thereby decreasing use of network bandwidth. A node establishes a route to its destination, only on demand. However, a node may suffer from long waiting time before it can transmit the data packets since a node may not know which neighbor to select as the next hop to forward the packet to due to the dynamic network topology. Consequently, the node has to find a new route to the destination on the fly. Protocols such as AODV[34] , DSR[20] , ABR[42] and TORA [30] fall into this category.

A hybrid protocol, the Zone Routing Protocol (ZRP) [16, 32], combines the advantages of both proactive and reactive protocols. In ZRP, the network is divided into zones with each node belonging to one of the zones. Each node proactively maintains a routing table for nodes within its zone and reactively finds a route to its destination if the destination node lies beyond its zone. Cluster based algorithms [18] are also characterized as hybrid routing protocols. In such a network, nodes are grouped together depending on a certain clustering criterion. Each cluster has a leader and the leader represents the cluster at higher levels. The same clustering scheme is applied iteratively to the cluster leaders leading to a hierarchy. Either reactive or proactive protocols can be used within and between the clusters. Both zone routing and cluster based protocols are highly scalable. However, zone routing framework is not based on hierarchy. ZRP's main advantage is in reducing control overhead by maintaining routes to nearby nodes proactively so that local traffic can be routed immediately. This results in significant reduction in reactive control overhead avoiding global search and this is important in scenarios that exhibit traffic locality.

The idea of bordercasting in ZRP is important in reducing the Routing Request Packets (RREQ), which are not blindly sent to all the nodes but only to peripheral nodes, thus reducing the overhead. By changing one single parameter, the zone radius, the balance of proactive and reactive contributions can be adjusted.

Over the last few years, self configuring, self healing algorithms have been considered as a solution to many large scale multihop networks such as in sensor networks [48]. Recently, there is increasing interest in the use of swarm intelligence (SI) [13] or nature inspired algorithms for routing in MANETs. Swarm intelligence is a computational intelligence technique that involves the collective behavior of autonomous agents that locally interact with each other in a distributed environment to solve a given problem in the hope of finding a global solution to the problem. Ant colonies, bird flocking, animal herding and fish schooling are examples in nature that use swarm intelligence. The foraging behavior of ants [4], bees [44] and the hill building behavior of termites [37] have inspired researchers in developing routing algorithms for mobile ad hoc networks. In this thesis, we propose a hybrid routing algorithm for MANETs based on ACO [4, 9, 12]. The algorithm consists of local proactive route discovery within a nodes neighborhood and reactive communication between the neighborhoods.

ACO is based on the behavior of a group of artificial ants in search of a shortest path from the source to the destination. These artificial ants mimic real ants in nature in search of food from the nest to the food source. The ants deposit a chemical substance called a *pheromone* that other ants can sense on their journey to the destination. The ants interact with each other and the environment using the pheromone concentration. As with any perfume, if not reapplied, the scent evaporates. As the ants travel, the longer paths lose their pheromone concentration diverting all ants to choose the shortest path. There are lots of similarities between mobile ad hoc network routing and ant behavior. A MANET environment is unstructured, dynamic and distributed like the ants environment. The foraging behavior of ants and the interaction behavior of MANET to deliver packets from source to destination is similar. The goal for both systems is to find the shortest path.

ACO has been applied to many combinatorial optimization problems [3, 10, 23]

. In network optimization problems, ant based routing has been applied to static telecommunication networks [41]. Existing ant based routing protocols for MANETs [4, 5, 15, 17, 19] are very promising when compared to conventional routing algorithms. They are more efficient, are more robust and are able to discover multiple paths. However, all of these protocols have scalability and control packet overhead problems due to the fact that each node has to keep in its routing table, the pheromone amount from all its neighbors to all other nodes in the network or to desired destinations. If the number of nodes in the network is small, for example, less than 100, the table size is not of a concern. However, when the network size grows, the routing table size of each node increases dramatically, which not only consumes a large portion of the mobile nodes' memory, but also costs a lot of computation power to retrieve, modify or insert a new record in the routing table. Consequently, end to end delay becomes large.

We propose an ant based hybrid routing protocol for MANET based on the concept of zone routing framework. In our method, a node proactively maintains a routing table to all its internal nodes by periodically sending out forward ants within its zone; a node initiates a reactive route discovery procedure to send data to nodes beyond its zone. This is done again by forward ants. In our algorithm, unlike other ACO based algorithms, these forward ants will not flood the network, but will instead be directed to the destination by using the nodes' local routing table. Many routing algorithms based on ZRP have been studied in the literature as discussed in the next section. The algorithms are enhancements of ZRP and try to solve scalability and traffic overhead in large scale networks. Research in this area is still on going. However, using ACO with ZRP has not been considered. AntHocNet, though is a hybrid routing algorithm, is not scalable. This paper shows that ACO together with ZRP is highly scalable and provides good packet delivery performance for high and low mobility.

1.1 Contribution of the thesis

- Develop a scalable hybrid routing algorithm based on zone routing framework and ant colony optimization algorithm
- Simulate the algorithm to study the performance of the algorithm
- Compare HOPNET to AODV
- Compare HOPNET to AntHocNet
- Compare HOPNET to ZRP
- Use two different mobility models and study what impact these models have on performance

The rest of the thesis is organized as follows: In the next chapter, we present a review of existing ad hoc routing protocols. In Chapter 3, we discuss Ant colony Optimization and we present an overview of some ACO routing protocols. In Chapter 4, we explain the algorithm presented in this thesis followed by details of its performance evaluation of the algorithm compared to AODV, AntHocNet and ZRP routing protocols in Chapter 5. Finally, in Chapter 6 and 7 respectively, we conclude the thesis and discuss future work.

Chapter 2

Ad Hoc Routing Protocols

The IETF MANET Working Group has developed a number of protocols for mobile ad hoc networks. As Figure 2.1 shows, these protocols can be generally classified into three groups: proactive, reactive and hybrid protocols.

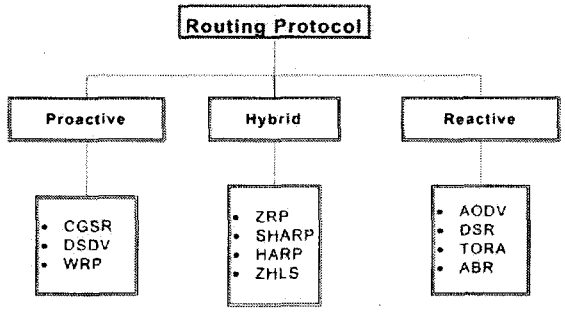


Figure 2.1: Routing Protocols

2.1 Proactive routing protocols

Proactive routing algorithms can be classified as either link-state routing or distance vector routing algorithms. In this section, we give a brief introduction to these routing mechanisms.

2.1.1 Link State Routing

In link-state routing, every node keeps a complete view of the network topology at all times. Each node periodically broadcasts its outgoing link cost to every other node in the network and constructs a routing tree used to forward packets. Most of today's wired network routing algorithms [25, 7] use some variant of the link-state method. Link-state routing algorithms have very low response time when the upper layer application needs to send out data to its peers or to the other nodes since it has all the routing information about the network. However, this advantage is at the expense of a fairly large portion of network traffic being consumed by the exchange of control packets. For an ad hoc network, whose topology changes frequently, link-state routing algorithms would spend most of the time constructing a routing tree to forward packets leading to a very inefficient algorithm. OLSR [7], TBRPF [28], FSR [31], STAR [14] are some link state routing protocols.

Fisheye State Routing (FSR)

Among these, FSR is regarded as the most efficient protocol. FSR attempts to reduce routing table update overhead while keeping a short response time. The network nodes are divided into different groups according to the number of hops needed between two nodes. The closer the groups are to a node, the more frequent are the node's corresponding link states propagated. This strategy makes the closer nodes be updated "constantly" for more accurate network topology information. For the distant nodes, the topology progressively becomes more accurate as the packet moves towards them. The advantage of FSR is that a considerable number of routing packets are eliminated, thus there is less control overhead. However, FSR is not scalable for large networks. The distant nodes have to wait a relatively long time before they get the updated link states. The nodes may therefore construct a routing tree with inaccurate information. This is disastrous for large networks. Furthermore, because each node has to store a complete network topology, a large part of the memory would be consumed as the network size grows.

2.1.2 Distance Vector Routing

In a distance vector routing algorithm, each node keeps in its routing table an entry (vector), containing destination node, next hop, hop count and other metrics, for other node. Every node periodically broadcasts its routing table to its neighbors. The neighbors then compare the received routing table entries with those of their own routing table, modify their routing tables accordingly and recursively broadcast their routing tables to their neighbors. DSDV [33] and WRP [26] are protocols based on distance vector routing.

Destination Sequenced Distance Vector(DSDV)

DSDV [33] is essentially based on distributed BellMan Ford routing method, in the effort to eliminate the routing loop problem. In DSDV, each node maintains for each desired destination node an entry in its routing table, the table also keeps for each destination node a sequence number, which is originated from the destination node. Node broadcasts its routing table in two ways: increment damp and full damp. In order to save bandwidth, only these table entries that are changed after last full damp are broadcasted to the neighbors. When the increment damp increases to some point, a full damp will be broadcasted. After a node receive a route broadcast from its neighbor, it will first compare the sequence number of the broadcast packet with that of its routing table, if the sequence number in the broadcast packet is greater than the sequence number in its routing table, the node will modify its routing table according to the broadcast. If the sequence number is equal to that in its own routing table, the node will compare the metrics, only a better metric (less hops) in the broadcast will be accepted. In any other cases, the broadcast will be discarded. In this way, DSDV can make sure the route is loop free.

Wireless Routing Protocol(WRP)

Shree Murthy and J.J. Garcia proposed a reliable, loop free proactive protocol–Wireless Routing Protocol (WRP) [26]. In WRP, each node keeps four tables: distance table, routing table, link-cost table and Message Retransmission List (MRL)

table. Whenever a node receives a update message from its neighbors or detects a link change, it first modifies its routing table and distance table for the entries whose successor is the node that sends the update. Then the node sends out a update to its neighbour, meanwhile it adds a entry to its MRL table. MRL keeps logs of acknowledge (ACK) indicating which neighbors have replied the receipt of update. For those neighbors which have not replied an ACK, the node retransmits a update to it. If the node can not receives ACK from its neighbor after a number of retransmissions, the node decides that the link to this neighbour has broken, hence, it delete the corresponding entries from its tables. If there is no update message or data packets to be sent out after a specific period, a node sends out hello message to its neighbours to let them know their connectivity with the node. Hello message is also used by a newly added node to notify other nodes in the network. Upon detecting a newcomer, a node sends to the newcomer all its routing tables. WRP avoids route cycle by adding the predecessor of each destination to the distance table. Through checking the consistency of predecessor information, the formation of temporary loops is prevented. Although WRP has some good features like reliable transmission and loop free routing, the fact that each node has to keeps a large number of tables and each table increase to very large as the network size grows makes it infeasible for routing in MANET because it consumes a large part of system resources to manage these tables. Furthermore, the periodical hello messages sent out from every node can result in the network congestion in a mobile environment.

Although proactive routing protocols have very low response time, broadcasting information to keep the nodes up to date on topology changes is a waste of bandwidth. In an ad hoc network, due to constant changes in network topology, proactive routing algorithms alone are not feasible since each node in the network has to be notified about the network topology changes by periodically broadcasting each node's routing table, even when some routes are never used.

2.2 Reactive Routing

To reduce the control packet overhead, reactive routing protocols have been proposed. With reactive routing algorithms, a node does not have to maintain routes to all other nodes constantly. Instead it finds a route to the destination on demand. An on-demand routing protocol usually consists of two phases: *route discovery* and *route maintenance*. In the *route discovery* phase, the source node sends out a route request packet (RREQ) to find a route to the destination node. When RREQ arrives at an intermediate node (not its destination), the node checks to see if it has a route to the destination and sends out a route reply packet (RREP) to the source if such a route is found. After the source receives the RREP, it begins to send out data packets using the newly created route. When a packet arrives at a node, and the link is found to be broken due to a node on the route moving away or due to node failure, *route maintenance* phase is initiated. In this phase, the node sends out an error packet to the source. Upon receiving the error packet, if the source still needs a route to the destination and does not have an alternative route to the destination, the source re-initiates a RREQ to find a new route.

AODV [34], DSR [20], TORA [30] and ABR [42] are a few reactive routing protocols.

Dynamic Source Routing(DSR)

DSR uses the source routing method. Each node keeps in its cache a nodes routing sequence from the source to the destination. Unlike AODV, DSR does not record the route in a hop by hop fashion. DSR also supports asymmetric links. This is important for an ad hoc network because nodes in an MANET may have different transmission range which results in asymmetric links. DSR supports multiple routes which can reduce the response time because a source with multiple routes does not have to wait to find a new route to the destination due to link failures.

Ad hoc On-demand Distance Vector Routing(AODV)

AODV is a well known reactive protocol. It is similar to DSR because it uses a modified route discovery mechanism. Its also maintains most recent routing information between nodes using the concept of destination sequence numbers from DSDV.

When a source node needs a route to a destination for which it has no route information, it initiates a route discovery by broadcasts a route request (RREQ) packet to all its neighbours within the network. The neighbours in turn forward the RREQ to their neighbours after updating their information for the source node. A node receiving the RREQ may send a route reply (RREP) following the reverse path of the RREQ packet. If it is either the destination or if it has a route to the destination with corresponding sequence number greater than or equal to that contained in the RREQ. If not, it rebroadcasts the RREQ. Nodes keep track of the RREQ's source IP address and broadcast ID. If they receive a RREQ which they have already processed, they discard the RREQ and do not forward it. Once the source node receives the RREP, it may begin to forward data packets to the destination. The routing information of the source is updated if it receives a RREP containing a greater sequence number or contains the same sequence number with a smaller hop count for that destination. Active Routes are maintained by every node keeping track of its neighbours with the use of periodic broadcasts (hello messages).

If a link break occurs in an active route, the upstream node of the break propagates a route error (RERR) message to the source node to inform it of the now unreachable destination(s). After receiving the RERR, if the source node still desires the route, it can reinitiate route discovery. AODV allows both unicast and multicast routing.

Temporary Ordered Routing Algorithm (TORA)

The Temporally Ordered Routing Algorithm (TORA) is a source-initiated loop-free distributed routing algorithm based on the concept of link reversal. It provides multiple routes for any desired source/destination pair. TORA uses the concept of localization of control messages by limiting them to a very small set of nodes.

The protocol performs three basic functions of route creation, route maintenance and route erasure. In the route creation phase, nodes use a "height" metric to establish a directed acyclic graph (DAG) rooted at the destination. Thereafter, links are assigned a direction (upstream or downstream) based on the relative height metric of neighbouring nodes. In times of node mobility, the DAG route is broken, thus route maintenance phase is necessary to re-establish and recreate a DAG rooted at the same destination. In the route erasure phase, a broadcast clear packet (CLR) is flooded throughout the network to erase invalid routes.

Associative-Based Routing(ABR)

ABR's focus is on finding a long-lived path. ABR is similar to DSR in the route discovery phase, and similar to AODV in the data transmission phase which is done on a hop-by-hop basis. ABR is therefore a hybrid source routing and hop-by-hop routing protocol. ABR relies on nodes sending "Hello" messages to their neighbors to maintain a stable route. For an MANET, this mechanism will consume the node's energy and cause some level of network congestion. Furthermore, ABR depends heavily on a local repair mechanism which could contribute to not only long end to end delay but also more network control overhead.

In summary, reactive routing produces less control overhead, thus increasing the packet delivery ratio and improving network efficiency. However, its response time could be very long. Furthermore, during the route discovery period, data packets may be dropped. Even worse, if nodes move frequently, the network spends most of its battery power to maintain its current routes.

2.3 Hybrid Routing

Zone Routing Protocol (ZRP)

Hass et al. [16, 32] proposed a hybrid routing protocol called Zone Routing Protocol in an effort to combine the features of proactive and reactive protocols. In ZRP, each node proactively maintains an internal routing table of the link information of

nodes that are within a variable sized routing zone of radius Υ. A reactive routing protocol is used for finding paths outside of the node's zone. In ZRP, packets are broadcast within a node's zone thereby preventing control packets from flooding the network. The response time is fast for finding paths within its zone. For finding routes outside of the node's zone, the node sends a query packet to the border nodes of the zone. Selecting the zone radius Υ plays a very important role in ZRP's performance. If Υ is set too small, a node has a very limited view about the network, which forces a node to send out route request packets frequently. Hence, the average delay time is too long making ZRP act much like a reactive protocol. On the contrary, if Υ is too large, a node has to maintain a large routing table. If the broadcast topology changes within its zone, a large portion of network traffic may be consumed by the control packets.

Dynamic routing with group construction

Chang and Hsu [6] proposed a cluster based routing approach using the characteristics of the minimum connected dominating set approach proposed by Das et al. [8]. Each mobile node communicates its connectivity information with its neighbors and determines a dominating value that is either a positive or negative value. Based on the dominating value, these nodes are grouped into positive cluster or negative cluster. The positive cluster helps in maintaining the topology information of the network. This adaptive dynamic network construction algorithm proposed in the paper reduces message complexity and provides an efficient network update. However, in order to maintain the routing group and the low level cluster, the network control overhead would be very high.

SHARP

Another hybrid routing protocol, SHARP, proposed by Ramasubramanian et al. [36] based on ZRP [16], finds a balance point between proactive and reactive routing by adjusting the degree to which information is propagated proactively versus the degree to which routing is discovered reactively. Unlike ZRP, SHARP adapts

to changing network characteristics and data traffic behavior. The protocol can be used as a purely reactive protocol in a quiescent network or be used as a purely proactive routing for hosts to which routes are in wide demand. SHARP uses efficient mechanisms to dynamically manipulate the radius of its routing zones. The proactive routing protocol in SHARP creates and maintains a directed acyclic graph rooted at the destination. The destination can vary the zone radius independently to achieve some application specific goals such as reducing packet overhead or delay jitter. AODV is used as the reactive routing protocol of SHARP. Since SHARP enables application specific adaptation strategies, different destination nodes achieve different application specific goals potentially leading to large traffic overhead.

Dynamic Hybrid Routing (DHR)

DHR [29] proposed by Park and VanVoorst is a hybrid routing algorithm that uses location information. DHR maintains the location of all nodes in existing routes. This is done by creating a proactive zone around the path or route under question. The information about the path is restricted to nodes within the zone. The zones in ZRP are static while zones in DHR are dynamic. Once a path is no longer needed the zone is destroyed. The proactive zones in DHR reuse existing portion of paths to find routes to destinations faster. In ZRP, the zones are used to route packets along border nodes. These border nodes may not necessarily find a path faster and may increase control overhead.

Independent Zone Routing Protocol (IZRP)

As an enhancement to the ZRP framework, Samar et al. [39] proposed the IZRP (Independent Zone Routing Protocol) framework. In IZRP, unlike ZRP, different nodes may have different sized routing zones to provide a balance between proactive and reactive routing. IZRP is more similar to the cluster centric approach. The results indicate that IZRP reduces the routing traffic control by 60% compared to ZRP. However, the trade off between the overhead and the benefits of the IZRP scheme is not investigated.

Two Zone Routing Protocol (TZRP)

Wang and Olariu [45] developed a two zone routing protocol based on ZRP. Each node maintains two zones, a Crisp Zone and a Fuzzy Zone. The size of these zones can be adjusted resulting in lower total running control overhead. The protocol bears some resemblance to FSLS/FSR [40] protocol. TZRP maintains a reactive component while FSLS does not. The results indicate that TZRP is more adaptive to high mobility when traffic locality holds. The paper claims that ZRP is a special case of TZRP.

Virtual Backbone Routing (VBR)

Recently, Liang and Hass [22] proposed the Virtual Backbone Routing (VBR) scheme that combines the philosophy of ZRP with hierarchical routing. VBR limits both proactive component to the local routing zones and reactive component within the virtual backbone (VB). It uses the Distributed Database Coverage Heuristic for backbone generation and maintenance and also for dynamic local zone construction and maintenance as the network topology changes. A node can join or leave the VB at any time depending on the movement of nodes and changes in the link topology. A VB node serves as a database queried by source nodes for the link-state information of nodes within the zone. When a node wants to communicate with another node outside its zone, it sends a query to its nearest database (VB node). If the route is not found in the database, the node broadcasts the route query to all other databases through the virtual links between the VB nodes. After receiving the route query, a database whose routing zone contains the destination, sends a route reply, through the reversed VB path and back to the query originating database. Each database along the way computes the best route segments that it can see within its zone and appends them to the route reply packet. Route queries are always directed to certain locations in the network, avoiding flooding in the network. This reduces the overall control traffic in the network. However, high node mobility in the network may pose a problem due to the operations that have to be performed to regenerate the VB. Furthermore, the reactive component tends to transmit a lot of control traffic.

Chapter 3

Ant Colony Optimization

Swarm Intelligence is a problem solving approach inspired by the social behaviour of insects and other animals. In particular, ants have inspired the development of a general purpose optimization techniques known as Ant Colony Optimization (ACO).

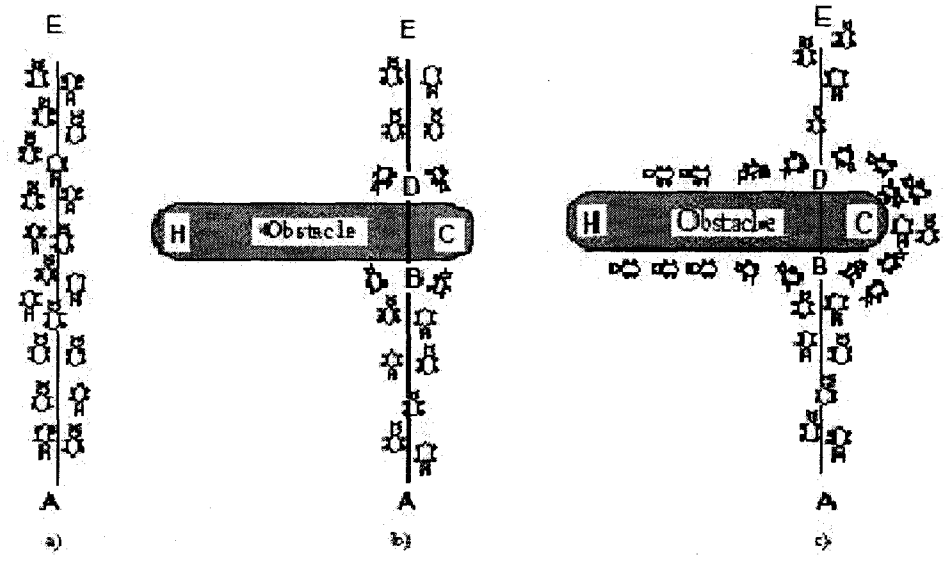
The ACO approach is based on the foraging behaviour of some ant species. Many researchers have shown particular interest in how ants are able to find shortest routes between their nest and food sources. Studies into this phenomenon has revealed that this is achieved by ants communicating indirectly by modifying their environment, this process is known as stigmergy. In the stigmergic process, the ants move to and from a food source, depositing on the ground a chemical substance known as pheromone. Other ants perceive the presence of this substance and tend to follow the path where the pheromone concentration is higher. Through this mechanism, ants are able to transport food to their nest in an effective way.

In ACO, a number of artificial ants build solutions to an optimization problem and exchange information on their quality via a communication scheme that is reminiscent of the one adopted by real ants[24].

3.1 Behaviour of real ants

Figure 3.1 is an example of an experimental setting. There is a path along which ants are walking for example from the nest A to food source E and vice versa (Figure

3.1a). Suddenly, an obstacle appears and the path is cut off. Then at position B, the ants walking from A to E, or at position D, those walking in the opposite direction have to decide whether to turn right or left (Figure 3.1b). The choice is influenced by the intensity of the pheromone trails left by preceding ants. A higher level of pheromone on the right or left path gives ants a stronger stimulus and thus a higher probability to turn right or left. The first ant reaching point B or D has the same probability to turn right or left as there was no previous pheromone on the two alternative paths. Because path BCD is shorter than BHD, the first ant following it will reach D before the first ant following BHD (Figure 3.1c).



a)Ants follow a path between points A and E.

b)An obstacle is interposed; ants can choose to go around it following one of the two different paths with equal probability.

c)On the shorter path more pheromone is laid down

Figure 3.1: ants facing an obstacle in their path [11]

The result is that an ant returning from E to D will find a stronger trail on path DCB caused by the half of all the ants that by chance decided to approach the obstacle via DCBA and by the already arrived ones coming via BCD: they will therefore prefer in probability path DCB to path DHB. As a consequence, the number of ants following path BCD per unit of time will be higher than the number of ants following BHD. This causes the quantity of pheromone on the shorter path to grow faster than on the longer one, and therefore, the probability with which any single ant chooses the path to follow is quickly biased toward the shorter one. The final result is that all ants will quickly choose the shorter path[11].

3.2 ACO Algorithms

Different ant colony optimization algorithms have been proposed. The very first published algorithm is known as the Ant System [11]. This paper introduces an optimization techniques based on ant behaviour and describes solutions for solving the Travelling Salesman Problem (TSP). In the TSP solution, an ant is placed on each city and it traverses from its current city, visiting other cities only once and returning to its origin when a tour is completed. The ants deposit pheromone on the edges connecting each city as they journey on. The pheromone concentration on the edges is constantly adjusted so that pheromone on unused edges eventually evaporate completely.

This work led to the development of a number of ACO algorithms with good results in their applications. Figure 3.2 is a non- exhaustive list of successful ant colony optimization algorithm in chronological order.

3.3 Application of ACO

There are numerous successful implementations of the ACO meta-heuristic and they have been applied to a number of different combinatorial optimization problems. Examples of applications to static combinatorial optimization problems are:

- **Traveling Salesman Problem**, where a salesman must find the shortest route by which he can visit a given number of cities, each city exactly once.
- **Quadratic Assignment Problem**, the problem of assigning n facilities to n locations so that the costs of the assignment are minimized.
- **Job-Shop Scheduling Problem**, where a given set of machines and set of job operations must be assigned to time intervals in such a way that no two jobs are processed at the same time on the same machine and the maximum time of completion of all operations is minimized.
- **Vehicle Routing Problem**, the objective is to find minimum cost vehicle routes such that:
  1. Every customer is visited exactly once by exactly one vehicle;
  2. For every vehicle the total demand does not exceed the vehicle capacity;
  3. The total tour length of each vehicle does not exceed a given limit;
  4. Every vehicle starts and ends its tour at the same position.

Algorithm	*Authors*	*Year*
Ant System (AS)	Dorigo et al.	1991
Elitist AS	Dorigo et al.	1992
Ant-Q	Gambardella & Dorigo	1995
Ant Colony System	Dorigo & Gambardella	1996
MAX-MIN AS	Stützle & Hoos	1996
Rank-based AS	Bullnheimer et al.	1997
ANTS	Maniezzo	1999
BWAS	Cordón et al.	2000
Hyper-cube AS	Blum et al.	2001

Figure 3.2: LIST OF some SUCCESSFUL ANT COLONY OPTIMIZATION ALGORITHMS [24]

- **Shortest Common Super sequence Problem**, where - given a set of strings over an alphabet - a string of minimal length that is a super sequence of each string of the given set has to be found (a super sequence S of string A can be obtained from A by inserting zero or more characters in A).
- **Sequential Ordering Problem**, which consists of finding a minimum weight Hamiltonian path 2 on a directed graph with weights on the arcs and on the nodes, subject to precedent constraints among the nodes.
- **Graph-Coloring Problem**, which is the problem of finding a coloring of a graph so that the number of colors used is minimal [1].

Another application is in communication networks, in particular on routing problems, where the problem is how to direct data traffic. There are a number of existing applications in this area as discussed in the next section.

3.4 Ant Colony Optimization Routing Protocols

This section explains some of the existing ant-based routing algorithms beginning with the very first algorithms to use Ant Colony Optimization methods to solve routing problems. The algorithms presented are for wired and wireless networks.

AntNet and ABC [4, 41] are among the earliest algorithms that use Ant Colony Optimization methods to solve routing problems for wired network. Both are proactive algorithms. AntNet uses two ants, forward and backward ants to find the shortest route from the source to the destination. The forward ant searches for the route from the source to destination, while the backward ant changes the status of the links by increasing the pheromone concentration on the links along the path. ABC, which is also similar to AntNet is considered to be more efficient since the algorithm uses less control overhead packets.

ARA [15] (Ant Colony based Routing Algorithm) proposed by Gunes et al. is one of the first ACO algorithms for finding routes in mobile ad hoc networks. It is a reactive algorithm which consists of three phases: route discovery, route maintenance

and route failure handling. Forward and backward ants are used in the route discovery phase. The route maintenance phase does not use any specialized ants, but rather uses data packets to maintain the route between the source and destination. If the source receives a route failure notification, the source re-initiates a route discovery phase. ARA paved the way for many more ant colony optimization algorithms for MANETs since it is not scalable and does not detect cycles.

ARAMA [17] is a proactive routing algorithm. The main task of the forward ant as in other ACO algorithms for MANETs is to collect path information. However, in ARAMA, the forward ant takes into account not only the hop count factor, as most protocols do, but also the links local heuristic along the route such as the node's battery power and queue delay. ARAMA defines a value called *grade*. This value is calculated by each backward ant, which is a function of the path information stored in the forward ant. At each node, the backward ant updates the pheromone amount of the node's routing table, using the *grade* value. The protocol uses the same *grade* to update pheromone value of all links. The authors claim that the route discovery and maintenance overheads are reduced by controlling the forward ant's generation rate. However, they do not clarify how to control the generation rate in a dynamic environment.

Islam et al. [19] propose an on demand routing algorithm called source update algorithm. The routing update scheme in this algorithm solves the all pair shortest path problem unlike other protocols which solves the single source shortest path problem. The algorithm uses an exploration technique instead of an exploitation technique and detects cycles. The disadvantage of this algorithm is that it does not scale well for large networks.

AntHocNet is a hybrid ant based routing protocol proposed by Di Caro[5] in the effort to combine the advantages from both AntNet and ARA. AntHocNet reactively finds a route to the destination on demand, and proactively maintains and improves the existing routes or explore better paths.

In AntHocNet, each ant maintains a list of nodes it has visited to detect cycles. The source node sends out forward ants and when it receives all the backward ants, one generation is completed. Each node i keeps the identity of the forward ants, the

path computation (number of hops) of the ant from the source to node i, and the travelling time the ant visited node i. Note that more than one ant may have reached node i and therefore the identity of the ant is important. When an ant arrives at a node, the node checks the ant's path computation and the travelling time it reached node i. If the path computation and time are within a certain limit of those produced by another ant of the same generation then the ant is forwarded. Otherwise, the ant is discarded.

In case of a link failure at a node and no alternative paths are available, the node sends a reactive forward ant to repair the route locally and to determine an alternative path. If a backward ant is received for the reactive forward ant, the data packets are sent along the newly found path and all its neighbors are notified about the change in route. Otherwise, the node sends a notification to all its neighbors of the lost destination paths which in turn initiates forward ants from the neighbors.

In the literature, AntHocNet is considered a well known hybrid algorithm. However, route maintenance is expensive with the reactive forward ants flooding the network. Each node keeps a table with all possible destination routes which leads to scalability problems.

Chapter 4

Hybrid Ant Colony Optimization Algorithm

In this chapter, we explain our new algorithm known as HOPNET (Hybrid Ant Colony Optimization Algorithm), which is a hybrid routing algorithm for MANETs based on ACO.

The HOPNET algorithm consists of local proactive route discovery within a node's neighborhood and reactive communication between the neighborhoods. The network is divided into zones which define the nodes' local neighborhood. The size of the zone is not determined locally but by the radius length measured in hops. Therefore, a routing zone consists of the specified node and all other nodes within the specified radius. A node may be within multiple overlapping zones and zones could vary in size. The nodes can be categorized as interior and boundary (or peripheral) nodes. Boundary nodes are at a radius distance from the central node. All other nodes less than the radius are interior nodes.

In Figure 4.1, if the radius of the zone is 2 then for node S, nodes A, D, F, G are boundary nodes, and nodes B,E,C are interior nodes. All other nodes are exterior nodes (outside the zone). To construct a zone, a node, and determining border nodes, a node needs to know its local neighbors. This is achieved by a detection process based on internal forward ants and internal backward ants transmitted by each node.

4.1 Routing Table

Each node has two routing tables: an *Intrazone Routing Table* (IntraRT) and an *Interzone Routing Table* (InterRT). The IntraRT is proactively maintained so that a node can obtain a path to any node within its zone quickly. This is done by periodically sending out forward ants to sample paths within its zone and determine any topology changes (such as nodes moving away, link failures, new nodes entering the zone, etc.). Once a forward ant reaches a destination, a corresponding backward ant is sent back along the path discovered. The InterRT stores the path to nodes beyond the node's zone. This source routing table is set up on demand when routes outside a zone are required. The peripheral nodes of the zone are used to find routes between zones.

4.2 Route Discovery

This section explains the route discovery process within a zone and between zones.

Route Discovery within a zone Route discovery within a zone is accomplished by using an intrazone routing table, IntraRT.

A node sends out internal forward ants periodically to all its neighbors to maintain the IntraRT. Note that we distinguish between an internal forward ant that is used within a zone and an external forward ant that is sent between zones. The size of the

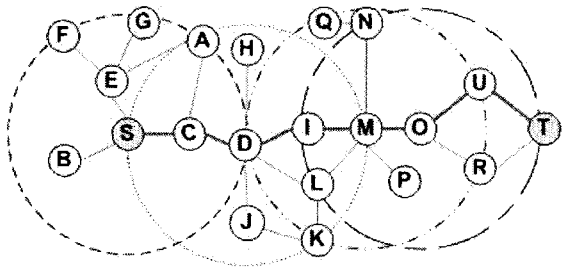


Figure 4.1: Example of Route Creation

IntraRT table is the degree of the node times all the nodes within its zone. The rows indicate the neighbors of nodes which are one hop away and the columns represent all the nodes in the zone. There are four elements in the routing table for a particular (*row, column*) pair: <*pheromone, times visited, hops, SeqNO*>. *Pheromone* is the amount of pheromone concentration on the link, *times visited* indicates the number of times the link has been visited by the ants, *hops* is used to indicate the number of hops between the node and nodes within its zone, *SeqNO* is used to prevent the same forward ant from updating the routing table repeatedly. Note that the *hops* field helps to distinguish between a peripheral node and an internal node. That is, if *hops* is equal to the zone's radius Υ, then the node in the corresponding column is an peripheral node, otherwise it is a internal node.

The data structure carried by each ant contains *Source*, *Destination*, *SequenceNO*, *Type*, *Hops*, *Path* and *Time*. Each ant is tagged with a sequence number, stored under *SeqNO* in the table, to uniquely identify the ant's generation and avoid duplication of packets from a node. The *Source* field stores the source node address; *Destination* field stores the destination address. This field is left blank for internal forward ants and stores the destination node's address for external forward ants. Source and destination nodes incrementally generate a sequence number *SequenceNO* each time forward ants or backward ants are sent out. The pair (*Source, SequenceNO*) can uniquely identify the ant's generation; There are five types of ants. This is indicated in the *Type* field. 0 for internal forward ant, 1 for external forward ant, 2 for backward ant, 3 for notification ant and 4 for error ant. The *Hops* field indicates the number of hops a forward ant can move. For internal forward ants, the zone radius is used and for external forward ant, we leave this field blank. *Path* represents the sequence of nodes between source and destination. The *Path* field has the ant travelling time since it left the source node.

When a source node S wishes to send data to a destination D, it first looks up the columns of its IntraRT to see if the destination lies in its zone. If it finds the destination in its IntraRT, then the route discovery process is finished. Note, if the destination is within a zone, there will always be a route to the destination since the IntraRT is proactively maintained. The pheromone (φ) update and node selection

procedure are given below.

When an ant selects a node v_j as its next hop node from v_i, the ant moves to node v_j and the pheromone update is as follows for entry (v_i,S) in v_j's routing table [19], where S is the source node.

$$\varphi(v_i, v_S) = \varphi(v_i, v_S) + \frac{\epsilon}{T(v_S, v_i) + w(v_i, v_j)} \tag{4.1}$$

Here ϵ is a run time parameter provided by the user, T is the total time of the path traversed by the ant and w is the time period on each edge. On all other nodes in column S, the pheromone value is decremented by

$$\varphi(v_l, v_S) = (1 - E)\varphi(v_l, v_S), \forall\ l \neq i \tag{4.2}$$

where E is the evaporation rate of the pheromone. E is a variable parameter also provided by the user. Since the source column in the routing table is updated, this is called a *source update* technique. Each ant also updates the total time of the path just traversed as $T(v_S, v_i) + w(v_i, v_j)$.

On its way back to the source, an ant again updates the pheromone concentration. However, the ant updates it for the *destination* column. It uses the *Path* to backtrack to the source. For example, an ant at node v_k traveling backwards from node v_b looks at the rows of v_b's neighboring nodes and column D. The pheromone concentration update for entry (v_b, v_D) is :

$$\varphi(v_b, v_D) = \varphi(v_b, v_D) + \frac{\epsilon}{T'} \tag{4.3}$$

where T' is $T(v_k, v_D) - T(v_k, v_b)$. This emphasizes more pheromone concentration on the path that is closer to the destination. All other neighboring node's pheromone concentration in column D are decremented as above (equation (2)).

Route Discovery between zones The interzone routing table, InterRT, is used when a node fails to find the destination within its zone (i.e. in its IntraRT table).

The InterRT table consists of four fields: *Destination*, *SeqNo*, *Path* and *Expire*. The path (stored in the *Path* field) to a destination (*Destination*) is stored in the table for a certain period of time as indicated in the *Expire* field. When a source wishes to send data packets to a destination out of its zone, it checks the InterRT to determine if a route has already been discovered by a previous ant recently. If the path has not expired, the node immediately sends out packets along the nodes indicated in the *Path* field. That is, the packet can be "self-directed" to the destination along the path. Otherwise, the node sends *external* forward ants to find a path to the destination.

The external forward ants are first sent by the node to its peripheral or border nodes. This is easy to do, since the IntraRT keeps updated path information for all the nodes within its zone which also includes the border nodes. At the boundary, the peripheral nodes check to see if the destination is within its zone by searching for the destination or path in its IntraRT table or InterRT table if the destination is outside the zone. If the destination is not within its zone or if the path has expired, the ants are forwarded to next zones via the other peripheral nodes within its zone. This process continues until the destination is found. Note that the sequence number of the ants allows duplicate ants to be discarded. At the destination, the sequence number of a forward ant just received is compared with the sequence number indicated in the InterRT table for destination. If the former is greater, then the forward ant is converted to a backward ant (type field changes from 1 to 2) in the ant's data structure. The backward ant inherits the path stored in the forward ant's data structure and traverses back to the source. On the other hand, if the sequence number of the forward ant is less or equal to the sequence number stored in the InterRT table for the destination, the ant dies since the smaller sequence number indicates that either the ant is outdated or the same sequence number indicates that another ant of the same generation has already passed through this node. When the backward ant reaches the source, the source node inserts a record in its InterRT table for the destination using the path obtained from the backward ant. The source then sends the packet along the path to the destination.

An example of route discovery between zones is given below using Figure 4.1.

Assume the source, S wants a route to the destination, T. Since T does not belong to S's zone, S will send an external forward ant to its peripheral nodes A, D, F and G using the route indicated in its IntraRT table. When the ant arrives at nodes A, G and F, these ants will die because these peripheral nodes do not have any neighbors to send the packet out. Node D, first looks through the IntraRT table to check if T is within its zone. In this example, T will not be in the table. Therefore, D will send the ant to its peripheral nodes K and M. Note that D will not send the ant to the other peripheral nodes S and A because the ant comes from C, which would discard any ants that are sent back to it. This is a routing overhead control mechanism. This mechanism will help the ant to be directed away from the source node. The mechanism prevents the ants flooding across the entire network. D appends its address to the *Path* field and sends the ant to its border nodes M (<S,D,M>) and K(<S,D,K>). Similarly, M can not find T in its zone. Node M, therefore, sends the ant to its peripheral nodes Q, U and R. Both U and R find the destination node T within their zone. U and R then send forward ants with their appended addresses to node T via the path indicated in the path field of node T in the IntraRT table.

The backward ant traverses in the reverse direction (for example, <T,U,M,D,S>) to the source S from the destination T with the type field in the data structure changed from 1 to 2.

Note that my HOPNET algorithm is able to find multiple paths from a source to a destination.

4.3 Route Maintenance

A route can be invalid due to nodes along the route moving away or a link being otherwise broken. If the damaged route is an internal one within a zone, it will recover after a short period because the IntraRT is proactively maintained. If an interzone route is invalid, the upstream node of the broken link will conduct a local repair procedure, trying to find an alternative path to the destination while buffering all the packets it receives. If the node successfully finds a new path to the destination, it will send all the buffered packets to the destination via the newly found route. Meanwhile,

a *notification* ant will be sent to the source to let the source node know about the change of route. All nodes on the path that the notification ant visits will update their interzone routing table (InterRT) to remove any invalid routes. The source will replace the related path with the *Path* value in the notification ant.

Figure 4.2 shows an example of local repair procedure. When link between O and U fails, O finds an alternative route to T via node R. So all data packets will be sent through this newly found route to T. Meanwhile, O sends a notification ant to S to announce the change of the route.

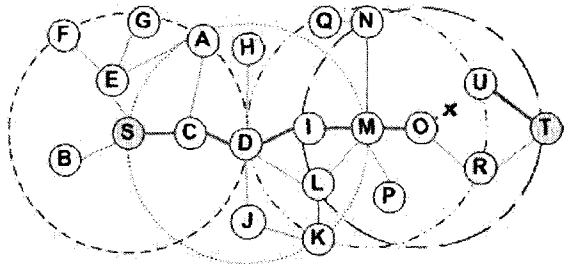


Figure 4.2: Example of Local Repair Procedure

If such an alternative path can not be found, an *error* ant will be sent to the source node. After receiving the error ant, if the source node still needs a route to the destination, it will initiate a new forward ant to attempt to find a route to the destination following the route discovery procedure described in the previous section.

Figure 4.3 explains the route rediscovery procedure. After the upstream node M detects that the link between M and O fails, M first attempts to repair the route to T by sending a forward ant to T. However, M can not receive a backward ant from T after a period of time, so it sends an error ant to S to notify the failure of the link on the route. If S still needs a route to T, it will re-initiate a route discovery process as described in the previous section.

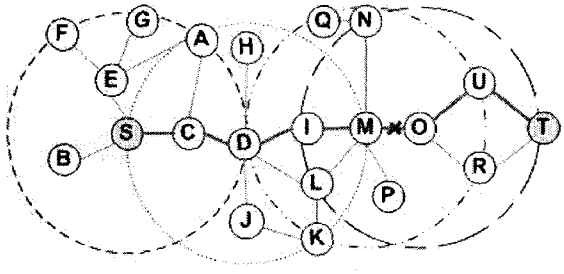


Figure 4.3: Example of Route Rediscovery

Chapter 5

Performance Evaluation

5.1 Simulation Environment

Our implementation is done using GloMoSim (Global Mobile Information System Simulator)[47, 27]. GloMoSim is a scalable simulation environment for large wireless and wired communication networks. it uses a parallel discrete-event simulation capability provided by Parsec[2]. GloMoSim is built using a layered approach, similar to the OSI seven layer network architecture. GloMoSim has several models available at each layer. Our algorithm is implemented in the network layer.

The node aggregation technique is introduced into GloMoSim to give significant benefits to the simulation performance. Initializing each node as a separate entity inherently limits the scalability because the memory requirements increase dramatically for a model with large number of nodes. With node aggregation, a single entity can simulate several network nodes in the system. Node aggregation technique implies that the number of nodes in the system can be increased while maintaining the same number of entities in the simulation. In GloMoSim, each entity represents a geographical area of the simulation. Hence the network nodes which a particular entity represents are determined by the physical position of the nodes [47].

5.2 Simulation Setup

We evaluate our algorithm in a number of simulation scenarios. The test scenarios are obtained by varying specific parameters in a base scenario. We compare the performance of our algorithm to one of the most important state-of-the-art protocols, AODV, supported in GloMoSim. In the base scenario, 50 nodes are randomly placed in a rectangular area of 3000x1000, they move according to the random way point mobility model(RWP) [21]. In this model, each node selects a random destination within the simulation area and move in a speed uniformly distributed between 0m/s and the maximum speed of 10m/s. The node travels towards the new destination and upon arrival it pauses for the specified time period and then proceeds as previously described. Simulations were run for a total of 900 seconds each time. The data traffic was generated by 20 constant bit rate (CBR) sources, with sending rates of single 64-bytes every 3 seconds. We use the 802.11 protocol at the MAC layer. The radio propagation range is set to 250 meters and the data rate is 2 Mbit/s. This base scenario is used for the experiments unless otherwise indicated.

5.3 Performance Metrics

The performance of the algorithm is measured using the following metrics;

- End to end delay - is the average time it takes for a data packet to travel from source node to destination node.
- Delivery ratio - is the total packet received by the destination node divided by the total amount of packets sent by the source node.
- Control Overhead - is the total amount of control packets divided by the data packets that were received.

5.4 Experimental Results

Figure 5.1 shows the end to end delay of HOPNET in comparison to AODV protocol. HOPNET produces better end to end delay results than AODV. This is attributed to the zone framework and the local intrazone routing table and interzone routing table. The intrazone table proactively maintains all the routes within its zone and interzone stores the the path to the destination that the ants recently visited. These tables contribute to fast end to end packet transmission since the paths are readily accessible. The evaporation rate also contributes to the paths. In HOPNET, we can adjust the evaporation rate. If the evaporation rate is too slow such as 10%, the simulation results indicate the ants travel in a cycle.

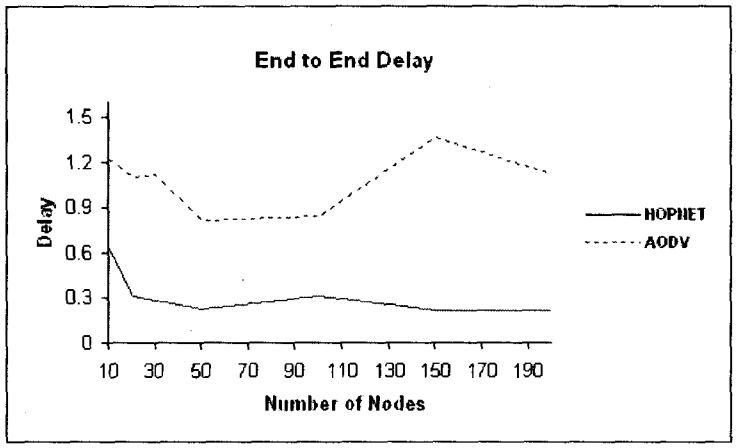


Figure 5.1: End to End Delay

We found the evaporation rate of 20% to 25% to be suitable for our algorithm. By adjusting the evaporation rate of pheromone on the links, the ants can traverse on the links or ignore the links by decrementing the pheromone concentration. The evaporation rate helps in discarding links that are broken. These reasons allow HOPNET to produce better end to end delay results.

Figure 5.2 shows the figure for delivery ratio for HOPNET and AODV. There are two reasons an intermediate node will not be able to deliver packets: the pheromone concentration along the neighboring links is zero or the node has moved away. In the

latter case, the upstream node of the broken link will conduct a local repair procedure, trying to find an alternative path to the destination while buffering all the packets it receives. If the node successfully finds a new path to the destination, it will send all the buffered packets to the destination via the newly found route, meanwhile, a *notification* ant will be sent to the source to let the source node know the change of route. In the former case, the ants cannot select any links to travel if all their links, upstream and downstream are zero, the data packet is dropped at that node, hence higher delivery ratio in AODV than HOPNET.

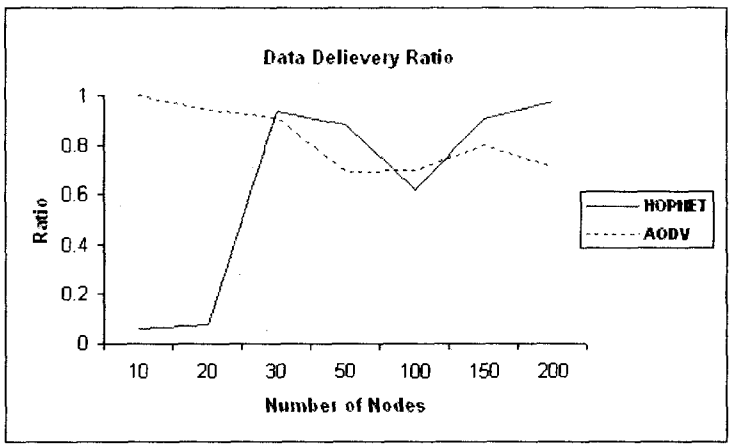


Figure 5.2: Delivery Ratio

Note that as the network size increases and with more neighbours for a node, the delivery ratio for HOPNET is better than AODV. This is because the ants can choose from multiple paths rather than a single path as in AODV.

Figure 5.3 shows the control overhead of HOPNET and AODV. AODV is a pure reactive protocol. HOPNET is proactive within a zone. The control packets are periodically sent out within a zone to maintain the routes in the zone. This is a major contributing factor to the overhead in HOPNET. As the network size increases with more neighbours for a node, the node has more choices for paths to destination and therefore, the routes between zones has multiple paths. This is seen in the figure for nodes larger than 30 and is better than AODV.

Figures 5.4, 5.5 and 5.6 show the end to end delay, delivery ratio and overhead respectively for various network sizes and zone radius. As the network size increases and radius increases, we see that the end to end delay in figure 5.4 is less, for example node 200, radius 4. This is because the algorithm starts to behave proactively accommodating a lot of nodes within one zone.

For nodes 200 and 100 after radius of 3, the delivery ratio in figure 5.5 increases. In this case, the network is dense and the ants are able to find multiple paths. On the other hand, for nodes less than 100 as the radius increases, we see from the figure that the delivery ratio decreases. This is because there is not many multiple paths for smaller networks.

As the zone radius increases for large networks, the overhead in figure 5.6 is also high. This is due to the congestion of the network with more neighbours and thereby sending more ants to keep the tables updated.

The next few experiments are comparison results between AntHocNet and HOPNET. AntHocNet [5] is a hybrid ant based routing algorithm that reactively finds a path to the destination on demand and proactively maintains and improves the existing routes. It is not based on the ZRP framework as HOPNET. In HOPNET, we form zones (or clusters) among the nodes and each node proactively maintains a

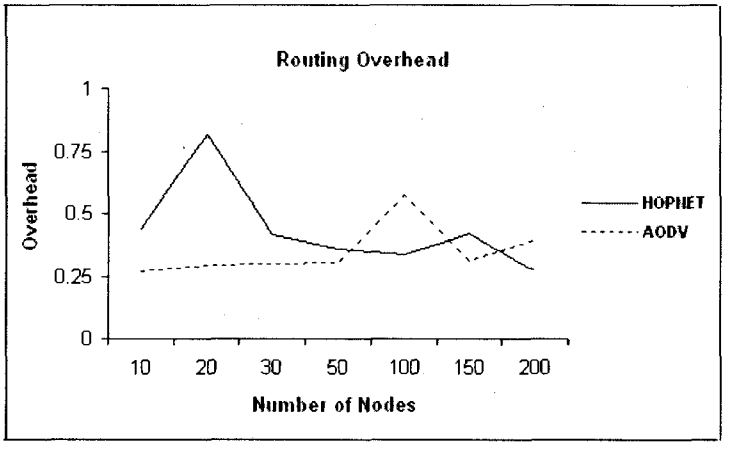


Figure 5.3: Overhead

routing table to all its internal nodes within a zone. A node initiates a reactive route discovery procedure for paths outside its zone. Though HOPNET is also a hybrid algorithm involving reactive and proactive protocols, it is based on cluster or zone formation unlike AntHocNet. This approach provides better scalability.

We compare HOPNET with AntHocNet in the next few figures. We used the same simulation set up used in AntHocNet [5]. AntHocNet also uses the random way point mobility model as HOPNET. The base scenario is as follows: the area is 1500X300 with 50 nodes randomly placed in this area with maximum speed of 20m/s and simulation time of 900 seconds and pause time of 30 seconds. Similar to the work in [5], we used different test scenarios derived from the base scenario by changing some of the parameters.

Figure 5.7 shows the delivery ratio by increasing the simulation area. The simulation area is increased from the base scenario of 1500x300 m^2 to 2500x300 m^2. As the authors [5] indicate, progressively extending the long side of the simulation area, makes the paths longer and the network sparser. We notice from the figure that HOPNET gives a slightly lower delivery ratio than AntHocNet. This we attribute to using a constant zone radius of 2. Though the network becomes sparse, combining the nodes into zones with a zone radius of two allows better route maintenance of

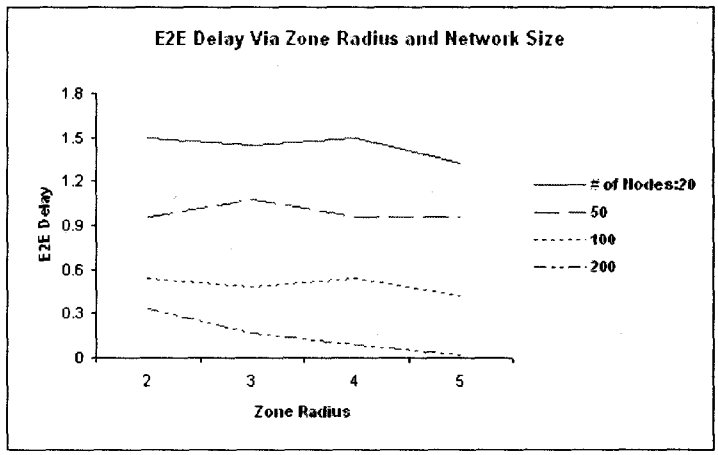


Figure 5.4: End to End Delay with Various Zone Radius

paths and route discovery. The ants hop from one zone to another zone rather than one node to another node as in AntHocNet. The information obtained about the paths within a zone is maintained within a zone in one central place and therefore, accessibility and route discovery by the ants is faster. This we think is the reason for not producing a degrading delivery ratio performance for HOPNET compared to AntHocNet.

Figure 5.8 illustrates the results due to the mobility of the nodes by varying the pause time from 0 and 900 seconds. At pause time 0 the nodes move constantly, and as we increase the pause time to 900 seconds, all nodes are static. This figure shows a similar trend as in the previous figure. For nodes with less mobility with longer pause times, the delivery ratio for HOPNET is comparable to AntHocNet. We notice that for high mobility from 0 to 10 seconds, HOPNET produces better delivery ratio than AntHocNet. However, as the mobility changes from 20 to 70 seconds, we see that the delivery ratio for HOPNET decreases then picks up after 70 seconds and is stable at higher pause times once the nodes become stable. AntHocNet on the other hand, increases the delivery ratio from 10 to 30 seconds and then decreases again until it reaches 70 seconds. The reason for the stability at higher pause times and better delivery ratios at low mobility is due to the zone radius which is set to

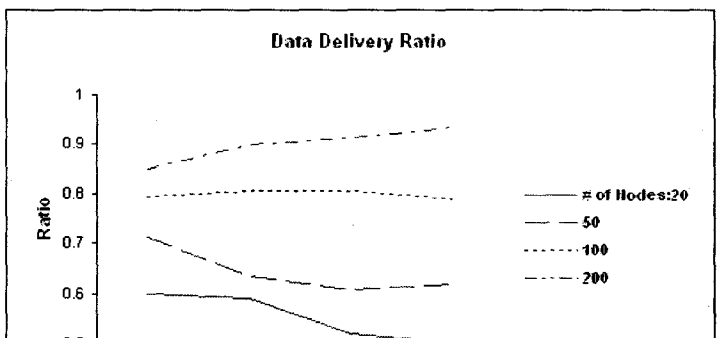


Figure 5.5: Delivery Ratio with Various Zone Radius

two. The authors [5] do not compare the overhead incurred by their algorithm in comparison to AODV. We speculate that since AntHocNet needs to sample the route constantly unlike HOPNET, which maintains within the zone, AntHocNet would produce substantial overhead.

Since HOPNET produces a comparable delivery ratio for high mobility (0 to 10 seconds) and from 70 seconds and beyond, we wanted to test our hypothesis that HOPNET may perform better if we increased the zone radius. Figure 5.9 illustrates the results for zone radius 4. Figure 5.10 shows the comparison for radius 2 and 4. The results show that the fluctuation we noticed from 20 to 70 seconds in Figure 5.8 is not seen in Figure 5.9 and is steady. Between 70 and 400 seconds, HOPNET outperforms AntHocNet. AntHocNet though does slightly better between 20 seconds and 30 seconds, the results fluctuate for high mobility and is not steady. This is because of the rapid moving nodes that cause an existing route to be invalid, and AntHocNet may happen to find an alternative route to the destination while HOPNET also finds an alternative route, and the source node has to update its internal routing table and create route for destination beyond its zone, which may lead to some packets getting dropped. However, as we can see, with the increase in pause time, HOPNET outperforms AntHocNet because most destination may fall into the source node's zone,

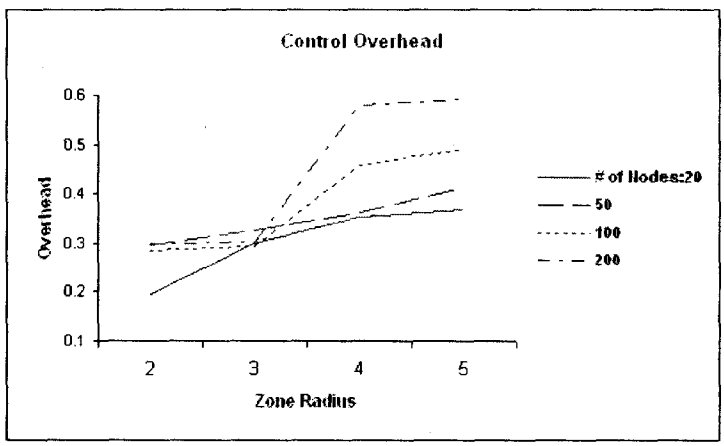


Figure 5.6: Overhead for various Zone Radius

thus data packets are sent out to the destination immediately since the algorithm has a good route maintenance policy. For those packets that are to be sent out of the zone, they are directed to the border node quickly. If the network is static (pause time = 900), HOPNET achieves 97.09% data packet delivery ratio. On the contrary, in AntHocNet, data packets may collide with its ongoing sampling control packets which get dropped due to its network congestion.

Figure 5.11 shows the scalability results of AntHocNet and HOPNET. As per the discussions in [5], starting from 50 nodes in a 1500x300 m^2 area, we multiply the terrain edges by a scaling factor and the number of nodes by the square of this factor, up to 200 nodes in a 3000x1000m^2 area. We notice that HOPNET scales extremely better than AntHocNet. The advantage of HOPNET in terms of delivery ratio increases in comparison to AntHocNet. The size of the network has very little impact on the routing algorithm. It is here that HOPNET shows its strength.

The experiments above indicate that HOPNET scales extremely well and is more advantageous in terms of packet delivery ratio in comparison to AntHocNet. The network size has no effect on the packet delivery ratio. If we increase the zone radius we noticed that the packet delivery ratio was stable for both high and low mobility.

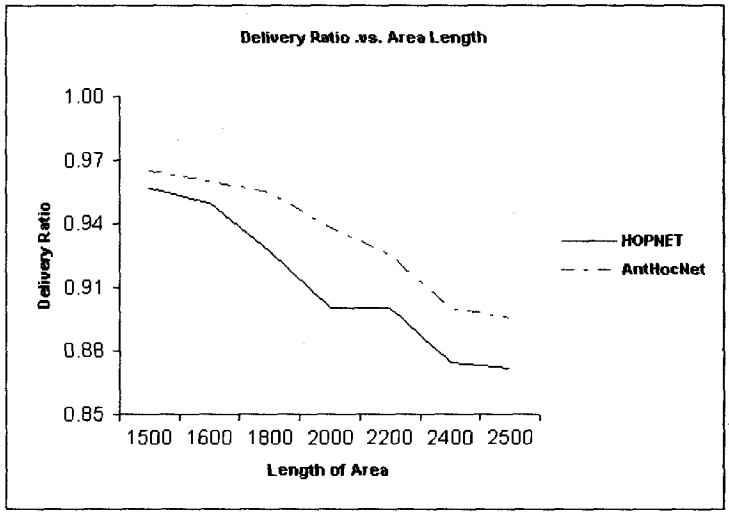


Figure 5.7: Data Delivery Ratio .vs. Length of Simulation Area

These experiments illustrate the strength of HOPNET's zone routing protocol. The zone routing framework together with an ant based routing algorithm is well suited for mobile ad hoc networks.

The next set of experiments were done using the Random Drunken model. In this model, each node moves independently with the same average speed and continuously within the region without pausing at any location. It changes direction after every unit of distance. The model can provide us with knowledge of HOPNET performance in an environment where nodes change directions very quickly but links stay relatively stable [46]. The simulation setup is the base scenario presented for the experiments using the random way point model at the beginning of the experimental section.

Figure 5.14 shows the performance results of HOPNET with AODV in terms of delivery ratio. As can be seen, the HOPNET and AODV perform the same from 10 to 50 nodes, and beyond 50 nodes, HOPNETs performance exceeds that of AODV. In the random drunken model, the nodes are continuously moving. When the number of nodes is small, some of the nodes which act as border nodes, move away. This implies that HOPNET has to perform route maintenance continuously for routes not within its zone and therefore the packet delivery ratio drops at node 50. On the other

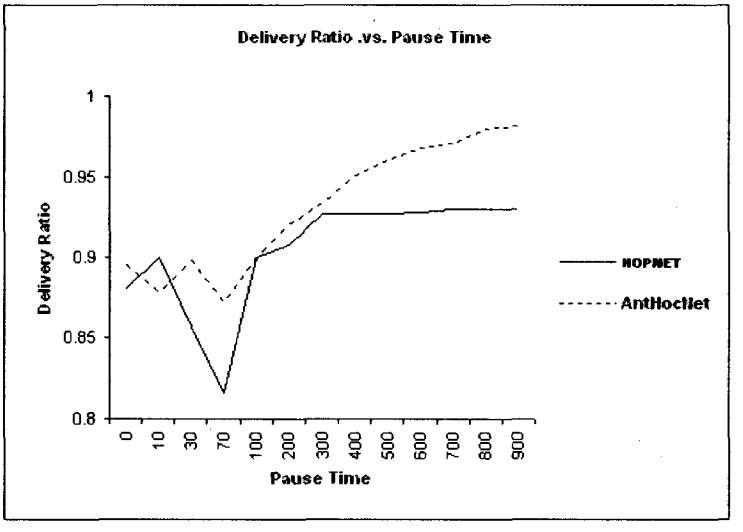


Figure 5.8: Data Delivery Ratio .vs. Pause Time

hand when the number of nodes is large, the border nodes, though they move, are still within the vicinity of their zones and the designated routes exist limiting the need for determining a new route. In AODV, when nodes move, route maintenance has to be done to find new routes by broadcasting and therefore the delivery ratio steeps low. This experiment emphasizes that bordercasting plays a significant and important role in high packet delivery ratio when nodes are on the move.

Figure 5.13 presents the overhead. When the number of nodes is small, as we explained above, when the border nodes move, new border nodes have to be determined and new routes have to be recalculated increasing the overhead in the system. As the number of nodes increase, the nodes are within its own zone and finding the new border nodes is not necessary thereby decreasing the overhead.

Figure 5.12 shows the end to end delay experienced by HOPNET and AODV. The end to end delay for HOPNET decreases as the number of node increases actually producing better packet delivery ratio and less overhead as the figures above emphasize. This is again due to bordercasting and as the nodes space gets dense the destination nodes are within their zones. The source node can route its packet to the destination without delay.

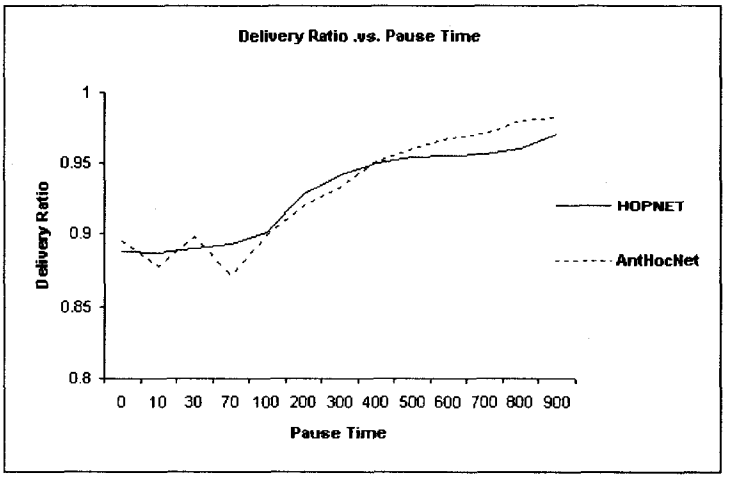


Figure 5.9: HOPNET vs. AntHocNet: Data Delivery Ratio .vs. Pause Time With Radius 4

Figure 5.15 shows the comparison between HOPNET and ZRP. The simulation setup is 5000x1250 m^2 with speed 20m/s and the number of nodes is 200. Transmission range is 250m and it uses random way point model and the nodes are uniformly distributed. This base scenario is obtained from [45]. When the zone size is small (1 or 2), the reactive and proactive algorithm work together to find a route to the destination outside the zone. We use ant colony optimization in the reactive algorithm therefore the number of control packet decreases. When the zone radius increases, the proactive part of the algorithm will have more input into the overall performance of the algorithm. ZRP uses link state routing protocol within the zones while we use ACO. Link state routing produces more overhead as can be seen in the figure. This experiment emphasizes that importance if ACO in HOPNET.

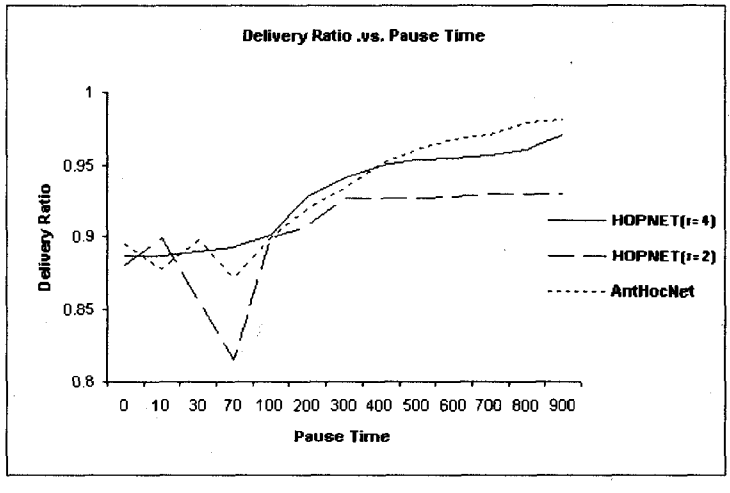


Figure 5.10: HOPNET vs. AntHocNet:Data Delivery Ratio .vs. Pause Time

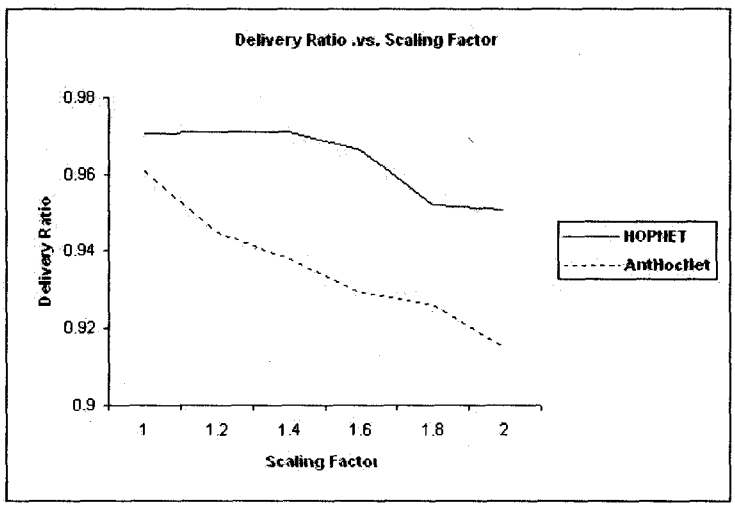


Figure 5.11: Data Delivery Ratio .vs. Scale of the Problem

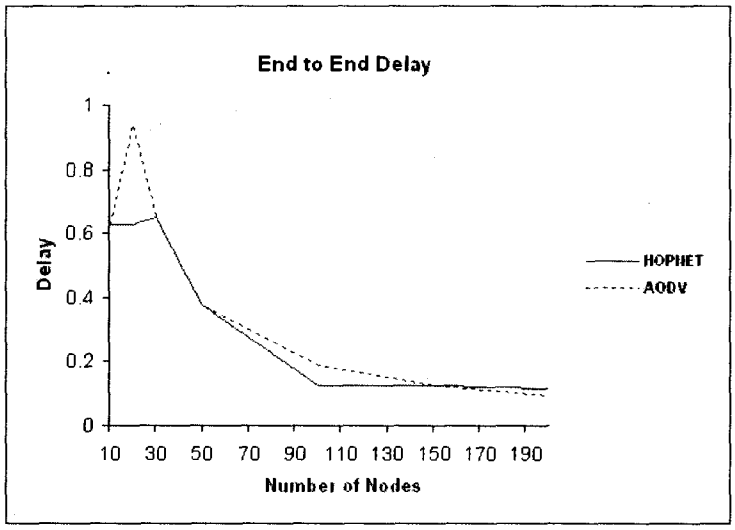


Figure 5.12: End to End Delay with Random Drunken Model

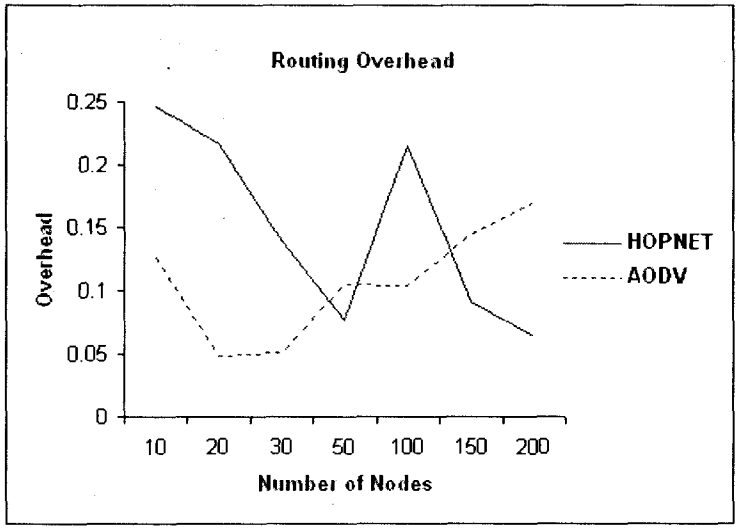


Figure 5.13: Routing Overhead with Random Drunken Model

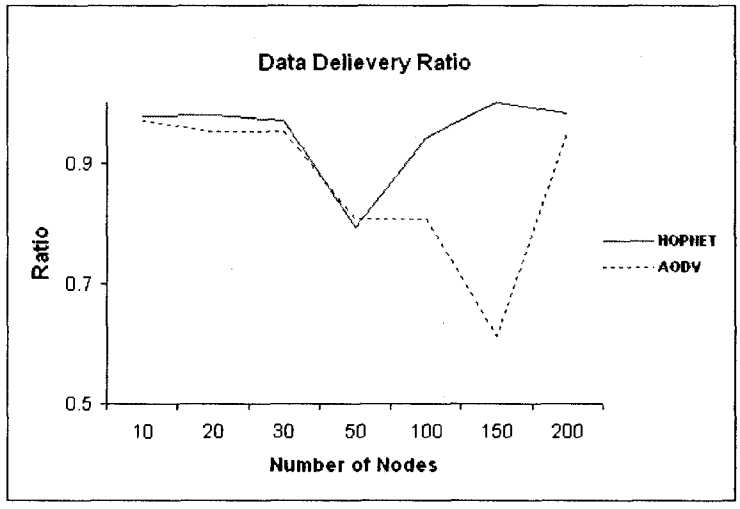


Figure 5.14: Delivery Ratio with Random Drunken Model

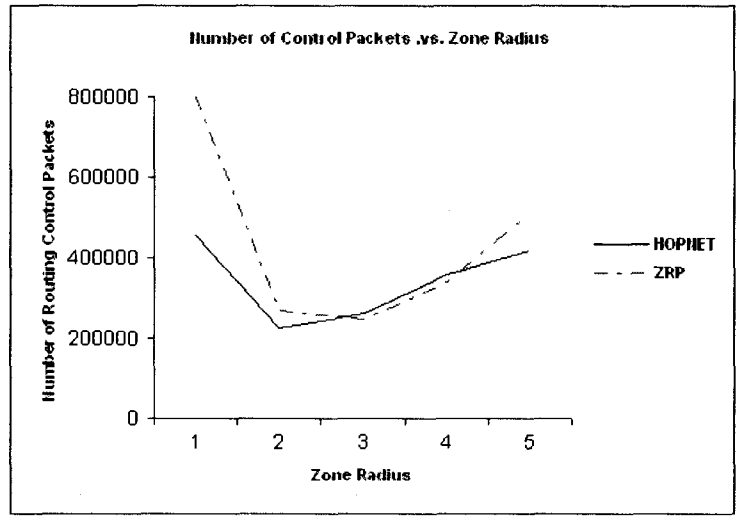


Figure 5.15: HOPNET vs. ZRP

Chapter 6

Conclusion

We proposed a novel routing algorithm for MANETs. The network is divided into zones to provide scalability. The algorithm supports proactive routing within a zone and reactive routing between zones. To our knowledge, our algorithm is the first effort to design an ant based routing protocol based on zone routing framework. The algorithm is efficient and is comparatively better than AODV for end to end delay and delivery ratio. The overhead decreases and is better than AODV as the network size increases. Experiments indicate that the zone radius has significant impact on packet delivery ratio for high and low mobility. When the zone radius is increased we notice that HOPNET outperforms AntHocNet for pause times 70 to 400 seconds. HOPNET is quite stable for high and low mobility unlike AntHocNet that fluctuates during the various pause times. We also show that the size of the network has no impact on the routing algorithm and is highly scalable compared to AntHocNet. The algorithm has been compared to random way point model and random drunken model. Experiments show that in the random drunken model, bordercasting has an impact on performance when the number of nodes is small.

Chapter 7

Future Work

Zone routing framework has been considered to provide scalable solutions to many large scale networks. Bioinspired algorithms have known to provide efficient, robust and stable solutions to self configuring, self organizing problems. In this thesis, we attempted to combine the ideas obtained from zone routing framework and ACO colony optimization algorithms for a hybrid, scalable routing algorithm for mobile ad hoc networks.

Through various experiments conducted on the algorithm, we noticed that the zone radius had a significant effect on low and high mobility in terms of packet delivery ratio. Performance greatly improves when the zone radius is increased. As a future work we plan to use the idea proposed in the paper, "Independent Zone Routing: An adaptive hybrid zone routing framework for ad hoc wireless networks", Samar et al: where the zone radius is not constant.

Our experiments indicate that a pure reactive algorithm using only ACO will encounter overhead in the overall performance of a routing algorithm. As future work, we would like to apply the idea of digital pheromones, which will share and use the experience gained by the ants. This may reduce the overhead experienced by ACO algorithm.

Bibliography

[1] T. H. Ahmed. Simulation of Mobility and Routing in Ad Hoc Networks using Ant Colony Algorithms. In *International Conference on Information Technology: Coding and Computing*, volume II, pages 698–703, Las Vegas, Nevada, USA, April 2005. IEEE Computer Society.

[2] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, B. Park, and H. Song. PARSEC: A Parallel Simulation Environment for Complex System. In *IEEE Computer Magazine*, volume 31, pages 77–85, October 1998.

[3] B. Bullnheimer, C. Strauss, and R.F. Hartl. An improved ant system algorithm for the vehicle routing problem. In *Annals of Operations Research*, volume 89, pages 319–328, 1999.

[4] G. Di Caro and M. Dorigo. AntNet: Distributed stigmergetic control for communication networks. *Journal of Artificial Intelligence Research*, 9:317–365, December 1998.

[5] G. Di Caro, F. Ducatelle, and L.M. Gambardella. AntHocNet: An adaptive nature inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications (Special Issue on Self-Organization in Mobile Networking)*, 16(2), 2005.

[6] Yu-Liang Chang and Ching-Chi Hsu. Routing in wireless/mobile ad hoc networks via dynamic group construction. *Mobile Networks and Applications*, 5:27–37, 2000.

[7] T. Clausen, P. Jacquet (editors), C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L.Viennot. Optimized link state routing protocol (OLSR). Technical report, October 2003. Network Working Group.

[8] B. Das, R. Sivakumar, and V. Bharghavan. Routing in ad hoc networks using a spine. In *IEEE International conference on computers and communication networks*, 1997.

[9] M. Dorigo, G. Di Caro, and L. Gambardella. Ant colony optimization: A new meta-heuristic. In *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1470–1477, Washington D.C, July 1999. IEEE Press.

[10] M. Dorigo and L.M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, April 1997.

[11] M. Dorigo, Maniezzo, and A. V. Colorni. The ant system–optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics*, 26:29–41, February 1996.

[12] M. Dorigo, Maniezzo, and A. V. Colorni. The ant system–optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics*, 26:29–41, February 1996.

[13] M. Dorigo E. Bonabeau and G. Theraulaz. *Swarm Intelligence: From natural to artifical systems.* Oxford University Press, New York, NY, 1999.

[14] J.J. Garcia-Luna and M. Spohn. Source-Tree Routing in Wireless Networks. In *Proceedings of the 7th International Conference on Network Protocols*, pages 273–282, Toronto, Canada, October 1999.

[15] M. Gunes, U. Sorges, and I. Bouazzi. ARA – the ant-colony based routing algorithm for MANETs. In *Proceedings of the international conference on parallel processing workshops (ICPPW'02)*, pages 79–85, Vancouver, B.C., August 2002.

[16] Z. J. Haas. A new Routing Protocol for Reconfigurable Wireless Networks. In *Proceedings of IEEE International Conference on Universal Personal Communication*, pages 1–11, Atlanta, GA, October 1997.

[17] O. Hossein and T. Saadawi. Ant routing algorithm for mobile ad hoc networks (arama). In *Proceedings of the 22nd IEEE International Performance, Computing, and Communications Conference*, pages 281–290, Phoenix, Arizona, USA, April 2003.

[18] Ting Chao Hou and Tz-Jane Tsai. An access based clusterinf protocol for multihop wireless ad hoc netowrks. *IEEE journal on selected areas in communications*, 10(7):1201–1210, 2001.

[19] M. T. Islam, P. Thulasiraman, and R. K. Thulasiram. Implementation of ant colony optimization algorithm for mobile ad hoc network applications: Openmp experiences. *Parallel and Distributed Computing Practices*, 2(5):177–191, 2004.

[20] D. B Johnson and D. A Maltz. Dynamic source routing in ad hoc wireless networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, volume 353, chapter 5, pages 153–181. Kluwer Academic Publishers, Boston, 1996.

[21] D.B Johnson and D. A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*, volume 353. Kluwer Academic Publishers, Boston, 1996.

[22] Ben Liang and Zygmunt J. Hass. Hybrid routing in ad hoc networks with a dynamic backbone. *IEEE Transactions on Wireless Communications*, 5(6):1–14, 2006.

[23] V. Maniezzo and A. Colorni. The Ant System Applied to the Quadratic Assignment Problem. *Knowledge and Data Engineering*, 11(5):769–778, 1999.

[24] Mauro Birattari Marco Dorigo and Thomas Stutzle. Ant Colony Optimization,Artificial Ants as a Computational Intelligence Technique. IRIDIA - Technical Report Series Technical Report No. TR/IRIDIA/2006-023, 2006.

[25] J. Moy. OSPF specification. Rfc 1131, Internet Engineering Task Force, October 1989.

[26] S. Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Mobile Networks and Applications*, 1(2):183–197, 1996.

[27] J. Nuevo. A Comprehensible GloMoSim Tutorial. http://externe.inrs-emt.uquebec.ca/users/nuevo/glomoman.pdf, 2004.

[28] R. Ogier, F. Templin, and M. Lewis. Topology Dissemination Based on Reverse Path Forwarding (TBRPF). Technical report, Internet Experimental RFC 3684, February 2004.

[29] Seungjin Park and Brian Van Voorst. Dynamic hybrid routing (DHR) in mobile ad hoc networks. In *Proceedings of the International Conference on Parallel Processing*, pages 1–8, Vancouver, BC, August 2002.

[30] V.D. Park and M.S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proceedings of the IEEE INFOCOM - The Conference on Computer Communications*, pages 7–11, Kobe, Japan, April 1997.

[31] G. Pei, M. Gerla, and T. Chen. Fisheye state routing in mobile ad hoc networks. In *Proceedings of the 20th IEEE international conference on distributed computing systems (ICDCS) workshop on wireless networks and mobile Computing*, pages 71–78, Taipei,Taiwan, April 2000.

[32] M. R. Peralman and Z. J. Haas. Determining the optimal configuration for the zone routing protocol. *IEEE Journal on Selected Areas in Communications, Special Issue on Wireless Ad hoc Networks*, 17(8):1395–1414, August 1999.

[33] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, volume 24, pages 234–244, New York, NY, USA, 1994.

[34] C. Perkins and E. M. Royer. Ad–hoc on-demand distance vector routing. In *Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, New Orleans, LA, February 1999.

[35] Charles E. Perkins. *Ad Hoc Networking.* Addison-Wesley, 2001.

[36] Venugopalan Ramasubramanian, Zygmunt J. Hass, and Emin Gun Sirer. SHARP: A hybrid adaptive routing protocol for mobile ad hoc networks. In *MobiHoc*, pages 303–314, Annapolis, Maryland, June 2003.

[37] M. Roth and S. Wicker. Termite:Ad-Hoc Networking with Stigmergy. In *Proceedings of IEEE Global Telecommunications Conference*, volume 5, pages 2937–2941, San Francisco, USA, December 2003.

[38] E. Royer and C. Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *IEEE Personal Communications Magazine*, pages 46–55, April 1999.

[39] Prince Samar, Mark R. Perlman, and Zygmunt J. Hass. Independent zone routing: an adaprive hybrid routing framework for ad hoc wireless networks. *IEEE/ACM transactions on networking*, 12(4), August 2004.

[40] C. Santivanez and R. Ramanathan. Making link state routing scale for ad hoc networks. In *Proceedings of MOBIHOC conference*, 2001.

[41] R. Schoonderwoerd, O. E. Holland, J. L. Bruten, and L. J. M. Rothkrantz. Ant-Based Load Balancing in Telecommunications Networks. *Journal of Adaptive Behavior*, 5(2):169–207, 1996.

[42] C.-K. Toh. Associativity-based routing for ad-hoc mobile networks. *Wireless Personal Communications*, 4(2):103–139, March 1997.

[43] C. K. Toh. *Ad Hoc Mobile Wireless Networks: Protocols and Systems.* Prentice Hall, December 2001.

[44] Horst F. Wedde, Muddassar Farooq, Thorsten Pannenbaecker, Bjoern Vogel, Christian Mueller, Johannes Meth, and Rene Jeruschkat. BeeAdHOC: An Energy Efficient Routing Algorithm for Mobile Ad Hoc Networks Inspired by Bee Behavior. In *Proceedings of Genetic and Evolutionary Computation Conference*, pages 153–160, Washington, DC, June 2005.

[45] Lan Wnag and Stephan Olariu. A two zone hybrid routing protocol for mobile ad hoc networks. *IEEE transactions on parallel and distributed systems*, 15(12), Decemeber 2004.

[46] Kui Wu and Janelle Harms. Performance study of proactive flow handoff for mobile ad hoc networks. *ACM/Kluwer Wireless Networks Journal*, 12(1):119–135, February 2006.

[47] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks. In *12th Workshop on Parallel and Distributed Simulation (PADS'98)*, pages 154–161, Banff, Alberta, CA, May 1998.

[48] Hongwei Zhang and Anish Arora. Gs^3: scalable, self configuration and self healing in wireless sensor networks. *Computer Networks*, 43:459–480, 2003.