
Student Work

7-1-2003

An evolutionary approach to routing in mobile AD HOC networks using dominating sets.

Hiranmayi Sreenivas

Follow this and additional works at: <https://digitalcommons.unomaha.edu/studentwork>

Recommended Citation

Sreenivas, Hiranmayi, "An evolutionary approach to routing in mobile AD HOC networks using dominating sets." (2003). *Student Work*. 3582.

<https://digitalcommons.unomaha.edu/studentwork/3582>

This Thesis is brought to you for free and open access by DigitalCommons@UNO. It has been accepted for inclusion in Student Work by an authorized administrator of DigitalCommons@UNO. For more information, please contact unodigitalcommons@unomaha.edu.



**AN EVOLUTIONARY APPROACH TO ROUTING IN MOBILE AD HOC
NETWORKS USING DOMINATING SETS**

A Thesis

Presented to the

Department of Computer Science

and the

Faculty of the Graduate College

University of Nebraska

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

University of Nebraska at Omaha

by

Hiranmayi Sreenivas

July 2003

UMI Number: EP74780

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

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



UMI EP74780

Published by ProQuest LLC (2015). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code

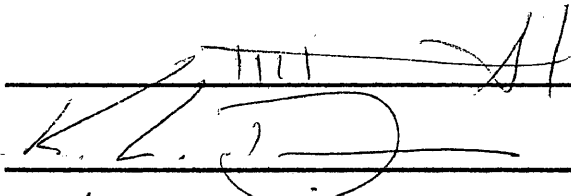


ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

THESIS ACCEPTANCE

Acceptance for the faculty of the Graduate College,
University of Nebraska, in partial fulfillment of the
requirements for the degree Master of Science in Computer Science,
University of Nebraska at Omaha.

Committee


Youn Jonghoon
Hamid Sharif (CHA)

Chairperson Hesham H. Ali

Date 7/28/03

Acknowledgments

I am indebted to my advisor, Dr. Hesham Ali, for his unending guidance, patience, encouragement and support during the course of my research.

My sincere thanks to the other members of my committee: Dr. Ken Dick, Dr. Hamid Sharif and Dr. Jong-hoon Youn for sharing their knowledge and for their helpful comments and suggestions.

A special thanks to Dr. Deepak Khazanchi for providing me with valuable advice during the course of my graduate studies.

I would also like to thank my colleagues, friends, faculty and staff at the Peter Kiewit Institute for their technical and non-technical support during my stay at the University.

Finally, I would like to thank my parents, sister, brother-in-law and young nephew for their endless love, support and patience without which this work would not have been possible.

AN EVOLUTIONARY APPROACH TO ROUTING IN MOBILE AD HOC NETWORKS USING DOMINATING SETS

Hiranmayi Sreenivas, MS

University of Nebraska, 2003

Advisor: Dr. Hesham Ali

This thesis presents a new approach to routing in ad-hoc wireless networks using virtual backbones that may be approximated by the graph theoretic concept of dominating sets. Ad hoc wireless networks provide a flexible and quick means of establishing wireless peer-to-peer communications. Routing remains the main challenging problem in an ad hoc network due to its multihop nature and dynamic network topology. Several protocols based on virtual backbones in ad hoc wireless networks have been proposed that may be used to simplify the routing process. However, little is known about the network routing performance of these protocols and no attempt has previously been made to directly compare them.

This thesis is the first research effort to implement, analyze and compare the routing performance of dominating-set-based routing protocols. In this study, we examine four existing routing approaches using a virtual backbone, or “spine”, imposed on the ad-hoc network. We then propose an evolutionary approach to constructing a stable minimum connected dominating set in an ad hoc wireless network: this employs the use of a genetic algorithm. Since the mobile nodes that constitute an ad hoc wireless network are constantly in motion, the network configuration is subject to constant change in a

manner that resembles the biological process of mutation. This evolution of networks over time lends itself naturally to a model based on genetic algorithms.

As part of an in-depth study of the application of genetic algorithms in the field of wireless networks, a scatternet formation protocol for Bluetooth networks was designed, developed and evaluated. This helped to build the knowledge base required to implement new routing protocols using the network simulator ns-2. Simulation studies were then conducted using ns-2 to compare the performance of previously proposed dominating-set-based routing approaches.

In this thesis, we analyze the performance of our evolutionary routing approach and compare it with the previous approaches. We present our simulation results and show that our evolutionary routing approach outperforms the other routing algorithms with respect to end-to-end packet delay, throughput, packet delivery ratio and routing overhead across several different scenarios. Thus, we demonstrate the advantages of utilizing a genetic algorithm to construct a backbone that is used to effectively route packets in an ad-hoc wireless network.

Table of Contents

Chapter 1	Introduction.....	1
1.1	Problem Introduction.....	1
1.2	Problem Motivation.....	6
1.3	Summary of Existing Methods.....	9
1.4	Approach.....	11
Chapter 2	Problem Definition.....	13
2.1	Problem Domain: Routing in Ad Hoc Wireless Networks.....	14
2.2	Problem Definition: Routing in Ad Hoc Wireless Networks using a Virtual Backbone.....	18
2.3	Problem Domain: Bluetooth Networks.....	21
2.4	Problem Definition: Scatternet Formation in Bluetooth Ad Hoc Networks....	22
Chapter 3	Background Information.....	25
3.1	Das et al's Algorithm.....	26
3.2	Wu and Li's Algorithm.....	29
3.3	Alzoubi et al's Algorithm.....	32
3.4	Cheng et al's Algorithm.....	35
3.5	Introduction to Genetic Algorithms.....	42
Chapter 4	An Evolutionary Bluetooth Scatternet Formation Protocol.....	45
4.1	Related Work.....	46
4.2	Genetic Algorithm for Scatternet formation in Bluetooth networks.....	50
4.3	Performance Evaluation.....	52
4.3.1	Assumptions and Performance Measures.....	52
4.3.2	Performance.....	54
Chapter 5	An Evolutionary Dominating-set-based Routing Approach in Mobile Ad Hoc Networks.....	60
5.1	Genetic Algorithm for Dominating-set-based routing in Mobile Ad Hoc Networks.....	61
Chapter 6	Performance Evaluation.....	70
6.1	End-to-end packet delay.....	72
6.2	Throughput.....	76
6.3	Packet delivery ratio.....	79
6.4	Routing Overhead.....	83
6.5	Varying Packet Size.....	87
6.6	Experiments related to the Genetic Dominating-set-based Routing Protocol.....	92
6.6.1	Size of Dominating Set or Virtual Backbone.....	93

6.6.2	Network Diameter.....	94
6.6.3	High-Energy Backbone Nodes.....	95
Chapter 7 An Evolutionary Dominating-set-based Routing Approach using Clusters of Nodes.....		108
7.1	End-to-end packet delay.....	109
7.2	Throughput.....	112
7.3	Packet delivery ratio.....	115
7.4	Routing overhead.....	118
Chapter 8 Summary and Conclusions.....		121
8.1	Background.....	121
8.2	Objectives Achieved.....	123
8.3	Performance Evaluation.....	124
8.4	Future Work.....	127
References.....		128

List of Figures

Figure 1	Ad hoc network with three participating mobile nodes.....	15
Figure 2	Classification of routing protocols for Mobile Ad Hoc Networks (MANETs).....	17
Figure 3	Dominating Set.....	20
Figure 4	Piconets and Scatternet.....	23
Figure 5	Basic MCDS Routing.....	29
Figure 6	Wu and Li's CDS Routing.....	32
Figure 7	Alzoubi et al's CDS Construction.....	35
Figure 8	Cheng et al's CDS Construction – Algorithm I.....	38
Figure 9	Cheng et al's CDS Construction – Algorithm II.....	41
Figure 10	Number of piconets <i>or</i> Number of masters.....	56
Figure 11	Number of rounds in Scatternet formation.....	57
Figure 12	Execution Time of the Genetic Algorithm.....	57
Figure 13	Number of bridges in a Scatternet.....	58
Figure 14	Dominating-set-based routing using the proposed genetic algorithm.....	65
Figure 15	Comparison of average end-to-end packet delay with 10 traffic sources and pause time = 100 seconds.....	74
Figure 16	Comparison of average end-to-end packet delay with 10 traffic sources and pause time = 300 seconds.....	74

Figure 17	Comparison of average end-to-end packet delay with 20 traffic sources and pause time = 100 seconds.....	75
Figure 18	Comparison of average end-to-end packet delay with 20 traffic sources and pause time = 300 seconds.....	75
Figure 19	Comparison of average throughput with 10 traffic sources and pause time = 100 seconds.....	77
Figure 20	Comparison of average throughput with 10 traffic sources and pause time = 300 seconds.....	78
Figure 21	Comparison of average throughput with 20 traffic sources and pause time = 100 seconds.....	78
Figure 22	Comparison of average throughput with 20 traffic sources and pause time = 300 seconds.....	79
Figure 23	Comparison of average packet delivery ratios with 10 traffic sources and pause time = 100 seconds.....	81
Figure 24	Comparison of average packet delivery ratios with 10 traffic sources and pause time = 300 seconds.....	82
Figure 25	Comparison of average packet delivery ratios with 20 traffic sources and pause time = 100 seconds.....	82
Figure 26	Comparison of average packet delivery ratios with 20 traffic sources and pause time = 300 seconds.....	83
Figure 27	Comparison of average routing overhead with 10 traffic sources and pause time = 100 seconds.....	85

Figure 28	Comparison of average routing overhead with 10 traffic sources and pause time = 300 seconds.....	86
Figure 29	Comparison of average routing overhead with 20 traffic sources and pause time = 100 seconds.....	86
Figure 30	Comparison of average routing overhead with 20 traffic sources and pause time = 300 seconds.....	87
Figure 31	Comparison of average end-to-end packet delay for 64 Byte packets with 10 traffic sources and pause time = 100 seconds.....	89
Figure 32	Comparison of average throughput for 64 Byte packets with 10 traffic sources and pause time = 100 seconds.....	90
Figure 33	Comparison of average packet delivery ratio for 64 Byte packets with 10 traffic sources and pause time = 100 seconds.....	91
Figure 34	Comparison of average routing overhead for 64 Byte packets with 10 traffic sources and pause time = 100 seconds.....	92
Figure 35	Average size of Dominating Set generated by the genetic protocol.....	94
Figure 36	Average network diameter generated by the genetic protocol.....	95
Figure 37	Comparison of average end-to-end packet delay for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds.....	97
Figure 38	Comparison of average end-to-end packet delay for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds.....	97

Figure 39	Comparison of average end-to-end packet delay for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds.....	98
Figure 40	Comparison of average end-to-end packet delay for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds.....	98
Figure 41	Comparison of average throughput for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds.....	100
Figure 42	Comparison of average throughput for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds.....	100
Figure 43	Comparison of average throughput for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds.....	101
Figure 44	Comparison of average throughput for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds.....	101
Figure 45	Comparison of average packet delivery ratio for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds.....	102
Figure 46	Comparison of average packet delivery ratio for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds.....	103
Figure 47	Comparison of average packet delivery ratio for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds.....	103
Figure 48	Comparison of average packet delivery ratio for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds.....	104

Figure 49	Comparison of average routing overhead for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds.....	105
Figure 50	Comparison of average routing overhead for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds.....	105
Figure 51	Comparison of average routing overhead for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds.....	106
Figure 52	Comparison of average routing overhead for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds.....	106
Figure 53	Comparison of average end-to-end packet delay for clustering with 10 traffic sources and pause time = 100 seconds.....	110
Figure 54	Comparison of average end-to-end packet delay for clustering with 10 traffic sources and pause time = 300 seconds.....	111
Figure 55	Comparison of average end-to-end packet delay for clustering with 20 traffic sources and pause time = 100 seconds.....	111
Figure 56	Comparison of average end-to-end packet delay for clustering with 20 traffic sources and pause time = 300 seconds.....	112
Figure 57	Comparison of average throughput for clustering with 10 traffic sources and pause time = 100 seconds.....	113
Figure 58	Comparison of average throughput for clustering with 10 traffic sources and pause time = 300 seconds.....	113
Figure 59	Comparison of average throughput for clustering with 20 traffic sources and pause time = 100 seconds.....	114

Figure 60	Comparison of average throughput for clustering with 20 traffic sources and pause time = 300 seconds.....	114
Figure 61	Comparison of average packet delivery ratio for clustering with 10 traffic sources and pause time = 100 seconds.....	116
Figure 62	Comparison of average packet delivery ratio for clustering with 10 traffic sources and pause time = 300 seconds.....	116
Figure 63	Comparison of average packet delivery ratio for clustering with 20 traffic sources and pause time = 100 seconds.....	117
Figure 64	Comparison of average packet delivery ratio for clustering with 20 traffic sources and pause time = 300 seconds.....	117
Figure 65	Comparison of average routing overhead for clustering with 10 traffic sources and pause time = 100 seconds.....	118
Figure 66	Comparison of average routing overhead for clustering with 10 traffic sources and pause time = 300 seconds.....	119
Figure 67	Comparison of average routing overhead for clustering with 20 traffic sources and pause time = 100 seconds.....	119
Figure 68	Comparison of average routing overhead for clustering with 20 traffic sources and pause time = 300 seconds.....	120

Chapter 1

Introduction

This thesis focuses on the study, implementation and performance evaluation of various approaches to routing in ad hoc wireless networks using graph theoretic concepts. This thesis also proposes an evolutionary routing approach based on genetic algorithms and compares the performance of this approach with previous approaches through extensive simulations. This chapter introduces the problem and the motivation for our work and also provides a high-level overview of our approach.

1.1 Problem Introduction

The origin of coded transmission using air as a medium began with radio telegraphy in the 19th century. With the advent of World War I, and subsequently World War II, there were rapid advances in wireless radio communications to support military tactical communications. Cellular telephony emerged in the 1970s and formed the basis for present day wireless networks. Today, wireless computing has taken the world by storm. With the falling cost of devices such as laptops, hand-held computers etc, wireless or mobile computing has become affordable to both business users and private consumers. People on the move use their mobile devices to read email, buy/sell stocks etc. On the other hand, disaster recovery, search-and-rescue

operations, and military operations in the battlefield use wireless networks to communicate important and sensitive information to each other.

A distributed system with mobile units is usually an *infrastructure* network composed of a reliable static network of fixed hosts and a number of mobile units connected by a slow and usually unreliable wireless link. This is the typical single hop cellular network model. Some of the fixed hosts, called Mobile Support Stations or Base Stations, also use a wireless interface and provide a gateway for communication between the wireless network and the static network. A mobile unit can communicate with a Base Station only within a limited geographical region; this region is known as the Base Station's "cell". A "handoff" occurs when mobile nodes move from the range of one base station into the range of another.

This wired backbone infrastructure comprising of base stations may be unavailable for use by mobile nodes, due to several reasons that include unexpected natural disasters or radio shadows. Also, it may be infeasible to construct this wired backbone in certain areas e.g. in wilderness areas, festival grounds, and battlefields. In emergency search-and-rescue operations or military maneuvers, it may be necessary to deploy a temporary communication network immediately.

In the above situations, a *dynamic* or *infrastructure-less* network would be the appropriate choice; mobile hosts in such a network communicate with each other over

wireless links, without any interaction with fixed hosts. This communication may be direct or indirect i.e. routed via another mobile host. The mobile hosts act as routers; they have the capability to relay packets. Such networks are known as “ad hoc” networks.

The precursor of ad hoc networking technology was the packet radio network. Packet radio applies packet communications to a radio channel rather than to a wire-based medium. This technology can be used to create Local Area Networks (LANs) that link devices, as well as to provide gateways to other network systems and databases [39].

An *ad hoc wireless network*, also known as Mobile Ad Hoc Network or MANET [6], is a multihop network in which mobile hosts/computers communicate over a shared wireless medium. An ad hoc network can be deployed quickly in situations that are characterized by lack of established infrastructure and that demand flexibility and ease of network setup: distributed computing, battlefield operations, disaster relief situations and search-and-rescue operations. The features of such networks include: dynamic topology, multihop communication, limited resources (e.g. bandwidth), security vulnerabilities and the absence of centralized infrastructure.

Since the mobile nodes that comprise an ad hoc network are constantly moving in and out of each other’s transmission ranges, the network topology is dynamic and poses a

challenge in the design of routing algorithms. The communication is multihop because other mobile nodes forward packets sent from a source mobile node to a destination mobile node. Since wireless links have much less available bandwidth than wired links, bandwidth is limited in such networks. Also, mobile nodes are battery-powered. Therefore, there is considerable resource constraint in ad hoc wireless networks. It is far easier to “sniff” a wireless communication on a particular radio frequency than it is to hack into a physical wire. This introduces inherent security vulnerabilities in wireless networks.

A well designed architecture for mobile ad hoc networks involves all networking layers, ranging from the physical layer to the application layer. Extensive research has been devoted to mobile ad hoc networks at every layer, such as medium access control, broadcast, routing, distributed algorithms, and Quality of Service (QoS) issues for user applications. The routing problem is one of the most important issues in ad hoc networks; the Internet Engineering Task Force (IETF) Working Group on Mobile Ad Hoc Networks (MANET) is standardizing routing in ad hoc networks with the goal of supporting networks scaling up to hundreds of routers.

This thesis focuses on the routing problem in mobile ad-hoc networks. Routing in ad-hoc networks involves the design of an efficient, robust routing protocol that differs from traditional routing approaches for wired networks because it must consider the following characteristics of ad-hoc networks [39]:

- The topology is highly dynamic and changes in this topology are difficult to predict.
- Mobile ad hoc networks are based on wireless links that have different characteristics than their wired counterparts, in particular: lower capacity, limited physical security, and greater susceptibility to loss, delay or jitter.
- Mobile nodes rely on batteries or other exhaustible power supplies for energy. Thus, energy savings is an important criterion in routing protocol design.

Desirable qualitative properties of routing protocols for ad hoc networks include

[17]:

- Compatibility with the underlying data link and physical layers
- Optimizes resource usage
- Distributed operation
- Loop-freedom
- Demand-based or proactive operation
- Security
- "Sleep" period operation
- Unidirectional link support

1.2 Problem Motivation

Traditional routing protocols for MANETs can be classified as: *proactive* (table-driven) and *reactive* (on-demand), depending on how they react to network topology changes [25], [35]. *Proactive*, or *table-driven*, protocols attempt to maintain routes continuously, such that the route is already available when it is needed for forwarding a packet. In such protocols, routing tables are exchanged among neighboring nodes every time a change occurs in the network topology. Examples of proactive protocols include Wireless Routing Protocol (WRP) [30] and Destination Sequenced Distance Vector (DSDV) [33]. In contrast, *reactive*, or *on-demand*, or *source-initiated*, protocols send control messages to discover a route between a given source-destination pair only when necessary. Examples of reactive protocols include Dynamic Source Routing (DSR) [13], Signal Stability-Based Adaptive Routing (SSA) [22], Ad Hoc On-Demand Distance Vector Routing (AODV) [34] and Temporally Ordered Routing Algorithm (TORA) [32].

The proactive approach is analogous to the connectionless approach of traditional datagram networks. This is based on the constant update of routing information [35]. Maintaining consistent and updated routes pertaining to each source-destination pair requires the propagation of a large amount of routing information. As a result, a route between any source-destination pair is always available but such protocols cannot perform properly when node mobility is high or when there are a large number of mobile nodes in the network. The control overhead in terms of network traffic and

power consumption proves to be a serious limitation in MANETs, where bandwidth and power are scarce resources.

The reactive approach is analogous to connection-oriented communications and is based on the observation that (i) routes in a dynamic topology expire frequently, and (ii) not all the routes are used at the same time. A reactive protocol creates and maintains routes between source-destination pairs only when necessary, i.e. when requested by the source. Therefore, in this approach, the control overhead is drastically reduced. However, a route is not initially available and the route discovery process generates a latency period.

An alternative hybrid approach is to combine both proactive and reactive characteristics in order to benefit from the short response time provided by proactive routing protocols and to limit control overhead as in reactive protocols. This can be done by proactively handling all routes that are frequently used and create all the other routes on demand [25].

As stated previously, to utilize the limited available bandwidth effectively, communication overhead must be minimized. We have seen that existing proactive as well as reactive routing protocols for ad hoc networks employ “flooding”, a broadcast mechanism, to disseminate control packets for topology updates and route discovery. Flooding causes redundancy, contention and collision. In addition, broadcast

mechanisms are inherently unreliable in conveying information to all nodes in a wireless network; not all hosts receive broadcast messages even in a collision-free environment.

The problem is to design a routing protocol for mobile ad hoc networks that minimizes the flooding problem, thereby minimizing the communication overhead and optimizing resource usage. This can be achieved by restricting broadcast to a subset of the mobile nodes that constitute the ad hoc network. At the same time, the routing protocol must ensure the availability of consistent and up-to-date routes.

Imposing a *virtual backbone structure* on an ad hoc network by approximating a *dominating set* uses unicast to replace flooding. This dominating set could be connected as well as minimum in size. This virtual backbone/spine is dynamic and must be recomputed periodically; its main function is to *compute* and *update* routes. It also provides a backup route in case of link failures. The primary route for packets is obtained by a shortest-path computation. Topology updates and route requests are restricted to the nodes comprising the backbone and a small subset of nodes not in the backbone; this reduces redundant transmissions as well as the accompanying overhead.

1.3 Summary of Existing Methods

The approach taken began with a literature search of theory and applications pertaining to the dominating set approach of routing in mobile ad hoc networks. Four main versions of this approach [8], [15], [18], [41] were identified and investigated. From this research, two aspects of the routing approach using dominating sets were identified. The first is the creation of the virtual backbone; this is accomplished by using a distributed algorithm that computes an approximation of a dominating set for the given network. The second is routing packets using this virtual backbone. Both aspects involve periodic re-computations of the backbone and must handle frequent node mobility.

Das et al's approach [18] proposes two distributed approximation algorithms [24] to construct a minimum connected dominating set (MCDS) that serves as a backbone in the network. The computation of the MCDS is based on a coloring process. Initially, all nodes are unmarked and colored white. The effective degree of a node is the number of its unmarked neighbors. A node is colored black, i.e. it joins the dominating set if it has the maximum effective degree compared to all nodes in its 2-hop neighborhood. The nodes that belong to the dominating set are then connected using a distributed minimum spanning tree algorithm as per the first proposed approximation algorithm, or by adding extensions to the current MCDS, as per the second proposed approximation algorithm. Das et al then propose a routing algorithm that will route through the MCDS thus computed.

Wu and Li propose a simple and efficient distributed algorithm [41] for calculating non-minimum connected dominating sets (CDS) using a marking process. While Das et al's algorithm first finds a dominating set and then connects this dominating set, Wu and Li's approach first finds a CDS and then removes certain nodes from the CDS via a selection process. Initially, all nodes are unmarked. Every node exchanges its neighbor set with all its neighbors. Every node now marks itself i.e. joins the dominating set if it has two unconnected neighbors. Wu and Li also propose a routing approach that routes all packets through the nodes that belong to the dominating set, or backbone, in the network.

Alzoubi et al propose an efficient distributed algorithm [8] to construct a connected dominating set (CDS) by building a rooted tree. The status of each node i.e. backbone or non-backbone node is then assigned based on the level of the node in the tree. Cheng et al propose two efficient distributed algorithms [15] to construct a CDS. The first algorithm is cost-aware and takes into consideration resource constraints in ad hoc wireless networks. The second algorithm is degree-aware and takes into consideration maximum effective degree, similar to the approach in [18]. The approaches in [8], [15] do not propose a specific routing approach to accompany the construction of the virtual backbone. For this study, we assume that the routing approach is as proposed by Das et al [18].

1.4 Approach

In all the existing approaches described in the Section 1.3, performance analysis of the dominating-set-based routing algorithms was based on graph theoretical background and/or simulations implemented using custom programs developed by the authors. Extensive theoretical analysis pertaining to the construction of the dominating set has been conducted for each proposed dominating-set-based routing approach. However, to the best of our knowledge, these routing approaches have not been implemented, evaluated or compared on a standard simulation platform. In particular, the dominating-set-based routing approaches proposed thus far in literature have not been evaluated or compared in terms of network routing performance.

This thesis is the first research effort to implement and analyze the routing performance of the protocols based on the dominating set approach using the network simulator ns-2 [1]. This simulator is targeted towards networking research and is the outcome of a collaborative research effort undertaken by UC Berkeley, USC-ISI, Xerox PARC and LBNL.

This thesis also proposes an efficient heuristic based on genetic algorithms [5] to construct a virtual backbone for an ad hoc wireless network and utilize this backbone to compute and update routes in the network. Unlike previous approaches, the proposed evolutionary approach takes advantage of the evolutionary nature of the underlying network: ad hoc wireless networks evolve as nodes move and change

positions. The backbone modeled by the dominating set must also be recomputed as the network changes, therefore the backbone also evolves. Using ns-2, a detailed routing performance comparison of the proposed evolutionary algorithm with the existing dominating set based routing protocols has been conducted.

The objectives of this thesis may be stated as follows:

- An in-depth study of existing dominating-set-based routing approaches in literature.
- A study of genetic algorithms and its applications in ad hoc wireless networks. This includes the design and development of a Scatternet formation protocol for Bluetooth networks based on genetic algorithms, implemented and evaluated using the network simulator ns-2.
- Implementation, detailed performance analysis and comparison of existing dominating-set-based routing approaches by means of extensive simulations conducted using the network simulator ns-2.
- Implementation and detailed performance evaluation of an evolutionary routing protocol that uses genetic algorithms to construct a minimum connected dominating set in ad hoc networks by means of extensive simulations conducted using the network simulator ns-2.
- Implementation and detailed performance evaluation of the impact of clustering nodes in a dominating-set-based routing approach by means of extensive simulations conducted using the network simulator ns-2.

Chapter 2

Problem Definition

Existing routing protocols for ad hoc networks employ “flooding” to disseminate control packets for topology updates and route discovery. Flooding is a broadcast mechanism that causes redundancy, contention and collision resulting in a high packet loss rate. The main problem is to design a routing protocol for mobile ad hoc networks that minimizes the flooding problem. This can be achieved by restricting broadcast to a subset of the mobile nodes in the ad hoc network, called the “virtual backbone”. This would minimize communication overhead and optimize the usage of critical resources in the network.

A virtual backbone may be constructed by approximating a *dominating set* that uses unicast to replace flooding. This backbone is used to compute and update routes as well as provide a backup route at the time of link failure. Since this backbone is dynamic, it must be recomputed periodically and this presents a challenging problem in the design and implementation of the routing protocol.

A secondary problem studied in this thesis involves the formation of multihop Bluetooth ad hoc networks, known as scatternets. In order to learn how to apply the genetic algorithm approach to wireless networks as well as learn how to use the network simulator ns-2, we began with the study of how genetic algorithms may be

used to solve a contemporary problem in Bluetooth personal area wireless networks: the problem of Scatternet formation. The knowledge gained from this work was further utilized to accomplish the primary goal of this thesis: to design an evolutionary routing protocol for ad hoc wireless networks.

In this chapter, Sections 2.1 and 2.2 refer to the main problem: routing in ad hoc wireless networks and Sections 2.3 and 2.4 refer to the secondary problem: scatternet formation in Bluetooth networks.

2.1 Problem Domain: Routing in Ad Hoc Wireless Networks

A mobile ad hoc network, or MANET, is a collection of mobile computers that form a temporary network without any pre-established infrastructure. Each mobile node has a wireless interface and communicates with other nodes using radio waves or infrared technology. Figure 1 shows a simple wireless ad hoc network comprising three nodes – A, B and C. Nodes A and C are not within transmission range of each other. However, node B acts as the intermediate router and forwards packets between A and C. This type of communication wherein intermediary nodes forward packets sent from a source mobile node to a destination mobile node is called *multihop* communication, and is one of the primary characteristics of a MANET. As mentioned in the introduction, other characteristics of MANETs that present challenges in the design of routing protocols for such wireless environments include: dynamic topology, limited energy and bandwidth resources and security vulnerabilities.

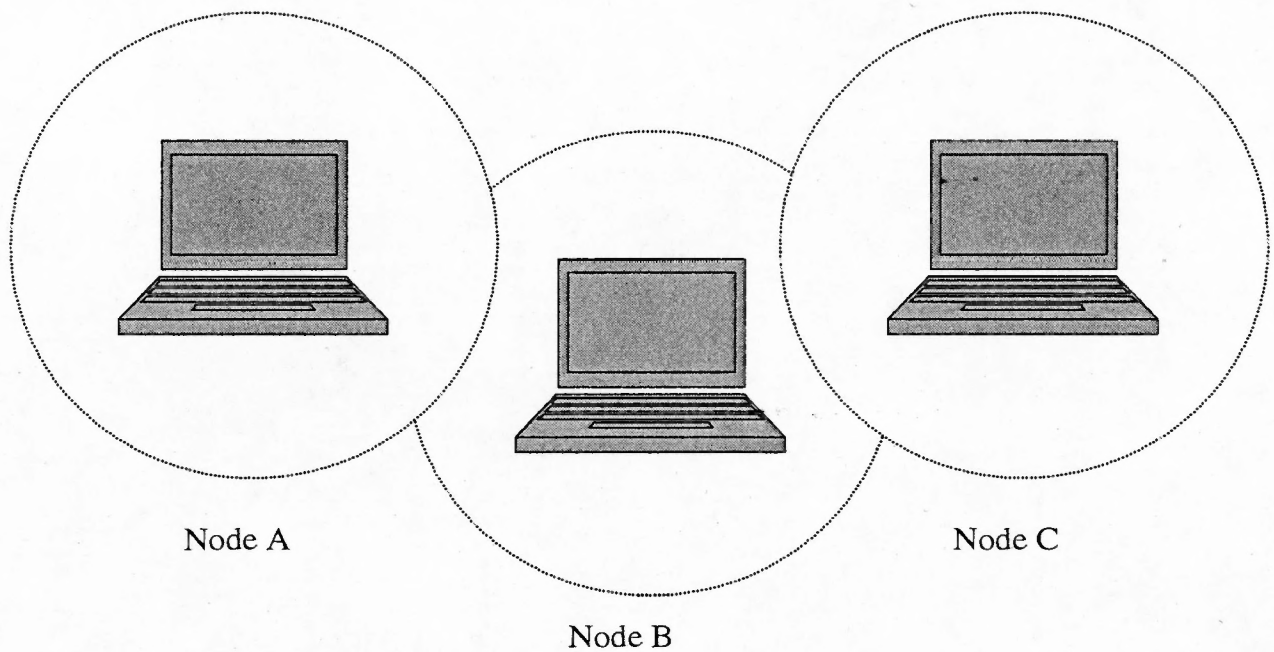


Figure 1 – Ad hoc network with three participating mobile nodes

As shown in Figure 2, routing protocols for MANETs may be classified as: *proactive* (table-driven) and *reactive* (source-initiated or on-demand), depending on how they react to network topology changes [35].

Proactive, or *table-driven*, protocols attempt to maintain consistent, up-to-date routing information at every node in the network. Such protocols require every node to maintain at least one table that stores routing information. Table-driven protocols respond to changes in network topology by propagating periodic updates through the network in order to maintain a consistent view of the network. This ensures that a route to every node is always available, whether it is needed or not. A table-driven

routing approach is analogous to a connectionless approach of forwarding packets with no regard to when the routes are actually required. This characteristic of table-driven protocols incurs substantial control overhead and power consumption. This can be a serious limitation of this category of protocols, since bandwidth and energy are constrained resources in MANETs. Examples of proactive protocols include Wireless Routing Protocol (WRP) [30] and Destination Sequenced Distance Vector (DSDV) [33].

In contrast, *reactive*, or *on-demand*, or *source-initiated*, protocols create routes only when desired by the source node. Therefore, control overhead is greatly reduced in this approach when compared to the table-driven approach. However, a route to every node is not always available and a node must wait until the route it desires has been discovered. When a node requires a route to a destination, it initiates a route discovery process by propagating route requests throughout the network. This process is completed when a route is found or when all permutations have been exhausted. Once a route to a particular destination is found, it is maintained by a route maintenance procedure. The reactive approach is analogous to connection-oriented communications and is based on the observation that (i) routes in a dynamic topology expire frequently, and (ii) not all the routes are used at the same time. Examples of reactive protocols include Ad Hoc On-Demand Distance Vector Routing (AODV) [34], Dynamic Source Routing (DSR) [13], Temporally Ordered Routing Algorithm

(TORA) [32] and Signal Stability-Based Adaptive Routing (SSA) [22].

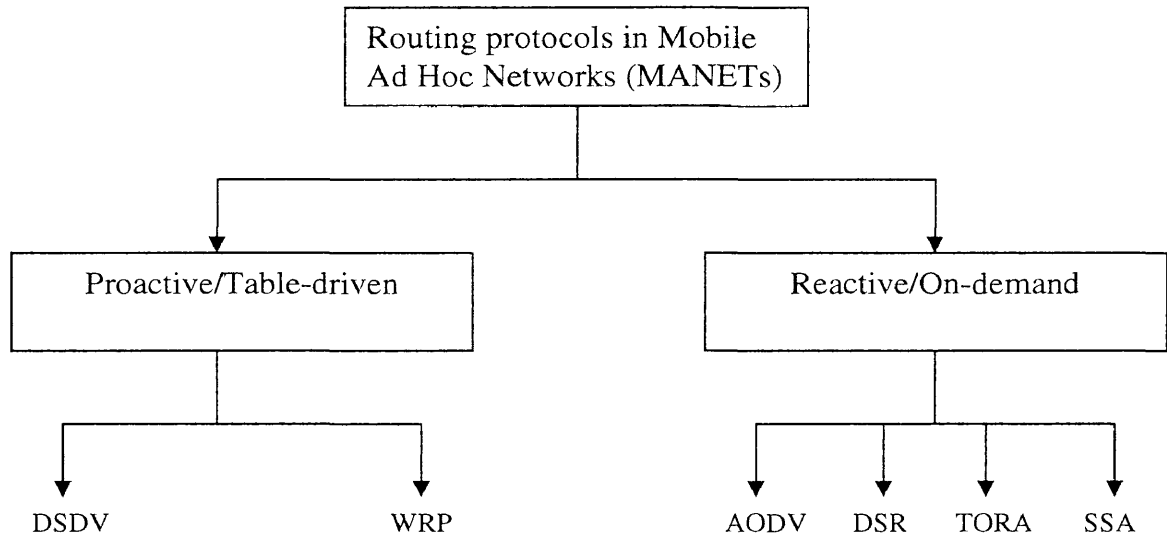


Figure 2 – Classification of routing protocols for Mobile Ad Hoc Networks (MANETs)

Both proactive as well as reactive routing protocols for ad hoc networks employ “flooding”, a broadcast mechanism, to disseminate control packets for topology updates and route discovery. Flooding results in high protocol overhead and has been proven to cause the *broadcast storm problem* [31] i.e. it causes redundant transmissions, contention because of neighboring nodes trying to forward the message that they just received to their neighbors, and collision because of the absence of RTS-CTS-DATA-ACK exchange in broadcasts. Flooding has also been shown to be unreliable by means of extensive performance evaluations of broadcast storms in ad hoc networks [27]; not all hosts receive broadcast messages even in a collision-free environment.

Therefore, we see that a routing protocol that minimizes flooding by restricting broadcasts to the members of the virtual backbone (or dominating set) is likely to be robust and route packets efficiently through the network. The dominating-set-based routing approach may be classified as a hybrid approach between the table-driven approach and the on-demand approach, wherein the nodes that constitute the dominating set maintain routing tables and the nodes outside the dominating set request for routes as required from the members of the dominating set.

2.2 Problem Definition: Routing in Ad Hoc Wireless Networks using a Virtual Backbone

As described in the introduction, the problem involves the design of an efficient routing strategy in an ad hoc wireless network comprising dynamic mobile nodes. Imposing a *virtual backbone* structure on the ad hoc wireless network replaces flooding with unicast packet-forwarding mechanisms and this backbone may be utilized to route packets efficiently through the network.

The virtual backbone may be constructed by approximating a *dominating set* in the graph underlying the network. This dominating set may be connected and/or minimum. The main function of the backbone is to compute and update routes; topology updates are restricted to the subset of nodes that constitute the backbone. Routes are obtained by shortest path computations. The backbone may also provide

backup routes in case of link failures. An important characteristic of a virtual backbone is its dynamic nature; it must be recomputed periodically during the routing process.

The underlying network model is described as follows: we assume that a given ad-hoc wireless network has 'n' hosts. Each host has an omni-directional antenna; thus, the host transmits in all directions and the transmission range of a host is a disk. Every mobile host behaves as a router and has a fixed transmission range known as its *neighborhood*. Two hosts within range of each other can communicate with each other, and are said to be *neighbors*. Each host may be powered on or powered off. All hosts use a common wireless channel as a communication medium. In this paper, communication is assumed to be bi-directional, although in practice it may be unidirectional, due to the *hidden terminal* problem.

In graph theoretic terminology, the network topology may be described as a *unit-disk* graph $G = (V, E)$, where V is the set of nodes (hosts) and E is the set of edges (links), in which each mobile node has the same communication range, $R=1$, and a link between two mobile nodes exists if and only if their distance is at most one. The graph underlying the network consists of nodes that represent the active, powered-on mobile hosts in the network.

The neighborhood of u is $N(u)$ and u 's degree is $\delta(u)$, the number of neighbors of u . $N_2(u)$ is the two-hop neighborhood of u ; it contains u , u 's neighbors and their neighbors. A *dominating set* S of G is a subset of V : each node u in V is either in S or has a neighbor in S . In Figure 1, $S = \{C, E\}$. A *connected dominating set* C of G induces a connected subgraph of G . A *minimum connected dominating set* C^* is the connected dominating set with minimum cardinality. In Figure 3, $C^* = \{C, D, E\}$

A node u in C^* is an MCDS node; an edge connecting two MCDS nodes u and v is an MCDS edge. Each u not in C^* has a neighbor v in C^* that is its dominator i.e. v is $\text{dom}(u)$. Each v in C^* has either a neighbor w in C^* such that w is $\text{dom}(v)$, or has no dominator at all.

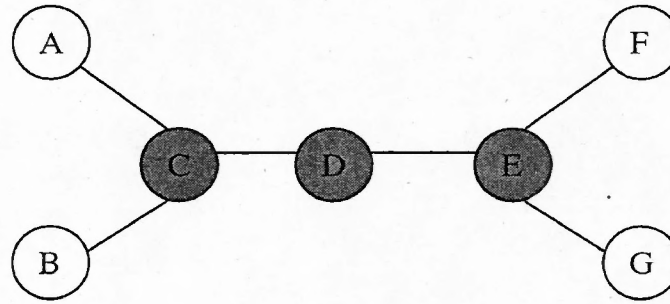


Figure 3 - Dominating Set

For a general graph G , finding the MCDS is a computationally intensive problem that is considered NP-hard [23]. In this thesis, we attempt to devise a polynomial-time

heuristic based on a genetic algorithm that finds an approximation to an MCDS. The dominating set thus generated may be utilized as a backbone in order to route packets efficiently through the network.

2.3 Problem Domain: Bluetooth Networks

Bluetooth [2], conceived initially by Ericsson, is a standard for a small, inexpensive radio chip to be plugged into computers, printers, mobile phones and an array of other devices that constitute a Personal Area Network (PAN). A Personal Area Network is technology that enables the interconnection of information technology devices within the range of an individual using wireless technology. Thus, Bluetooth is likely to become an important platform for wearable computing [11] and ad hoc networking [26], by means of which devices can establish connections anytime, anywhere, without the aid of centralized infrastructure. For example, in a conference room, participants can share documents and exchange business cards using their Bluetooth-enabled mobile phones and handheld computers.

The Bluetooth technology operates in the unlicensed 2.4 GHz Industrial, Scientific and Medical (ISM) band and uses a frequency hopping Time Division Duplex (TDD) scheme for transmission. Bluetooth devices share 79 channels of 1 MHz bandwidth within the ISM band. On the channel, each packet is transmitted at a different frequency. Frequency hopping facilitates the co-existence of several piconets that can

independently communicate in close proximity without affecting the performance significantly. The maximum bandwidth possible is 1 Mbps (Megabit per second).

Up to 8 Bluetooth devices (1 Master and 7 slaves, as per Bluetooth 1.0b specifications) can form a centralized network called a *piconet*, controlled by a *master* node that allocates transmission slots to all the other nodes, called *slave* nodes, in the piconet. Any Bluetooth device can perform the role of a master/slave node. In a *scatternet* topology, when piconets are connected via shared slave nodes, the shared slave node is also known as a *bridge* node (Figure 4). All packets are exchanged between a master and its slaves within a piconet; there is no master-master or slave-slave communication. A Bluetooth device may be a slave in several piconets but is a master in only one piconet.

2.4 Problem Definition: Scatternet Formation in Bluetooth Ad Hoc Networks

Bluetooth technology can provide much more beyond cable-free connectivity amongst a small number of devices in a piconet. Bluetooth may be extended to interconnect multiple piconets to form a *scatternet*. In Figure 4, master **M1** and the slaves within its transmission range form one piconet; master **M2** and the slaves within its range form another piconet. The bridge node **b** is a slave node that is within transmission range of both M1 and M2 and therefore, the node 'b' is slave to both masters M1 and M2. Together, the two piconets interconnected by the bridge node 'b', constitute a scatternet. Bridge nodes participate in multiple piconets; a bridge can

be a master node in one piconet and a slave node in another (M/S bridge) or a slave node in both piconets (S/S bridge, as shown in Figure 4).

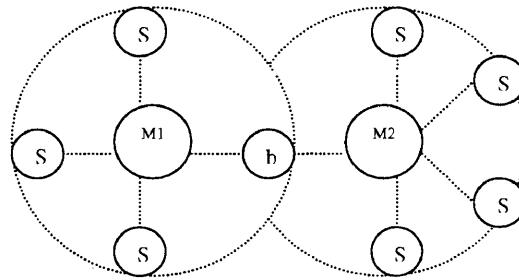


Figure 4 – Piconets and Scatternet

M1, M2 – Master nodes

S – Slave node

B – Bridge node or shared slave

A scatternet may consist of hundreds of devices and multi-hop scatternets can provide connectivity over distances greater than that typical of short-range radio technology, for e.g. across a football field, industrial plant or to support tactical operations in a battlefield.

The performance of Bluetooth depends largely on the scatternet topologies. A scatternet can be viewed as a Bluetooth ad hoc network that is formed by interconnecting piconets. The problem involves designing a scatternet formation protocol with the following desirable properties: 1) the scatternets are connected, i.e. every Bluetooth device can be reached from every other device, 2) piconet size is

limited to eight nodes to avoid “parking” of slaves and the associated overhead, 3) the number of piconets is close to the universal lower bound that defines the optimal number of piconets, resulting in low interference amongst piconets, and 4) end-user delay is minimized during scatternet formation.

There exist several methods to construct a Bluetooth scatternet topology [12]. The configuration of a scatternet [29] influences the performance of the network considerably. Therefore, it is important to devise an efficient scatternet construction protocol and fully utilize the capabilities of Bluetooth ad hoc networking.

As part of this thesis, we study the problem of scatternet formation amongst Bluetooth devices that are within range of each other. We propose and evaluate an evolutionary approach to scatternet construction, wherein we use a genetic algorithm to determine the topology of the Bluetooth network prior to data exchange in the network.

Chapter 3

Background Information

The work for this thesis began by conducting an extensive literature search in the area of routing approaches in mobile ad hoc networks (MANETs) that use the graph theoretic concept of minimum connected dominating sets to construct a virtual backbone in the network. This search provided detailed information on different approaches and models that have been applied to solve the problem of virtual backbone construction and routing. These approaches were studied carefully and classified into four main categories based on the backbone construction algorithm used. The algorithms presented by these dominating-set-based routing approaches were carefully analyzed in order to determine how they may be implemented and evaluated.

A new evolutionary dominating-set-based routing approach based on genetic algorithms was then designed. In order to study and test the use of genetic algorithms in the field of wireless networks, a Scatternet formation protocol for multihop Bluetooth ad hoc networks based on genetic algorithms was designed and implemented. This chapter presents existing routing approaches based on dominating sets as well as background information on genetic algorithms. Scatternet formation in Bluetooth networks will be discussed in Chapter 4. The new evolutionary approach to dominating-set-based routing in MANETs is discussed in detail in Chapter 5.

Existing literature pertaining to dominating-set-based routing approaches in MANETs includes different algorithms used to construct the virtual backbone as well as algorithms used in the routing process. The algorithms presented in Sections 3.1-3.4 use the concept of dominating sets to construct and utilize a virtual backbone.

3.1 Das et al's Algorithm

This approach [18] proposes two distributed approximation algorithms to compute the MCDS, based on two centralized algorithms given by Guha and Khuller [24]. Das et al then propose a routing algorithm using the MCDS.

Distributed Approximation Algorithm I:

Stage 1 – This stage finds a dominating set S . Initially, let all the nodes in G be unmarked (white). The number of unmarked neighbors of a node u is its effective degree $\delta^*(u)$. Node u collects the effective degree of all nodes present in its two-hop neighborhood, $N_2(u)$. u joins the dominating set S if its effective degree is greater than the effective degree of all nodes in $N_2(u)$.

Stage 2 – This stage connects the dominating set S . Each component formed by the $(u, \text{dom}(u))$ edges is labeled; the components are now connected using a minimum spanning tree algorithm. The edges in G that connect two nodes in the same component are discarded; the remaining edges are weighted to favor those that will

not increase the size of the dominating set too much. The minimum spanning tree of G is now computed.

Distributed Approximation Algorithm II:

Stage 1 – This stage remains the same as Stage 1 of Algorithm I.

Stage 2 – This stage adds extensions to the current MCDS fragment. An extension to the fragment is either a one or two-edge path consisting of one node in the fragment, and one or two nodes not in the fragment. The *effective combined degree* of an extension is the number of unmarked nodes adjacent to the non-fragment nodes in the extension [18]. The best extension is the one with the highest effective combined degree; this is added to the current MCDS fragment. The process is repeated until the dominating set is connected.

These approximation algorithms have been used to compute the MCDS or *Spine* in [19], [20] and [21].

The MCDS routing algorithm proposed in [18] consists of the following steps to determine routes:

1. Compute the MCDS C .
2. Collect topology information transmitted from non-MCDS nodes (dominatees) to MCDS nodes (dominators).
3. Broadcast global topology to all dominators.

4. Each dominator runs an all-pairs shortest paths algorithm on its local copy of the topology.
5. Propagate routing tables out to the dominatees.
6. Every t seconds, dominators broadcast topology to all nodes, providing a periodic maintenance update.

This routing approach is used in [19], [20] and [21] with slight modifications. In [19], it is proposed that instead of periodic topology updates, the dominators should unicast the topology only on request from dominatees. This saves time and messages per update. In [21], the authors suggest that only partial topology information should be maintained at each dominator; this results in good routes at low overhead.

Example [18]: The steps involved in the routing process using the MCDS may be illustrated by the example in Figure 5. After Step 1, the MCDS consists of nodes 3, 5 and 6. Node 3 dominates 1, 2, 4 and 5. Node 6 dominates 7, 8 and 9. After Steps 2 and 3, each node in the MCDS constructs its local copy of the topology. Using these local copies, Nodes 3, 5 and 6 compute shortest paths in Step 4. In Step 5, 3 and 6 transmit routing tables to the non-MCDS nodes. In addition to finding routes, the MCDS can also provide backup routes for temporary use while shortest paths are being updated. For example, in Figure 5, if nodes 2 and 4 move apart then the MCDS can provide the backup route $\langle 2, 3, 4 \rangle$.

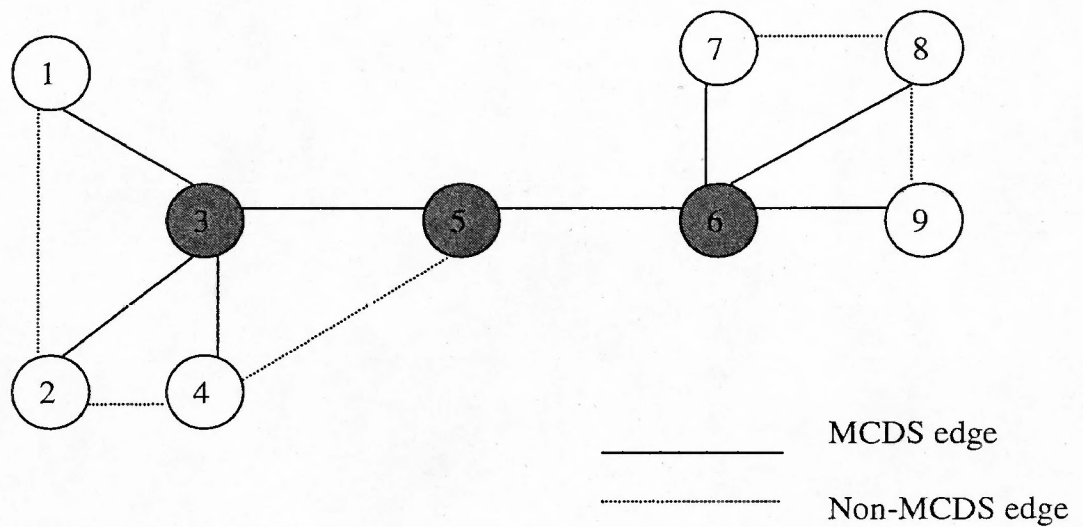


Figure 5 – Basic MCDS Routing

3.2 Wu and Li's Algorithm

This approach [41] proposes a simple and efficient distributed algorithm for calculating non-minimum connected dominating sets (CDS) as well as a dominating-set-based routing approach. While Das et al's algorithm first finds a dominating set and then connects this dominating set, Wu and Li's approach first finds a CDS and then removes certain nodes from the CDS via a selection process.

In this approach, dominators are referred to as *gateway* hosts and dominatees are referred to as *non-gateway* hosts. The computation of a virtual backbone is based on a marking process that marks every node in a given connected and simple graph $G = (V, E)$. Each node may be either *marked* or *unmarked*.

Initially, let all nodes in G be unmarked. Every node v exchanges its open neighbor set (all neighbors excluding v) with all its neighbors. Every node v now marks itself if it has two unconnected neighbors; v is now a gateway host.

To reduce the size of the connected dominating set generated by this marking process, this algorithm proposes two rules based on unique node ID [41]:

Rule 1:

Consider two vertices v and u in the induced graph G' (the graph of nodes that form a dominating set of G , derived from the marking process). If the closed neighbor set of v (all neighbors including v) is covered by the one of u , v can be unmarked if its node ID is smaller than that of u .

Rule 2:

Consider two vertices u and w that are marked neighbors of gateway host v in G' . If the open neighbor set of v is covered by the open neighbor set of u and w , and if v has the minimum node ID of the three nodes, then v can be unmarked.

This marking process has been used to construct the virtual backbone in [42] with extended rules for selective node removal, based on node degree and energy level.

The dominating-set-based routing algorithm proposed in [41] consists of the following steps to determine routes:

1. Each gateway host v maintains a *gateway domain membership list* (non-gateway hosts adjacent to v) and a *gateway routing table* (all gateway hosts and their domain membership lists).
2. If the source is not a gateway host, it forwards the packets to an adjacent source gateway.
3. The source gateway routes the packets in the induced graph generated from the connected dominating set.
4. The packets are routed to a destination gateway; this could be the destination host or the gateway of the destination host. The gateway then forwards the packets to the destination host.

Example [42]: In Figure 6, the open neighbor set of u , $N(u) = \{v, y\}$, $N(v) = \{u, w, y\}$, $N(w) = \{v, x\}$, $N(y) = \{u, v\}$ and $N(x) = \{w\}$. After each node exchanges its open neighbor sets with its neighbors, node u has $N(v)$ and $N(y)$, node v has $N(u)$, $N(w)$ and $N(y)$, w has $N(v)$ and $N(x)$, y has $N(u)$ and $N(v)$, and x has $N(w)$. The nodes v and w have unconnected neighbors, therefore they are marked as “gateway hosts”, or members of the dominating set.

The gateway domain membership list for the gateway host v is $\{u, y\}$ and for gateway host w the only member on the list is node x . The gateway routing table consists of a set of entries for each gateway, along with its membership list. For node v , the gateway routing table entry is $w(x)$ and for node w , the gateway routing table is $v(u, y)$. If node u wishes to send a packet to node x , it first sends the packet to its gateway host, node v . Node v then looks up the destination node and forwards the packets to the destination gateway, node w . Node w then forwards the packet to the destination node, node x .

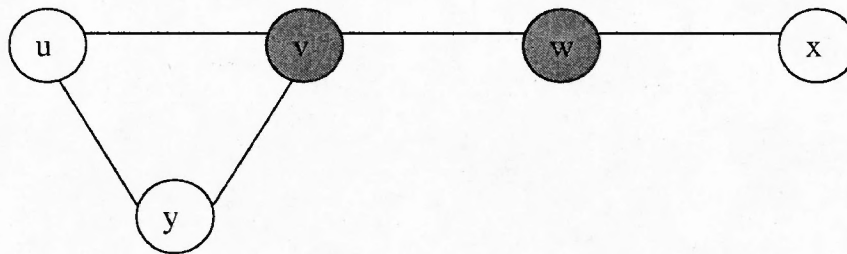


Figure 6 – Wu and Li's CDS Routing

3.3 Alzoubi et al's Algorithm

This approach [8] proposes an efficient distributed algorithm to construct a connected dominating set (CDS) in two stages; the first stage constructs a *maximal independent set* (MIS) and the second stage constructs a *dominating tree*.

The computation of the CDS that constitutes the virtual backbone proceeds in two stages:

Stage 1 – This stage constructs an MIS. A distributed leader election algorithm [16] is applied to construct a rooted spanning tree T rooted at node v . Each node then identifies its graph distance, or tree *level*, from the root. The root first announces its level (0). This level announcement message propagates through the tree. Each node calculates its own level and also records the levels of its neighbors. Each leaf node transmits a LEVEL-COMPLETE message to its parent. When all nodes have reported LEVEL-COMPLETE and the root receives this message from all its children, each node is then ranked using an ordered pair (level, node ID). These ranks are sorted in lexicographic order; therefore, the root has the lowest rank at level 0.

Once the rooted spanning tree has been constructed, the MIS is constructed. Each node that has the lowest rank amongst its neighbors declares itself to be a dominator and broadcasts this message. When a node receives a dominator message for the first time, it declares itself to be a dominee and broadcasts this message. If a node receives dominee messages from all of its neighbors with lower ranks, it declares itself to be a dominator. This marking process continues until the leaf nodes are all marked and the root receives MIS-COMPLETE messages from all other nodes in the tree.

Stage 2 – This stage constructs a dominating tree that spans all the dominator nodes such that all nodes in this tree form a CDS. Initially, the root joins the dominating tree. When each dominator node joins the dominating tree, it invites all dominator

nodes that are two hops away by communicating through the intermediate dominatee nodes. Each dominator joins the tree when it receives the invitation for the first time, along with the dominatee node that relayed the invitation. This process is repeated until all dominator nodes are in the tree.

This approach focuses on the construction of a CDS and does not propose a specific routing approach to accompany the construction of the virtual backbone. For our study, we assume that the routing approach is as outlined in Section 3.1.

Example: Figure 7 shows a rooted spanning tree rooted at node 1 that is constructed from the graph underlying the ad hoc wireless network by applying a distributed leader election algorithm. The construction of the MIS proceeds as follows: Each node identifies its level with respect to the root (node 1). The level of the root itself is 0. The levels of 2, 3, 4, 5 and 6 are 1, 1, 2, 2 and 2 respectively. The ranks of nodes 1, 2, 3, 4, 5 and 6 represented by the ordered pair (level, node id) are (0, 1), (1, 2), (1, 3), (2, 4), (2, 5) and (2, 6) respectively. Node 1 has the lowest rank and declares itself to be a dominator node, marking itself black. Nodes 2 and 3 receive the dominator message from Node 1 and declare themselves to be dominatee nodes, marking themselves gray. Nodes 4, 5 and 6, upon receiving the dominatee messages from their lower ranked neighbors, declare themselves to be dominator nodes. This completes the process of MIS construction.

The second phase constructs a dominating tree that spans all black nodes; the black nodes in this tree structure form the CDS. Node 1 joins the dominating tree first and invites all black nodes two hops away, i.e. Nodes 4, 5 and 6. In response, Nodes 4, 5 and 6 join the tree together with the gray nodes 2 and 3. Therefore, Nodes 1, 4, 5 and 6 constitute the CDS.

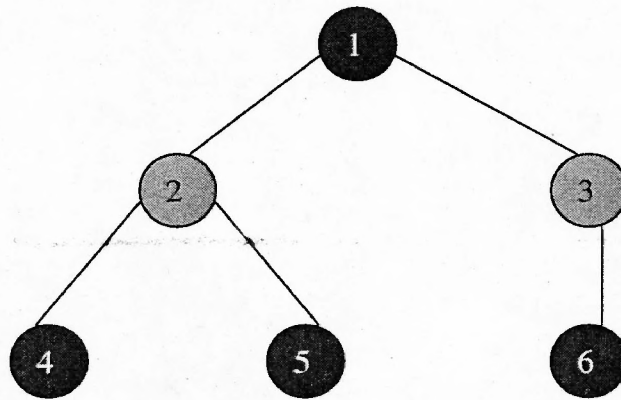


Figure 7 – Alzoubi et al's CDS Construction

3.4 Cheng et al's Algorithm

This approach [15] proposes two efficient distributed approximation algorithms to compute a CDS.

Distributed Approximation Algorithm I:

This algorithm is cost-aware, taking into account the scarcity of resources in wireless networks. Initially, a leader is elected. Then, this algorithm proceeds through a series of steps to compute a tree rooted at the leader. Each mobile node is associated with a

cost (host id or inverse of residual power or load) and each node is ranked, depending on its level in the tree. The nodes may be classified into four states: white (active or inactive), gray (dominatee) and black (dominator). Active white nodes are those that have at least one neighbor that is a dominatee.

Initially, let all nodes be unmarked (white, inactive). The leader begins the algorithm, declares itself to be a dominator and broadcasts this message. A white node u that receives dominator messages from node v chooses either v to be its dominator, or $\text{dom}(v)$ to be its dominator. An active white node u chooses v as its dominator if it receives a dominatee message from v and v has a low cost. If there is no broadcast for a time period t and if an active white node u has the smallest (cost, id) value amongst u 's active white neighbors, then u declares itself to be a dominator and broadcasts this message. If u is a dominatee node and receives a dominator message from v , where $u = \text{dom}(v)$ then u declares itself to be a dominator and broadcasts this message. If u is a dominator node and none of its neighbors select it to be a dominator, then it declares itself to be a dominatee and broadcasts this message. This algorithm terminates when there are no white nodes remaining. The set of black nodes form the CDS.

Distributed Approximation Algorithm II:

This algorithm is degree-aware and thus, performs better than Algorithm I.

Stage 1 – This stage computes a maximal independent set (MIS). Initially, let all nodes be unmarked (white inactive). Each mobile node keeps track of its effective

degree (number of white neighbors) and each node is ranked, depending on its level in the tree. A leader is elected from amongst the inactive white nodes, declares itself to be a dominator and broadcasts this message. A white node u that receives dominator messages from node v chooses v to be its dominator. A white node u may receive active and dominee messages; u responds by updating its list of active white neighbors and effective degree of its neighbors respectively. Active white nodes keep track of all dominee messages being broadcast in the network, in order to update the effective degree and rank accordingly. If there is no broadcast for a time period t and if an active white node u has the biggest (effective degree, id) value amongst u 's active white neighbors, then u declares itself to be a dominator and broadcasts this message. Every dominee node keeps track of its *blackdegree*, the number of higher rank black neighbors. Every dominator node u keeps track of its dominee neighbors with lower rank and highest value of (blackdegree, id); these dominee neighbors are candidate dominators for u in Stage 2 of this algorithm.

Stage 2 – This stage designates dominators for all dominator nodes. Each dominator u selects its lowest rank dominee neighbor v with the highest value of (blackdegree, id) to be $\text{dom}(u)$. If u is a dominee node and receives a dominator message from v , where $u = \text{dom}(v)$ then u declares itself to be a dominator and broadcasts this message. If u is a dominator node and none of its neighbors select it to be a dominator, then it declares itself to be a dominee and broadcasts this message. The algorithm terminates when all nodes have been assigned dominators.

This approach focuses on the construction of a CDS and does not propose a specific routing approach to accompany the construction of the virtual backbone. For our study, we assume that the routing approach is as outlined in Section 3.1.

Example [15]: This example demonstrates how to apply Algorithm I to compute a connected dominating set. The given unit-disk graph G is shown in Figure 8. There are 9 nodes and 12 links. Node 0 is the leader. We assume the cost is the node id.

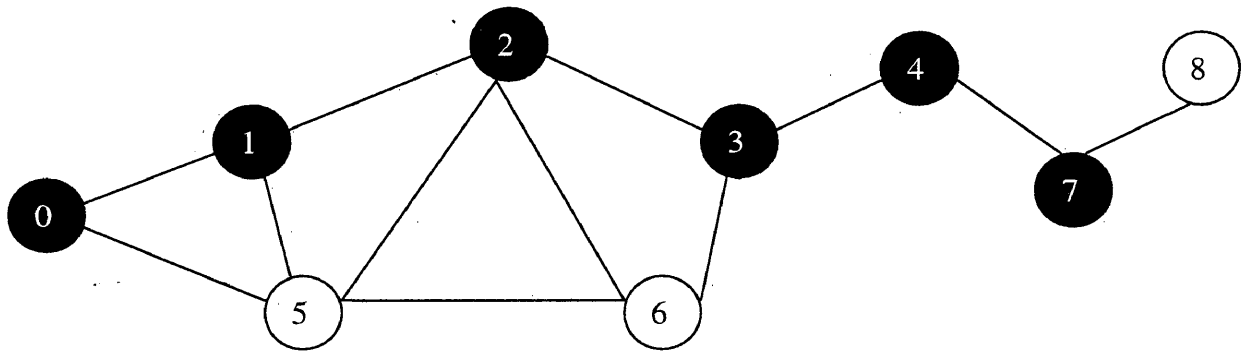


Figure 8 – Cheng et al’s CDS Construction – Algorithm I

For this example, 4 steps are needed to compute the connected dominated set:

1. Node 0 declares itself to be a dominator. Nodes 1 and 5, in response to Node 0’s broadcast, declare themselves to be dominee nodes. Nodes 2 and 6 then declare themselves to be active nodes, since they are now neighbors of the dominee nodes 1 and 5 respectively.

2. Between the two active white nodes 2 and 6, node 2 has a smaller cost (node ID). Therefore, node 2 declares itself to be a dominator and selects node 1 to be its dominator, since node 1 has a smaller cost compared to node 5. Node 1 then declares itself to be a dominator and selects node 0 as its dominator. Nodes 3 and 6 then declare themselves to be dominatee nodes. In response to node 3's dominatee status, node 4 then declares itself to be active. Nodes 0, 1 and 2 constitute the dominating set at this stage.
3. Node 4 is the only active node and declares itself to be dominator; it selects node 3 as its dominator. In response, node 3 declares itself to be dominator. Node 7 then declares itself to be a dominatee, rendering node 8 active. Nodes 0, 1, 2, 3 and 4 constitute the dominating set at this stage.
4. Node 8 being the only active node, it declares itself to be a dominator and selects node 7 as its dominator. Node 7 then declares itself to be a dominator. Node 8 detects that its children list is empty and declares itself as a dominatee, selecting node 7 as its dominator. The algorithm now terminates with nodes 0, 1, 2, 3, 4 and 7 as members of the dominating set.

Here is an example of how Algorithm II constructs a CDS. Stage 1 contains 4 steps:

1. Node 0 declares itself to be a dominator. Nodes 1 and 5, in response to Node 0's broadcast, declare themselves to be dominatee nodes. Nodes 2 and 6 then declare themselves to be active nodes and broadcast their effective degrees, since they are now neighbors of the dominatee nodes 1 and 5 respectively.

2. Between the two active white nodes 2 and 6, node 6 has a larger value of (effective degree, node id). Therefore, node 6 declares itself to be a dominator. Nodes 2 and 3 then declare themselves to be dominatee nodes. In response to node 3's dominatee status, node 4 then declares itself to be active. Nodes 5, 1 and 2 broadcast their blackdegrees of 1, 0 and 0 respectively.
3. Node 4 is the only active node and declares itself to be dominator. In response, Node 7 declares itself to be a dominatee, selecting node 4 as its dominator and rendering node 8 active. Node 3 broadcasts its blackdegree of 1.
4. Node 8 being the only active node, it declares itself to be a dominator. Node 7 then declares its blackdegree of 1.

After Stage 1, the dominator nodes are 0, 6, 4 and 8. Stage 2 contains the following steps:

1. Node 6 declares itself to be a dominator after receiving the blackdegree message from node 5. In response, node 5 declares itself to be a dominator and node 2 selects node 5 as its dominator.
2. Node 4 declares itself to be a dominator after receiving the blackdegree message from node 3 and also selects node 3 as its dominator.
3. Node 8 declares itself to be a dominator after receiving the blackdegree message from node 7 and also selects node 7 as its dominator. Since node 8 does not dominate any nodes, it declares itself to be a dominatee and the algorithm

terminates. The final connected dominating set consists of the nodes 0, 5, 6, 3, 4 and 7, as shown in Figure 9.

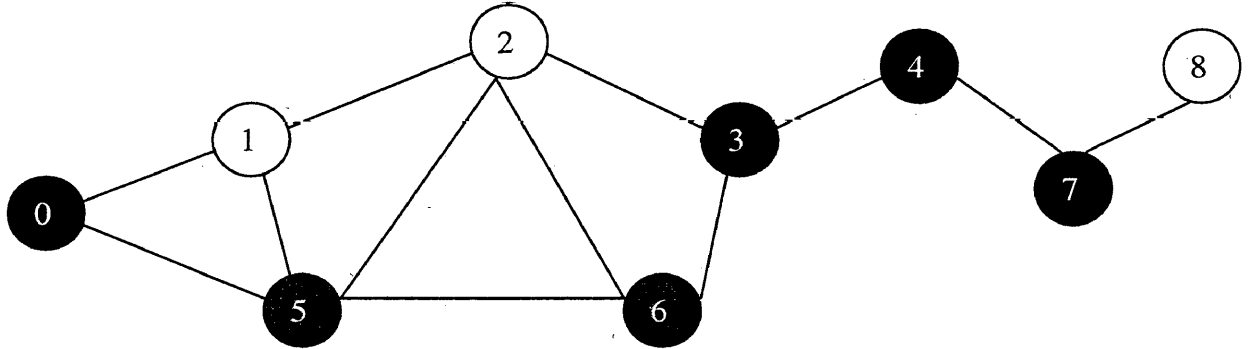


Figure 9 – Cheng et al’s CDS Construction – Algorithm II

Although theoretical performance evaluations of all the dominating-set-based routing protocols surveyed in this chapter have been researched and documented thoroughly, performance evaluations based on simulations have not been addressed in detail. In particular, a performance comparison with respect to the routing performance across the different dominating-set-based routing algorithms has not been specifically addressed. Also, genetic algorithms have not been utilized in the routing process in ad hoc wireless networks thus far. These factors provided a good opportunity to make an original contribution to existing literature through this thesis.

3.5 Introduction to Genetic Algorithms

A genetic algorithm is a stochastic, directed search algorithm that has proved useful in finding global optima in both static and dynamic environments. Genetic algorithms work with large populations of randomly generated solutions that are repeatedly subjected to selection pressure and which undergo naturally occurring genetic operations, such as selection, crossover and mutation, in order to find improved solutions.

Genetic algorithms, introduced by John Holland in the 1960s, seek to emulate the biological process of evolution; this is the driving process in the creation of complex organic structures that are well adapted to their dynamic environment. An individual in a population competes with other individuals for food and shelter and is affected by environmental conditions, such as climate. The result is that an individual that is better suited to its environment, or “fitter”, has a higher probability of surviving longer and generating more offspring, which carry its genetic information into future populations. This eventually leads to the “survival of the fittest”: a population of individuals with above average fitness than the previous populations.

Starting with an initial, randomly chosen population of strings, the genetic algorithm proceeds in a sequence of steps, as outlined below [5]:

1. **[Start]** Generate random population of n chromosomes (suitable solutions for the problem)

2. **[Fitness]** Evaluate the fitness $f(x)$ of each chromosome x in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete
 1. **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
 2. **[Crossover]** With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
 3. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
4. **[Accepting]** Place new offspring in a new population
5. **[Replace]** Use new generated population for a further run of algorithm
6. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
7. **[Loop]** Go to step 2

A genetic algorithm represents a solution as a string (or chromosome), usually binary. The selection operation determines which strings will be utilized as parent strings for the next generation. These strings are chosen such that the “fitter” strings, as determined by the fitness function, have a higher probability of being selected. The crossover operation produces two offspring by breaking the two parent strings into two smaller substrings at the same string location, and then recombining the substrings such each now fits with a substring from the other parent. The mutation

operation occurs with low probability and changes the value of a string position; this is used to maintain population diversity.

The sequence of steps in a genetic algorithm may be described in detail as follows: a genetic algorithm begins with a set of strings. This set is known as a “population” and each string is considered an individual member of the population. The string values are randomly initialized and a “fitness function” is used to evaluate the fitness of each string. This fitness value is used by the selection operation to choose the “parent” strings for reproduction. Selection, crossover and mutation operations are then applied to the strings. The mutation operation serves to maintain population diversity, which prevents premature convergence on a sub-optimal solution. These steps continue in an iterative manner until the population converges to a solution or a certain number of generations is reached. Although this algorithm does not guarantee an optimal solution, it searches the solution space extensively and is not likely to be deceived by local optima. The algorithm is simple, yet powerful because of the parameter choices that can be efficiently represented by a string.

Chapter 4

An Evolutionary Bluetooth Scatternet Formation Protocol

Wireless technologies such as 802.11 do not impose topology constraints on the network; however, Bluetooth imposes certain constraints for constructing valid topologies. The performance of Bluetooth depends largely on these topologies. This chapter presents and evaluates the performance of a new evolutionary scatternet topology construction protocol for Bluetooth networks.

A scatternet can be viewed as a Bluetooth ad hoc network that is formed by interconnecting piconets. The scatternets formed have the following properties: 1) the scatternets are connected, i.e. every Bluetooth device can be reached from every other device, 2) piconet size is limited to eight nodes to avoid “parking” of slaves and the associated overhead, 3) the number of piconets is close to the universal lower bound that defines the optimal number of piconets, resulting in low interference amongst piconets, and 4) end-user delay is minimized during scatternet formation.

Chapter 2 describes the problem domain and definition pertaining to scatternet formation in Bluetooth networks. This chapter describes existing approaches to

scatternet topology construction, the proposed evolutionary scatternet protocol and the performance evaluation of this protocol.

4.1 Related Work

The scatternet formation algorithms proposed in literature may be broadly classified into two categories: the first category comprising [7], [28], [29], [36], [40], [44] and our algorithm assumes that all Bluetooth devices are within transmission range of each other and the second category [10], [38], [43], [45], propose protocols for networks where the devices are not necessarily one hop away from each other.

Aggarwal et al. [7] introduce a two-phase scatternet algorithm that partitions the network into independent piconets, and then elects a super-master that knows about all the nodes. However, in the resulting topology, the piconets are not interconnected. A reorganization process interconnects the piconets.

Miklos et al. [29] use a statistical approach to investigate the performance of scatternets. They propose heuristics to generate scatternets with certain desirable properties, such as the number of piconets, and evaluate the performance of these scatternets through simulations.

Salonidis et al. [36] propose a two-phase Bluetooth Topology Construction Protocol (BTCP) for scatternet formation. In this approach, a coordinator is elected in the first phase such that this coordinator has complete knowledge of all devices. The coordinator then determines the roles of each node in the network and the topology of the scatternet. A significant limitation of this protocol is that it constructs a fully connected scatternet; therefore the maximum number of devices it can handle is limited to 36.

Tan et al. [40] propose a Tree Scatternet Formation (TSF) algorithm that is decentralized and builds scatternets by connecting nodes into a tree structure. Master/slave roles are dynamically assigned to each node. However, this increases the complexity of a scatternet formation algorithm and the tree structure is prone to disconnections. A healing algorithm is also proposed to minimize disruptions due to disconnections.

Law et al [28] present a scatternet formation algorithm inspired by their research on resource discovery algorithms. This algorithm consists of a *single* phase wherein the devices are partitioned into components. Each component is a set of interconnected devices (a single device, a piconet or a scatternet) and there is one leader per component. The leader executes “seek” and “scan” procedures to connect with other Bluetooth devices. Multiple piconets that can form a single piconet are merged.

However, this requires constant movement of nodes from one piconet to another as the protocol forms larger scatternets; this causes disruption in communications.

Basagni et al. [9] compare the scatternet formation protocols BlueTrees [43] and BlueStars [10]. The BlueTrees protocol designated a node called the *blueroot* that initiates the protocol and builds a tree-like scatternet rooted at itself. Then, a re-configuration procedure bounds the number of slaves per master such that, at the end of this procedure, each master has no more than seven slaves. This tree-like topology limits the robustness of the scatternet generated by this protocol. In addition, BlueTrees depends on the blueroot to initiate the protocol; this creates a single point of failure, and if the network is not connected after the device discovery phase then this solution will not work. The BlueStars protocol proceeds from the device discovery phase into two phases. In the first phase, piconet formation is initiated by all nodes in a distributed manner wherein each node decides to be a master or a slave based on a locally computed weight. The second phase generates the scatternet by interconnecting the piconets through bridges selected by each master.

Zhen et al. [45] present a two-phase scatternet formation algorithm: in the first phase, all the nodes are self-organized into “blue-star islands” that consist of piconets interconnected by a joint slave node. In the second phase, the “blue-star islands” are bridged by means of a routing trigger to form a fully connected network. The routing

trigger is implemented by means of route request messages, that have been used in on-demand routing protocols.

Stojmenovic [38] proposed a dominating-set-based three-phase scatternet formation protocol for Bluetooth networks. In the first phase, all neighbors within transmission range are discovered, the unit graph is constructed and a localized sparse subgraph is constructed. In the second phase, the Yao subgraph construct is applied simultaneously on all nodes with excessive degree, to limit the degree of each node to 7. In the third phase, master-slave roles are assigned based on dominating set membership.

The Bluetooth link formation mechanism requires that each node is pre-configured to serve as a master or a slave. In the first phase of our algorithm, the roles of all the nodes in the network are determined by means of a genetic algorithm. Therefore, unlike the protocols in [7], [28] and [36], this phase does not require the election of a leader or coordinator and is less time consuming. There is no disruption in the network during scatternet formation, as in [28] and [40].

In the second phase of our algorithm, connections are established amongst the devices based on the roles assigned to them in the previous phase. This does not require the flooding of routing trigger messages through the network, as proposed in [45]. Instead, connections amongst Bluetooth devices are established via standard Inquiry

and Page procedures (defined in the Bluetooth 1.0b specifications) until the scatternet is connected, i.e. every node in the scatternet is reachable from every other node.

4.2 Genetic Algorithm for Scatternet formation in Bluetooth networks

The proposed scatternet formation algorithm applies the genetic algorithm described in Chapter 3 to find the best combination of masters, slaves and bridges in a Bluetooth network. The algorithm executes in two stages, as outlined below:

Stage 1 – Role Determination

In the beginning of this phase, we assume all Bluetooth devices are isolated. A genetic algorithm selects random groups of nodes: these groups constitute the initial population. Each group corresponds to a combination of masters, slaves and bridge nodes and is represented by a string of length 'n' (the number of nodes in the network). Each position in the string may be assigned the value 0, 1 or 2 indicating that the node is a slave, master or bridge node respectively.

The fitness of each group of nodes is evaluated based on the number of master nodes (or piconets), slave nodes and bridge nodes in the network. A desirable property of scatternets is to minimize the number of masters, maximize the number of slaves and ensure that the bridge nodes are not too few such that bottlenecks are created. Taking this property into consideration, a fitness value is derived for each group in the population.

The following steps are repeated until a complete, new population is generated:

- *Selection* – Half the current population is retained in the new population. From the remaining half of the current population, two *parent* groups are selected from according to their fitness values; the better the fitness of a group, the better the likelihood of that group being selected.
- *Crossover* – The parent groups are crossed over with a crossover probability to form new offspring, or *children*.
- *Mutation* – The children produced by the crossover process are mutated with a mutation probability at each position in the string that represents the given child.
- The children are placed in a new population and this population replaces the previous population for a further run of the algorithm.

The end condition of this algorithm is based on the universal lower bound for the number of piconets formed. This corresponds to the value $\lceil (n-1)/k \rceil$ where k = maximum number of slaves in a piconet = 7 as per Bluetooth 1.0b specifications. This end condition is tested to see if it has been satisfied. If the number of piconets in a given group is equal to this lower bound, or within a limit of 2 greater than the lower bound, this group is chosen to be the best solution in the current population and the algorithm terminates.

Stage 2 – Connection Establishment

Once each node is assigned to the role of master, slave or bridge node then the master establishes connections with the slaves within range through the inquiry and paging procedures, as described in Chapter 3, Section 3.5. Each master discovers the devices within its transmission range via the inquiry process. Once device discovery is complete, each master then contacts its slaves by paging them. This completes connection establishment in the network and data exchange can now take place.

This approach uses the same device discovery protocol as the asymmetric link establishment protocol provided by the Bluetooth specifications. This is because the Bluetooth asymmetric device discovery protocol yields short connection establishment delays when the roles of the nodes in the Bluetooth network are pre-assigned [36], as in our algorithm.

4.3 Performance Evaluation

In this section, we investigate the performance of our scatternet formation protocol.

4.3.1 Assumptions and Performance Measures

Our scatternet formation protocol assumes that devices are within communication range of each other (10m – 100m according to Bluetooth 1.0b specifications). The input to the algorithm is: (i) the number of nodes in the network n and (ii) the

population size. The performance of the resulting scatternet may be evaluated using the following measures:

- *Time complexity* – amount of time involved in scatternet formation. This value must be minimized to reduce the delay experienced by end-users.
- *Number of piconets* – the number of collisions increases with an increase in the number of piconets, since all piconets share the same set of 79 channels.
- *Number of rounds required in Scatternet formation* – this parameter is equivalent to the number of generations that the genetic algorithm in Stage 1 iterates through and must be minimized in order to reduce end-user delay.
- *Number of slaves* – the number of slaves in a piconet should be bounded by 7; this avoids the time and bandwidth overhead associated with slave parking and unparking operations.
- *Number of bridge nodes* – a large number of bridge nodes that switch between piconets implies reduced performance due to the overhead associated with piconet switching. This is estimated to be an average of 2 time slots [9]. However, bridge nodes must not be too few, because then they will be bottlenecks for communication.
- *Maximum number of piconets that any device belongs to* – this is also known as the maximum degree of the device. If a shared slave belongs to several piconets, this slave could become overloaded and become a bottleneck for

communication between piconets. In a wireless network with many resource constraints, this factor is of paramount importance.

4.3.2 Performance

The scatternet formation algorithm was simulated using C++ code and IBM's Bluehoc/Bluescat modules [3] for the network simulator ns-2 [1]. Experiments were conducted for varying numbers of nodes, starting with 5 nodes up to 100 nodes and for population sizes 5, 10 and 20. The number of generations (equivalent to the number of rounds), number of masters (number of piconets), execution time of the genetic algorithm and the number of bridge nodes were measured and graphs were plotted (Figures 10, 11, 12 and 13).

It was found that the number of rounds required in Scatternet formation was about the same for all population sizes (5, 10 and 20) until the number of nodes was 40. Beyond this, the number of rounds increases for small population sizes, as shown in Figure 11. The execution time (Figure 12) was much greater for smaller population sizes as the number of nodes in the network increased beyond 50. The number of piconets did not vary much between the different population sizes, as shown in Figure 10. This value was always within a limit of 2 greater than the optimal number of piconets, as defined by the end condition of the genetic algorithm.

The number of slaves per piconet was found to be bounded by $k = 7$. This is because the algorithm ensures that the number of masters is optimal, or within a limit of 2 greater than the optimal value. This calculation of the number of masters is based on the requirement that the size of each piconet must be less than or equal to 8. Since all the nodes are within radio transmission range of each other, (i) each node is paged by the master nearest to it, and connects to that master, and (ii) no node remains isolated. This avoids a situation wherein one master connects to all the nodes such that the piconet size exceeds 8 and one or more masters remain isolated. The average connection establishment time between two Bluetooth devices was found to be about 2 seconds. This increases linearly with respect to the number of nodes in the network.

The average number of bridge nodes in a Bluetooth network of varying size is plotted in Figure 13. The number of bridges increases in a linear fashion with respect to the number of nodes in the network, for all three population sizes. At any given time, the average percentage of bridge nodes in a network is about 33%. This does not cause any apparent bottlenecks for communication in the Scatternet. In addition, the algorithm ensured that even if there were multiple shared devices (bridge nodes) assigned to a piconet, the actual number of bridge nodes would be equal to the *degree* of the piconet i.e. how many piconets were adjacent to the given piconet. This implies that any two piconets shared only one bridge. The remaining nodes assigned

to the piconet, whether assigned as slaves or bridge nodes, functioned as slave nodes.

The shared devices did not contribute to any bottlenecks in communication.

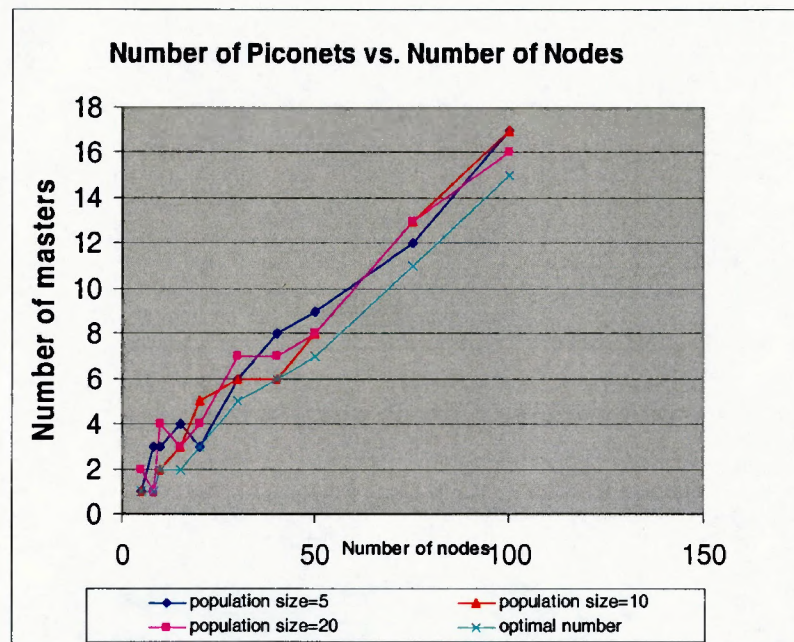


Figure 10 – Number of piconets or Number of masters

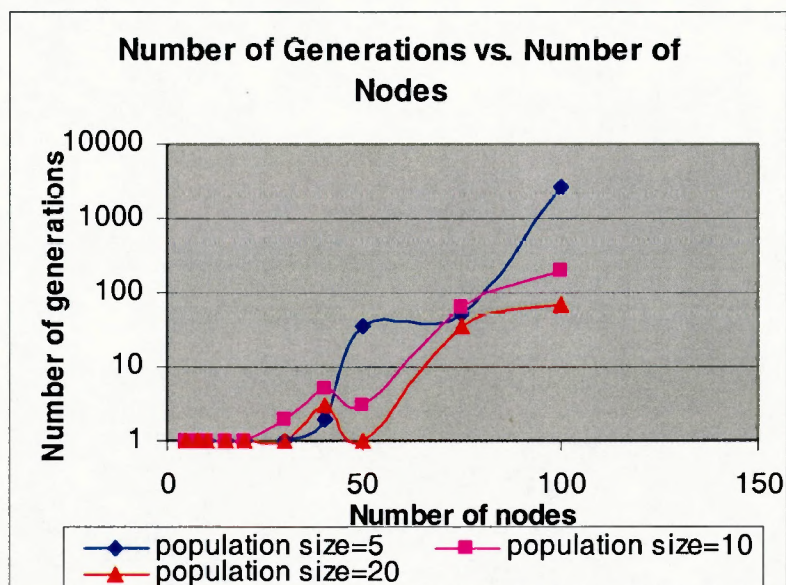


Figure 11 – Number of rounds in Scatternet formation

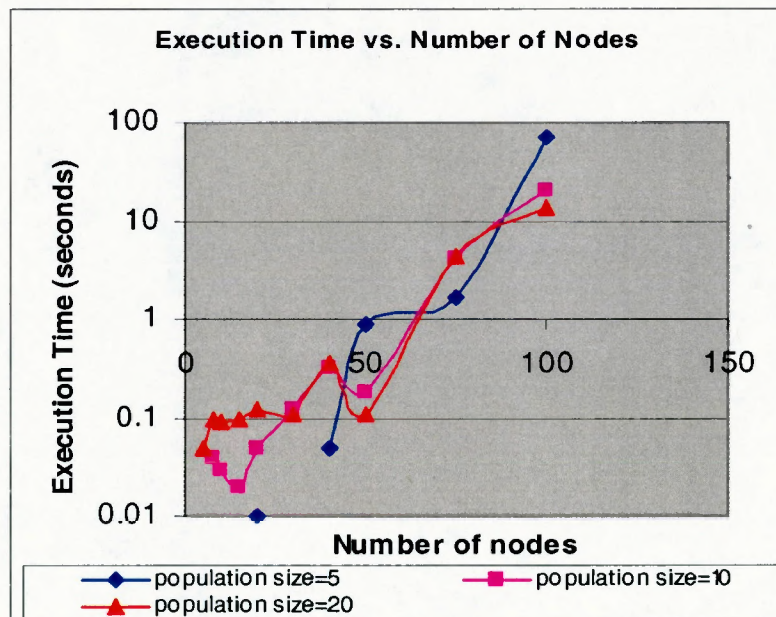


Figure 12 – Execution Time of the Genetic Algorithm

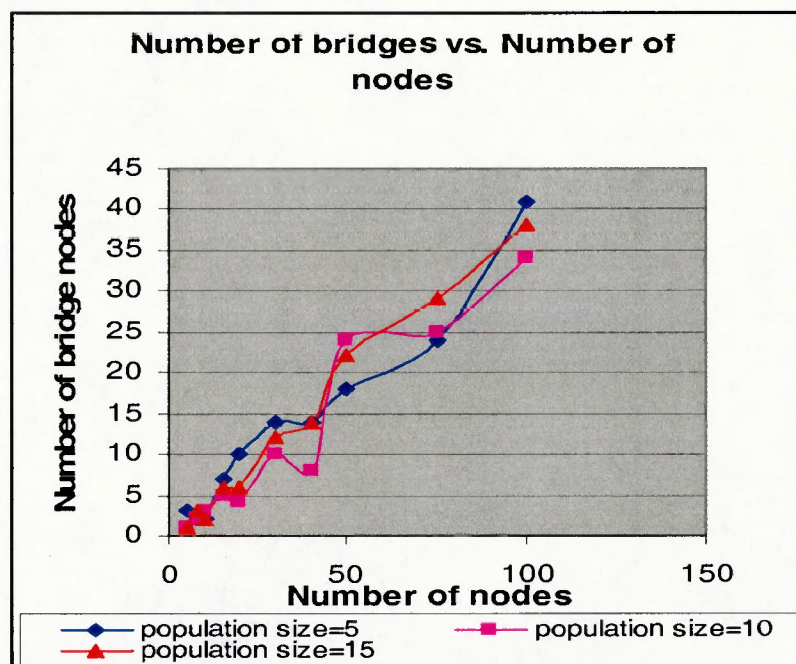


Figure 13 – Number of bridges in a Scatternet

The simulation results match the expected outcome of any genetic algorithm: if the number of nodes is increased beyond a threshold value, the performance of the algorithm degrades significantly due to an increase in the number of generations to be processed. However, when the population size is increased, this increases the variation in the population and thus allows for a greater probability of the best solution being found quickly, even in large Bluetooth networks. As a result, the performance of the genetic algorithm improves considerably. Population sizes of 20 and above may be used to ensure the efficiency of this algorithm.

The end condition defined by our genetic algorithm ensures that the best solution is returned only when the number of piconets is equal to, or very near, the optimal

number of piconets for the given number of nodes. This implies that the number of piconets is minimized for any size of the network or current population (Figure 10); this in turn reduces interference between piconets.

The study of genetic algorithms that led to the development of this evolutionary scatternet formation protocol helped pave the way to apply genetic algorithms to design an efficient routing strategy for ad hoc wireless networks. Conducting scatternet formation simulations using the Bluehoc module for the network simulator ns-2 assisted in the implementation and performance evaluation of the dominating-set-based routing protocols for ad hoc wireless networks using ns-2.

Chapter 5

An Evolutionary Dominating-set-based Routing Approach in Mobile Ad Hoc Networks

The proposed approach to routing in ad hoc wireless networks using a virtual backbone is based on genetic algorithms. This chapter presents the reasons for using such an evolutionary approach. We then describe the sequence of steps involved in a genetic algorithm and illustrate how our approach uses the same sequence of steps to determine roles for the mobile nodes in the graph underlying the network.

Because of the dynamic nature of mobile nodes, mobile ad hoc networks (MANETs) evolve over time. Each change in the network topology may be viewed as a mutation in the sequence representing the nodes in the network. As the network changes, the virtual backbone for the given network must be recomputed, this in turn triggers a change in the routing process. This continuous series of changes in the network fits naturally with an evolution process and therefore may be modeled using a genetic algorithm. This eventually results in populations of nodes with above average fitness than in previous populations and a certain desired value of fitness leads us to the best solution.

Although genetic algorithms take time to converge, we adapt the algorithm to the rapidly changing network environment by selecting a sequence of nodes at the very

beginning that is likely to produce an optimal solution quickly by virtue of being the “fittest” group of nodes. We assume that the time taken by the algorithm to find the best solution is balanced by the quality of the solution returned, since the very nature of genetic algorithms relies on solutions that have worked well in the past and are therefore guaranteed to work in the present.

A desirable attribute of an evolutionary approach to the routing problem in ad hoc wireless networks include the ability of such an approach to handle large, non-linear and dynamic solution spaces. In addition, this approach is robust and flexible; different objectives may be explored by using different fitness functions.

5.1 Genetic Algorithm for Dominating-set-based routing in Mobile Ad Hoc Networks

Our objective is to design a heuristic to find a *stable* approximation to a minimum connected dominating set (MCDS) in an ad-hoc wireless network. This algorithm assigns the attribute *stability* to a mobile node: an estimate based on the rate of movement of the node indicative of how long the node will remain static. We utilize this attribute to generate the “fittest” dominating set by selecting a group of nodes with the maximum value of stability as one of the parent groups. This stability will then be inherited by the offspring via the selection process.

In our model, a binary string represents a possible solution, or combination of backbone and non-backbone nodes in the network. Each bit in the string represents a mobile node: a value of 1 represents a backbone, or dominator, node and a value of 0 represents a non-backbone, or dominee, node. The algorithm repeatedly applies the genetic operations of selection, crossover and mutation to the members of a population of strings, eventually producing strings that represent good solutions.

Our routing algorithm proceeds in the following sequence of steps:

1. Construct the dominating set in the graph that models the given ad hoc wireless network.
2. Collect topology information transmitted from non-dominator nodes (dominees) to dominator nodes.
3. Broadcast global topology to all dominators.
4. Each dominator runs an all-pairs shortest paths algorithm on its local copy of the topology.
5. Propagate routing tables out to the dominees.
6. The dominating set is recomputed periodically based on previous solutions such that the solution evolves until the optimal solution is found.
7. The dominators provide a periodic maintenance update by periodically broadcasting topology updates to all dominees in their respective domains.

The construction of a stable dominating set in the network (Step 1 of the routing algorithm) follows the sequence of steps in a genetic algorithm and proceeds as outlined below:

1. Random groups of nodes are selected from the n nodes in the network; each group corresponds to a possible dominating set. Each group may be represented as a binary string of length n ; a value of 1 in position 'i' indicates that the node i belongs to the dominating set and a value of 0 in position 'i' indicated that the node i does *not* belong to the dominating set. Let one group be such that it contains the nodes with maximum stability.
2. The *fitness* of each group of n nodes in the population is evaluated; let this fitness be a function of the stability of the dominating set.
3. A new population is generated by repeating the following steps until the new population is complete:
 - *Selection* – select two groups of nodes from a population according to their fitness (the better the fitness of a group, the better the likelihood of that group being selected).
 - *Crossover* – Cross over the selected groups of nodes to form new groups.
 - *Mutation* – With a certain probability, change or mutate the new groups of nodes at each position in the binary string that represents each group.
 - Place new groups of nodes in a new population.

4. The previous population is replaced with the new population for a further run of the algorithm.
5. The end condition (a certain value for stability, connectivity and size of dominating set) is tested to see if it has been satisfied. If so, then the algorithm stops and the best solution in the current population is returned
6. Return to step 2 and repeat.

Example 1: Let us consider ten nodes that constitute an ad hoc wireless network. We can set the population size to be eight i.e. the population processed by the genetic algorithm consists of eight members, or solution groups. In step 1 of our genetic algorithm, the set of node groups generated may be: 10010011, 00010100, 11001001, 01010001, 10001000, 11010000, 00010010, and 01110000. This notation implies that in string 1, nodes 1, 4, 7 and 8 are dominators or backbone nodes, and nodes 2, 3, 5 and 6 are dominatees, or non-backbone nodes. One of these groups is generated by retrieving the stabilities, based on current speeds, of all the nodes, and choosing the most stable nodes to be members of the dominating set.

The fitness of each string is then evaluated by summing the stabilities of the dominator nodes in the string. Two parent groups of nodes are selected on the basis of their fitness; two groups that have the highest fitness values are chosen. Let us assume that the two groups selected are: 10001000 and 11010000. The crossover operation produces the groups: 10000000 and 11011000. The mutation operation is now applied with a certain probability. Let us assume that this operation changes the

groups to produce: 10010000 and 10011100. These two groups of nodes are then assigned to a new population. The selection, crossover and mutation operations are applied iteratively to the old population until the new population is complete.

The new population is then subject to further runs of the algorithm until a certain maximum value of stability for a group of nodes that denotes the best solution is obtained. If the best solution is found, the algorithm returns the best solution and terminates. The best solution yields the combination of backbone and non-backbone nodes and the backbone for the network is assigned accordingly. If the best solution is not found, then the algorithm proceeds in an iterative manner until the best solution is found, or a maximum number of generations (measured by the number of iterations of the algorithm) is achieved.

Example 2: The steps involved in the routing process using our genetic algorithm may be illustrated by the example network in Figure 14.

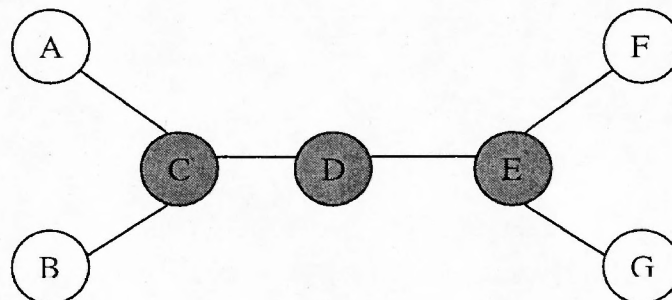


Figure 14 – Dominating-set-based routing using the proposed genetic algorithm

We can set the population size to be eight i.e. the population processed by the genetic algorithm consists of eight members, or solution groups. In step 1 of our genetic algorithm, the set of node groups generated may be: 1001000, 1000101, 0001010, 0011000, 0101110, 1100100, 0100001, and 0011100. This notation implies that in group 1, nodes A and D are dominators or backbone nodes, and nodes B, C, E, F and G are dominatees, or non-backbone nodes. One of these groups is generated by retrieving the stabilities, based on current speeds, of all the nodes, and choosing the most stable nodes to be members of the dominating set. Let us assume that the nodes C and D have maximum stability amongst all the nodes. Therefore, the group 0011000 represents the group with maximum stability.

The fitness of each string is then evaluated by summing the stabilities of the dominator nodes in the string. Two parent groups of nodes are selected on the basis of their fitness; two groups that have high fitness values have a greater probability of being chosen. Let us assume that the two groups selected are: 0011000 and 0011100. The crossover operation produces the groups: 0011100 and 0011000. The mutation operation is now applied with a certain probability. Let us assume that this operation changes the groups to produce: 0111100 and 0001100. These two groups of nodes are then assigned to a new population. At this stage:

Old population - 1001000, 1000101, 0001010, 0011000, 0101110, 1100100, 0100001, and 0011100

New population – 0111100, 0001100

The selection, crossover and mutation operations are applied iteratively to the old population until the new population is complete. This time, let us assume that the two parents selected from the old population are: 0001010 and 0011100. The crossover operation produces the groups: 0001100 and 0011010. The mutation operation is now applied with a certain probability. Let us assume that this operation changes the groups to produce: 0011100 and 0111001. These two groups of nodes are then assigned to the new population. At this stage:

Old population - 1001000, 1000101, 0001010, 0011000, 0101110, 1100100, 0100001, and 0011100

New population – 0111100, 0001100, 0011100, 0111001

In the next step, let us assume that the two parents selected from the old population are: 0101110 and 0011000. The crossover operation produces the groups: 0101000 and 0011110. The mutation operation is now applied with a certain probability. Let us assume that this operation changes the groups to produce: 0111000 and 0011101. These two groups of nodes are then assigned to the new population. At this stage:

Old population - 1001000, 1000101, 0001010, 0011000, 0101110, 1100100, 0100001, and 0011100

New population – 0111100, 0001100, 0011100, 0111001, 0111000, 0011101

In the final iteration of building a new population, let us assume that the two parents selected from the old population are: 1000101 and 0011100. The crossover operation produces the groups: 1000100 and 0011101. The mutation operation is now applied with a certain probability. Let us assume that this operation changes the groups to produce: 0100100 and 0111100. These two groups of nodes are then assigned to the new population. At this stage:

Old population - 1001000, 1000101, 0001010, 0011000, 0101110, 1100100, 0100001, and 0011100

New population – 0111100, 0001100, 0011100, 0111001, 0111000, 0011101, 0100100 and 0111100

The new population is now tested with the end condition that involves the connectivity and size of the dominating set. The group 0011100 meets the criteria for connectivity since nodes C, D and E are connected, as shown in Figure 14. It is also the minimum connected dominating set for the network. Therefore, the end condition is met and the algorithm terminates producing the best solution.

After Step 1 of the routing algorithm, the MCDS consists of nodes C, D and E. Node C dominates nodes A and B. Node E dominates nodes F and G. After Steps 2 and 3, each node in the MCDS constructs its local copy of the topology. Using these local copies, Nodes C, D and E compute shortest paths in Step 4. In Step 5, C and E transmit routing tables to the non-MCDS nodes.

Upon node movement, the backbone must be reconstructed. The genetic algorithm is now rerun with one of the initial groups assigned to the previous solution, i.e. 0011100. This should yield a new solution quickly, since the new solution should be simply a mutation of the previous solution in accordance with a changed, or mutated, network topology. Therefore, a previous best, or “fittest” solution should pass on its fitness to its offspring and thus a new best solution is found, as directed by the process of evolution.

Chapter 6

Performance Evaluation

In this chapter, we evaluate and compare the performance of the protocols described in Chapters 3 and 5 through our implementation of the dominating-set-based routing protocols using the ns-2 network simulator [1] and simulations conducted using this implementation. We chose to implement Distributed Approximation Algorithm I in Das et al's algorithm and Distributed Approximation Algorithm II in Cheng et al's algorithm, since it outperforms Algorithm I [15].

Our evaluations are based on the simulation of 10, 20, 30 and 50 nodes forming an ad hoc network, moving within a 670m X 670m grid for 500 seconds of simulated time. CBR traffic with a packet size of 512 bytes was used as data traffic in all simulations. We chose this packet size because a larger packet size causes congestion in the network and results in a substantial increase in the number of packets being dropped. The packet size was changed to 64 bytes and detailed simulations were conducted to observe the improved performance of the routing protocols. We used 10 and 20 traffic sources with a sending rate of 4 packets/second across all scenarios. Varying the number of sources has been found to be approximately equivalent to varying the sending rate [14]. Therefore, we fixed the sending rate at 4 packets per second and varied the network load by using 10 and 20 traffic sources. We chose CBR traffic instead of TCP traffic because TCP changes the times at which packets are sent

according to the ability of the network to carry packets [14]; therefore the position of a node while sending a packet and the time at which a packet is originated will be different for different routing protocols resulting in an unfair comparison. During these experiments, the pause times were varied between 0 seconds and 500 seconds. The speed of the mobile nodes was set to a maximum of 20 m/s.

The performance of the dominating-set-based routing protocols may be evaluated using the following measures:

1. *End-to-end packet delay*: this is a measure of the average delay experienced by a packet from the time it was originated at the source until the time it was received at the destination.
2. *Throughput*: this is a measure of the total number of bytes received during the simulation; this is an indirect measure of the ability of the network to deliver packets from the source to the destination effectively.
3. *Packet delivery ratio*: this parameter represents the total number of packets delivered or received over the total number of packets sent.
4. *Routing overhead*: this defines the number of routing packets transmitted during the simulation as compared to the total number of packets transmitted.

These measures reflect on the benefits of the virtual backbone infrastructure. This backbone comprising the mobile nodes that form the dominating set allows for the

reduction in the number of nodes performing route computations and avoids flooding in the network by using unicast mechanisms for packet flow. This in turn results in improved performance due to a decrease in route response time and consequently, queuing time. Therefore, if the dominating set construction algorithm yields a near-optimal number of dominator nodes in a given network, this implies that the algorithm is more efficient, resulting in improved throughput and packet delivery ratio and lower routing overhead. End-to-end delays depend on the number of dominator nodes in the virtual backbone; this is because the dominator nodes participate in route computation for the entire network and thus have larger numbers of packets queued. This results in larger end-to-end packet delays. Again, the closer the number of dominators is to the optimal number of dominators in a given network, the lower the value of end-to-end delay and routing overhead. In summary, the metrics being used to evaluate and compare the performance of the dominating-set-based routing protocols are directly related to the efficiency of the virtual backbone in routing packets through the network.

6.1 End-to-end packet delay

This performance index is comparable across all the dominating-set-based routing protocols while using 10 traffic sources, as shown in Figures 15 and 16. It increases with an increase in the number of nodes since the packets have to travel across an increased number of hops. Alzoubi's protocol results in a substantially larger delay when the network consists of 50 nodes. Also, Das et al's protocol exhibits a large

delay for large networks when the node mobility is decreased, as shown in Figure 16. When the number of traffic sources is increased to 20, the end-to-end delay increases for Das et al's protocol as well as Wu and Li's protocol across most scenarios. Alzoubi's protocol as well as the genetic protocol exhibits a larger delay than the others for large networks when 20 traffic sources are used and the node mobility is decreased, as shown in Figure 18. These large delays could be because the protocol yielded a large number of dominators for that particular scenario or because of increased queuing delay due to the increased traffic load on the network.

When 10 traffic sources are used in the network and node mobility decreases, Alzoubi's protocol performs better than the majority of the other dominating-set-based routing protocols and exhibits similar trends as the genetic protocol across all scenarios, as shown in Figure 16. Overall, the protocol based on the genetic algorithm appears to perform best with respect to end-to-end delay, as shown in Figures 15, 16, 17 and 18.

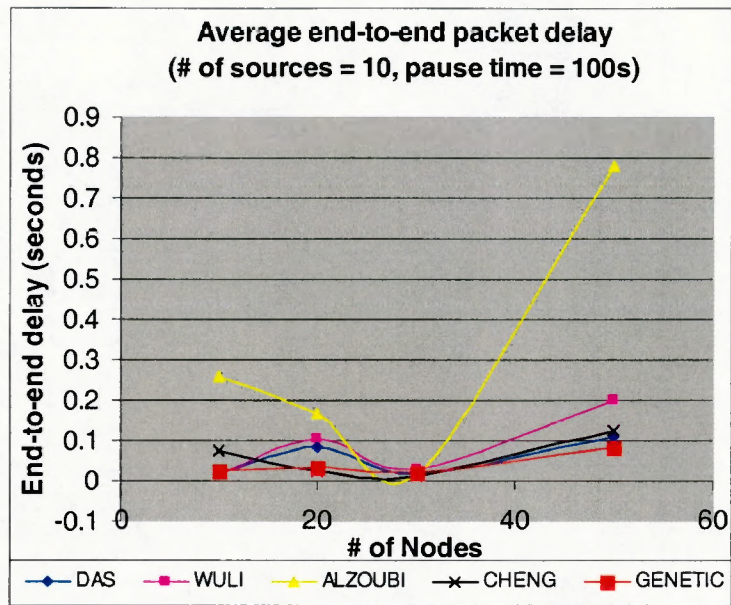


Figure 15 - Comparison of average end-to-end packet delay with 10 traffic sources and pause time = 100 seconds

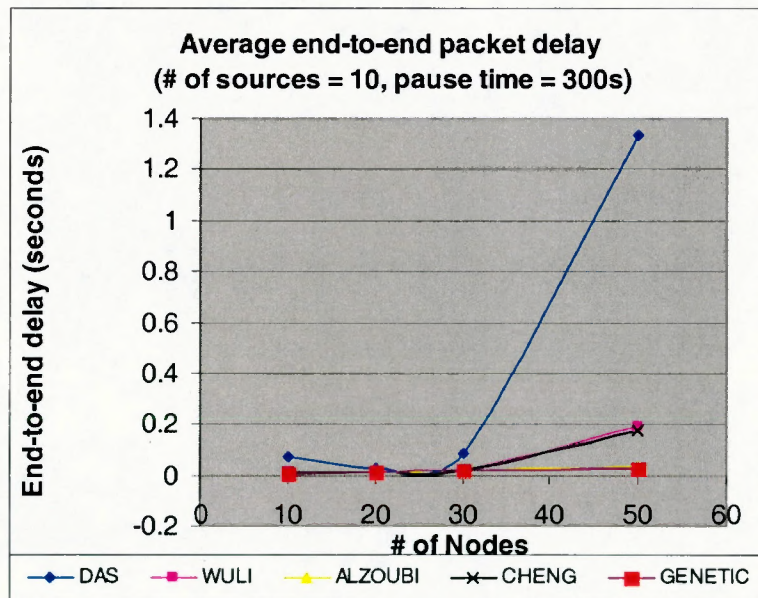


Figure 16 - Comparison of average end-to-end packet delay with 10 traffic sources and pause time = 300 seconds

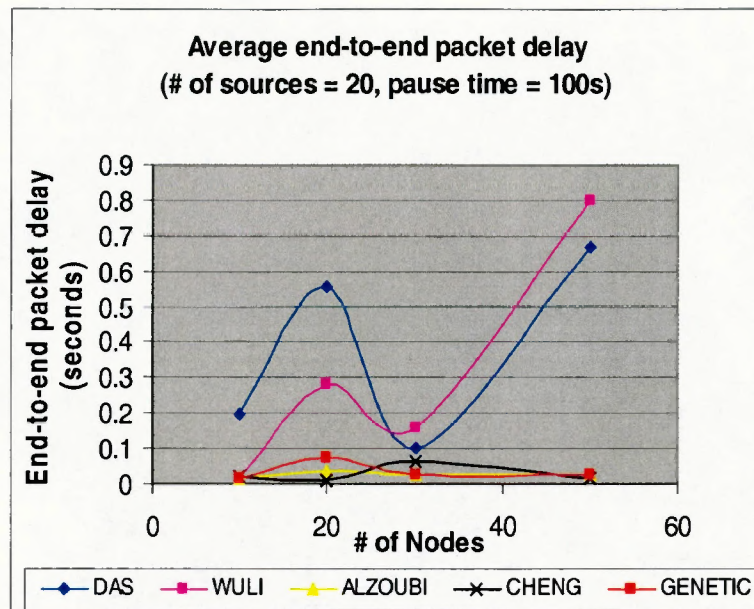


Figure 17 - Comparison of average end-to-end packet delay with 20 traffic sources and pause time = 100 seconds

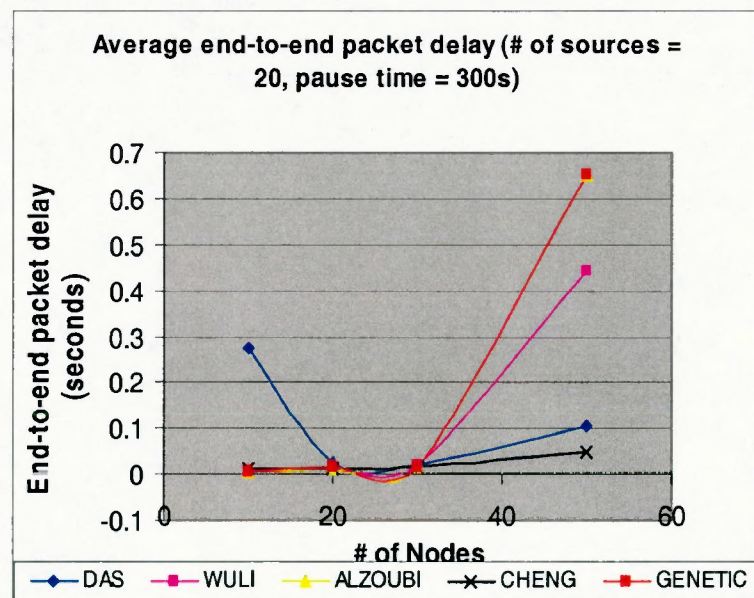


Figure 18 - Comparison of average end-to-end packet delay with 20 traffic sources and pause time = 300 seconds

6.2 Throughput

This metric is also comparable across all protocols, as shown in Figures 19, 20, 21 and 22. As the network size and traffic load is increased, the throughput increases. This can be attributed to an increase in the node density while maintaining grid dimensions; this implies that more nodes are within transmission range of each other and there are less chances of the network being partitioned. It also means that there are more nodes available to forward packets between nodes, and more packets flowing through the network when the traffic load is increased, thus increasing throughput. If the throughput is low for networks comprising 30 or 50 nodes, this is due to the increased likelihood of broken links and stale routes and the increased delay in propagating topology information through larger networks, resulting in an increase in packet loss.

Das et al's protocol appears to perform best with respect to the number of bytes delivered in networks comprising less than 50 nodes when 10 traffic sources are used and the pause time is 100 seconds, as can be seen in Figure 19. When node mobility is decreased, the genetic protocol performs best, as can be seen in Figure 20.

When the number of traffic sources is increased to 20, Wu and Li's protocol performs best across all scenarios, followed by Alzoubi's protocol and the genetic protocol for networks consisting of 30 nodes and 50 nodes. For networks of 50 nodes and when

node mobility is decreased, the genetic protocol performs best across most scenarios, as can be seen in Figures 19, 20, 21 and 22. The substantial difference in the performance of our protocol compared to the other protocols in large networks may be attributed towards the ability of the genetic algorithm to construct efficient backbones even for large networks.

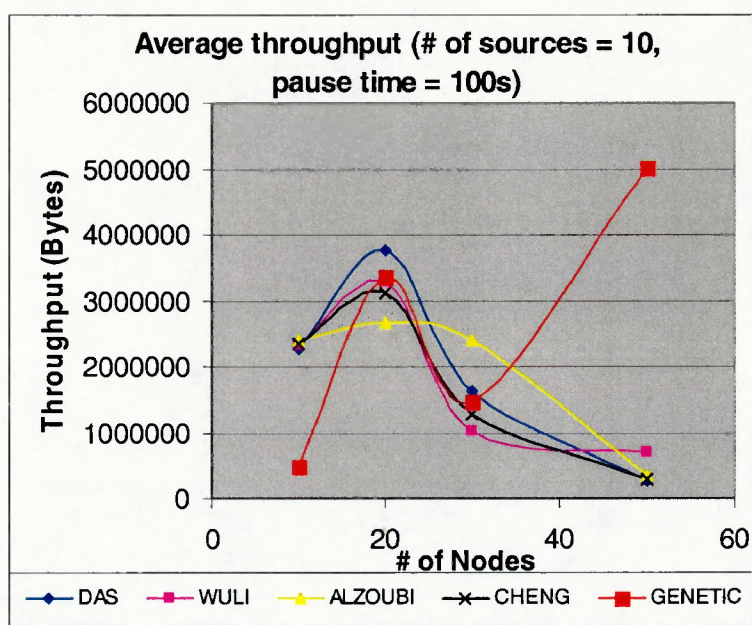


Figure 19 - Comparison of average throughput with 10 traffic sources and pause time = 100 seconds

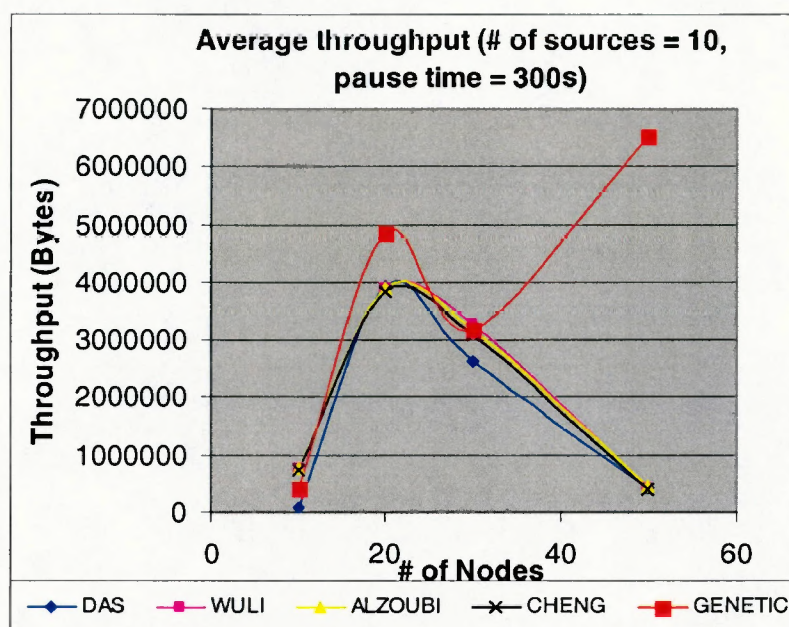


Figure 20 - Comparison of average throughput with 10 traffic sources and pause time = 300 seconds

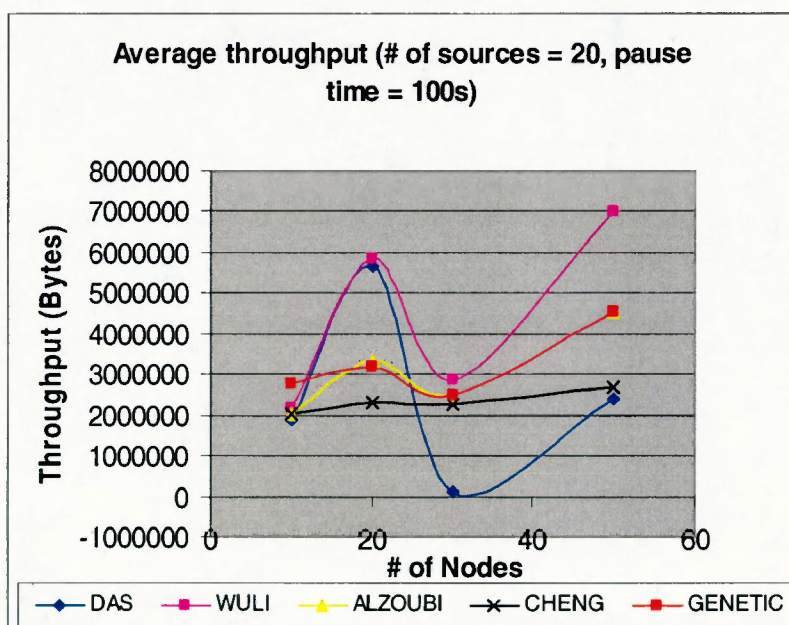


Figure 21 - Comparison of average throughput with 20 traffic sources and pause time = 100 seconds

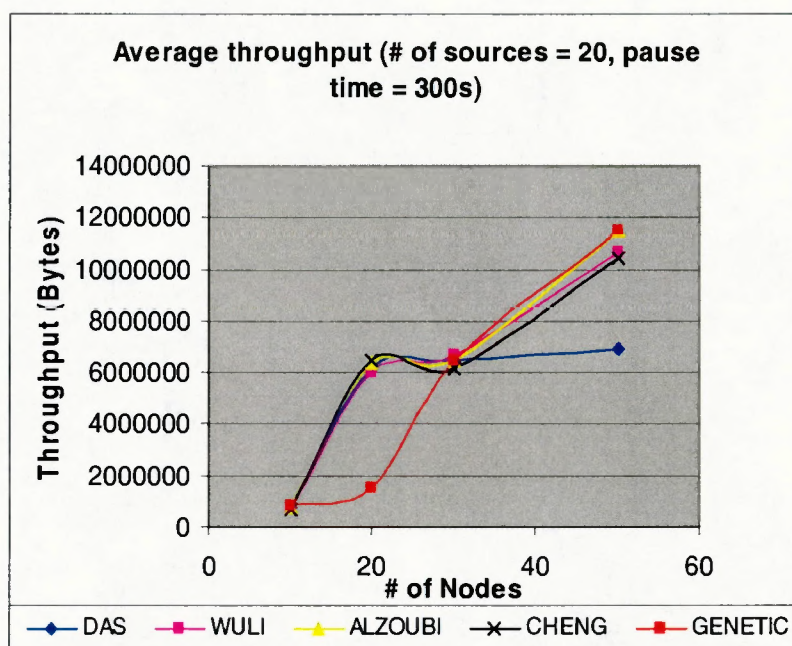


Figure 22 - Comparison of average throughput with 20 traffic sources and pause time = 300 seconds

6.3 Packet delivery ratio

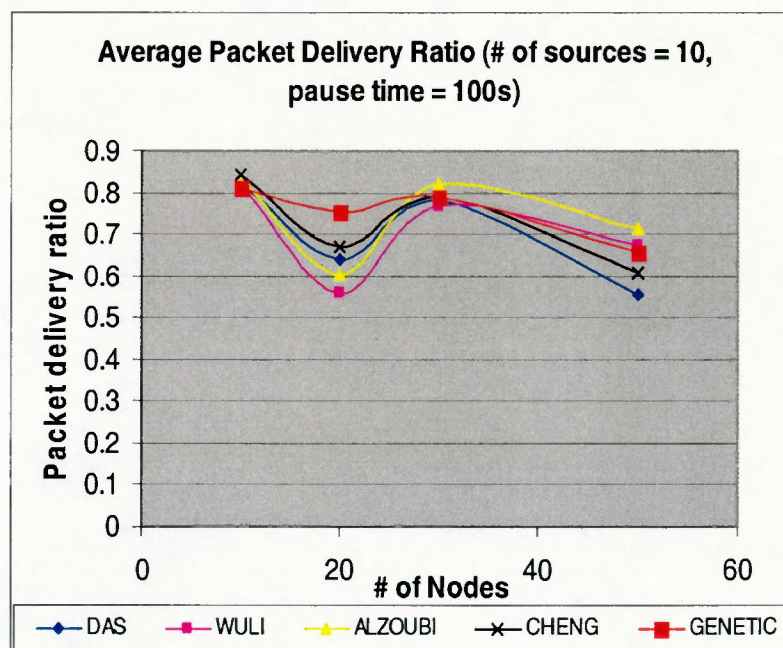
This parameter exhibits a similar trend across all the protocols, as shown in Figures 23, 24, 25 and 26. A low percentage of packets delivered may be caused by the mobility pattern of the nodes in the given scenario, resulting in many packets being sent that are dropped. Therefore, when the node mobility decreases, the packet delivery ratio improves considerably, as shown in Figures 24 and 26. Packet loss can also be attributed to the queue buildup in dominator nodes that perform route computations. Since the queue size of the nodes is limited, this leads to queue overflow and packets are dropped. In large networks, the increased number of

dominator nodes may contribute towards increased packet loss i.e. decreased packet delivery ratio.

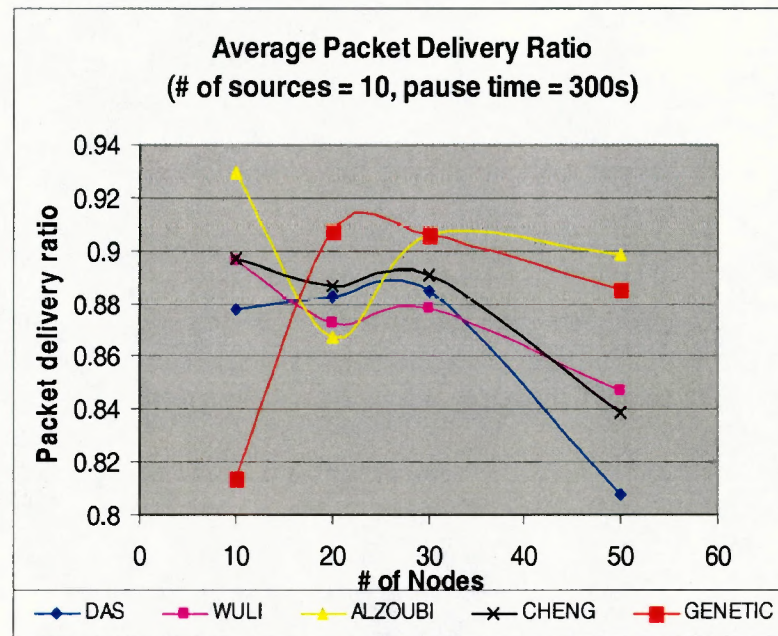
The values obtained for packet delivery ratios appear to be lower than that of other previously obtained results for routing protocols used in ad hoc networks [14]. This may be attributed to the recomputation of the backbone during which different nodes may become dominators and begin route computations. Cached routes that are stale may be used while the topology information is refreshed, resulting in an increased packet loss.

The genetic protocol appears to perform best with respect to the percentage of packets delivered in a majority of the scenarios, as shown in Figures 23, 24, 25 and 26. This could be because the construction of the backbone is efficient and topology refreshing is quicker than in the other protocols. When the node mobility decreases, the genetic protocol has a lower packet delivery ratio than the other protocols for networks comprising 10 nodes, as shown in Figure 24. This could be because decreased node density may result in partitioning of the network, and decreased node mobility implies that there are fewer chances that the network will be reconnected by node repositioning. However, the genetic protocol performs well across all other scenarios when node mobility is decreased, as can be seen in Figures 24 and 26.

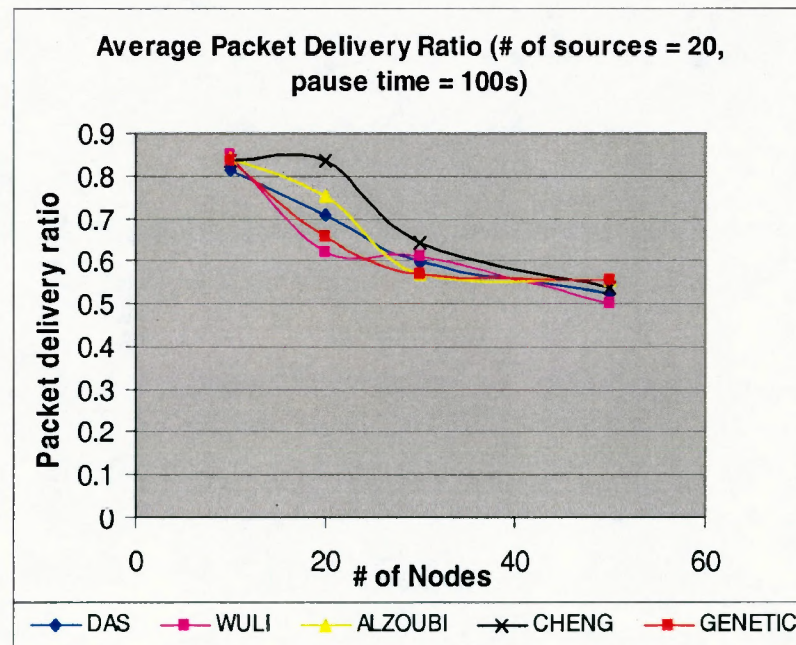
Overall, Alzoubi's protocol and the genetic protocol perform best when 10 traffic sources are used. When the number of traffic sources is increased to 20 and the pause time is 100 seconds, Cheng's protocol performs best across all scenarios and the genetic protocol outperforms all the other protocols for large networks. When the node mobility is decreased and 20 traffic sources are used, the performance of all the dominating-set-based routing protocols is comparable; Das et al's protocol performs best for large networks, as shown in Figure 26.



**Figure 23 – Comparison of average packet delivery ratios with 10 traffic sources
and pause time = 100 seconds**



**Figure 24 – Comparison of average packet delivery ratios with 10 traffic sources
and pause time = 300 seconds**



**Figure 25 – Comparison of average packet delivery ratios with 20 traffic sources
and pause time = 100 seconds**

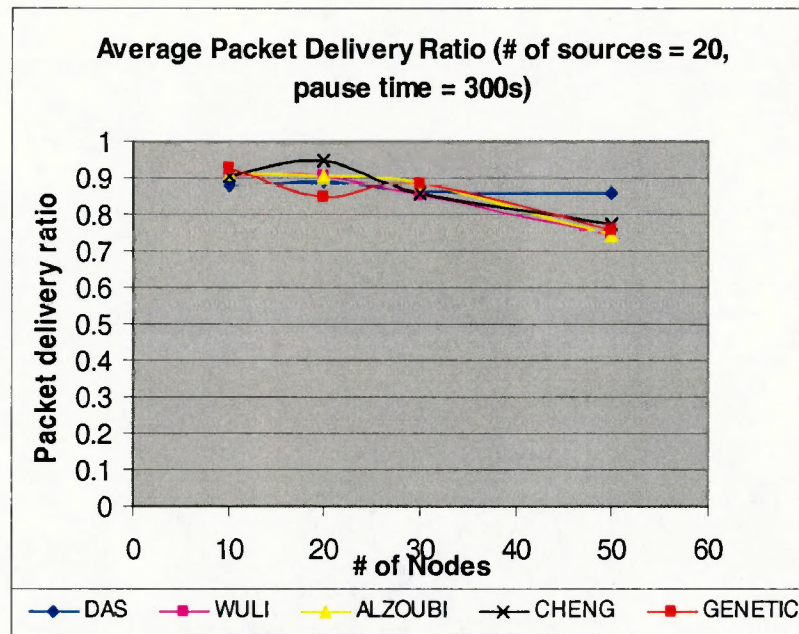


Figure 26 – Comparison of average packet delivery ratios with 20 traffic sources and pause time = 300 seconds

6.4 Routing Overhead

This performance index is important in the comparison of routing protocols, since it measures the scalability of a protocol, the ability of the protocol to perform well in congested or low-bandwidth environments and the overall efficiency of the protocol. In addition, a large number of routing packets imply a higher probability of packet collisions and increased end-to-end packet delays.

Routing overhead exhibits a similar trend across all the protocols, as shown in Figures 27, 28, 29 and 30. We have established earlier in this thesis that dominating-set-based routing protocols are hybrid protocols that follow a part-proactive and part-reactive

routing process (see Chapter 2). Therefore, broadcasts within the backbone, or dominating set, occur with every change in network topology and periodic routing table updates are broadcast to the non-backbone nodes, resulting in fairly high overhead. Since the routing overhead is being measured as the percentage of routing packets sent compared to the total number of packets sent in the network, we observe that the routing overhead as a percentage decreases as the network size increases. This may be attributed to the substantial increase in the number of data packets, and therefore the total number of packets in the network, while the number of routing packets increase but not as much as the increase in the number of data packets.

When 10 traffic sources are used with a pause time of 100 seconds, Wu and Li's protocol performs best, as shown in Figure 27. The genetic protocol has high overhead compared to the other protocols for networks comprising 10 and 20 nodes, but performs best for larger networks. When node mobility is decreased, the protocols exhibit very similar performance, as shown in Figure 28; the genetic protocol outperforms the other protocols in 10-node networks but has the highest overhead in a 50-node network.

Again, when 20 traffic sources are used, all the protocols exhibit very similar performance, as shown in Figures 29 and 30. The genetic protocol performs best for networks comprising 30 and 50 nodes when the pause time is 100 seconds, as shown

in Figure 29. When the node mobility is decreased, Das et al's protocol performs best for large networks, as shown in Figure 30.

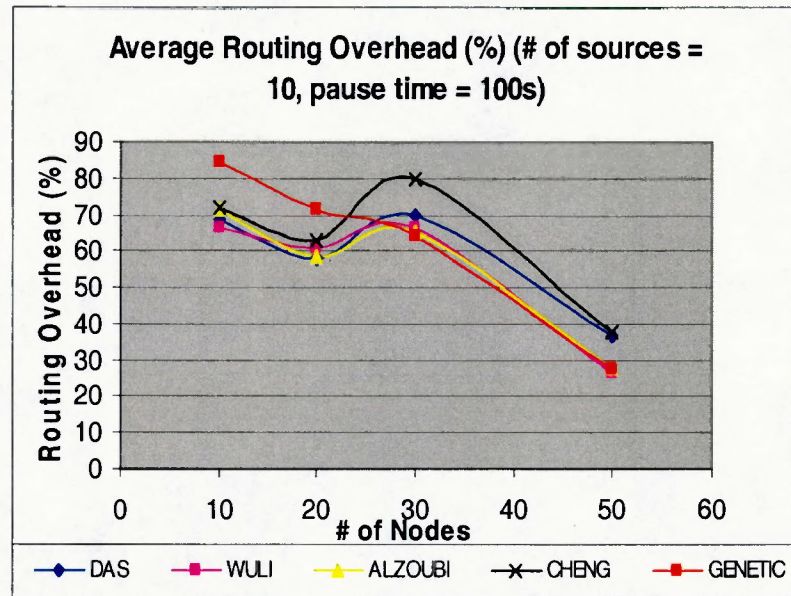


Figure 27 – Comparison of average routing overhead with 10 traffic sources and pause time = 100 seconds

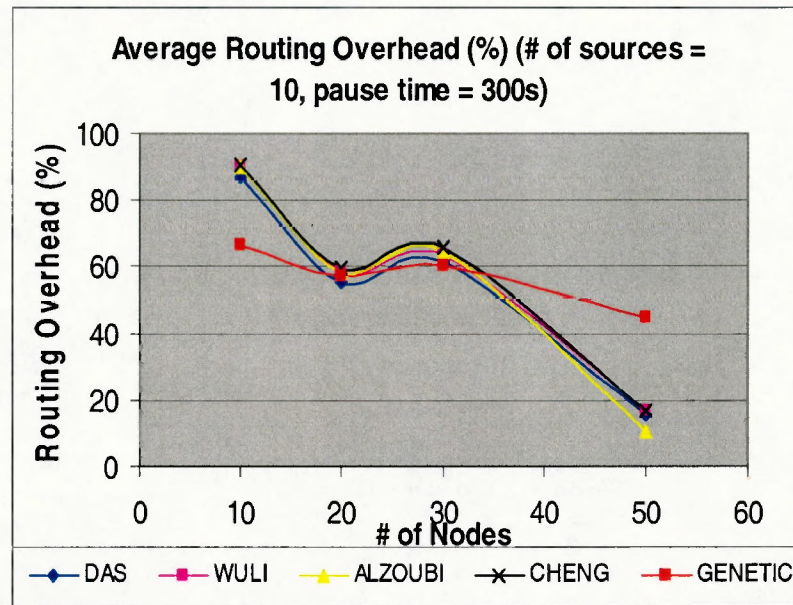


Figure 28 – Comparison of average routing overhead with 10 traffic sources and pause time = 300 seconds

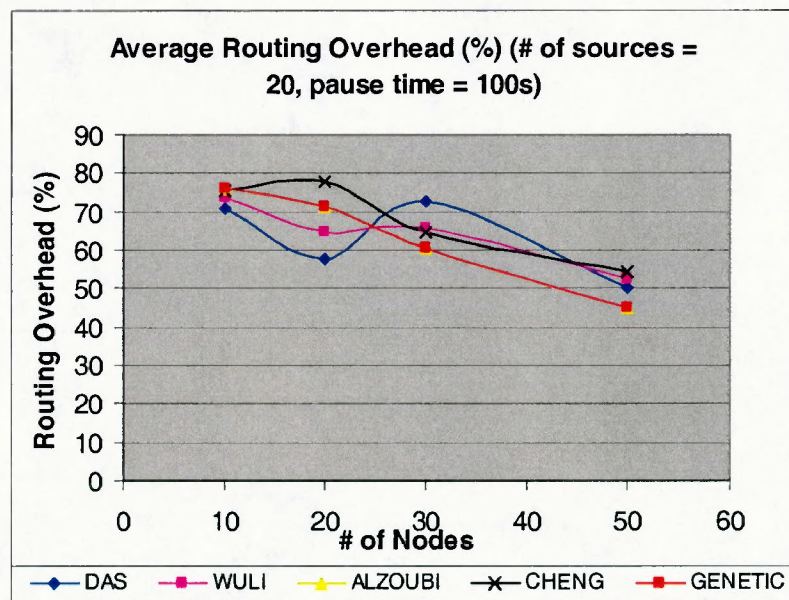


Figure 29 – Comparison of average routing overhead with 20 traffic sources and pause time = 100 seconds

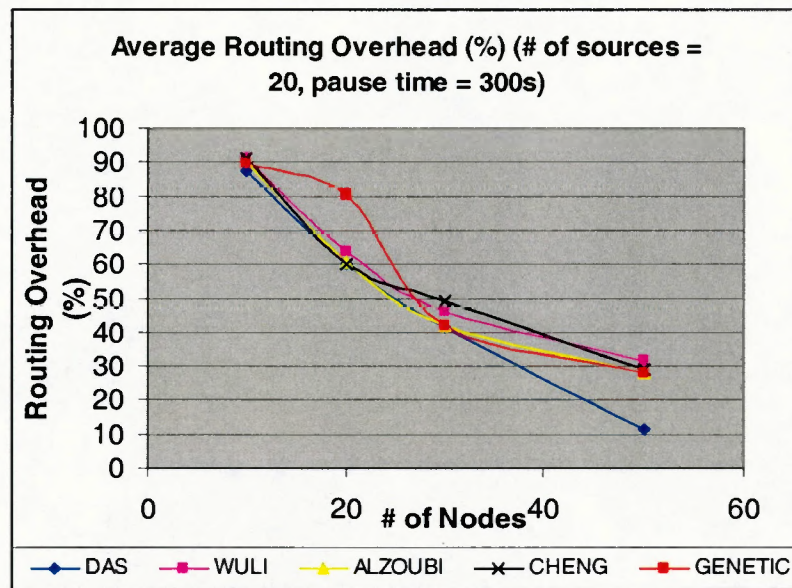


Figure 30 – Comparison of average routing overhead with 20 traffic sources and pause time = 300 seconds

6.5 Varying Packet Size

A decrease in packet size should reduce congestion as well as packet loss in the network. The packet size was varied from 512 Bytes to 64 Bytes and the performance of the dominating-set-based routing protocols was re-evaluated to observe the effect of the decrease in packet size. We chose to use 10 traffic sources with a pause time of 100 seconds across all experiments. This is because from our previous experiments, we observe that the trend exhibited by each performance index does not vary much when the traffic load is increased or node mobility is decreased. Therefore, the parameters used should be sufficient to measure and observe the improvement in performance when a smaller packet size is used.

The indices used to measure and evaluate routing performance were the same as those used in the previous simulations: end-to-end packet delay, throughput, packet delivery ratio and routing overhead. The highlights of these experiments are presented in this section.

End-to-end packet delay: The trend exhibited by this parameter is similar to the scenarios where 512 Byte packets were used. All the protocols exhibit better performance than the 512 Byte scenarios, except for Wu and Li's protocol and Das et al's protocol, as shown by comparing Figure 15 and Figure 31. Wu and Li's protocol as well as Alzoubi's protocol yield the maximum delays for large 50-node networks, as before. This delay could be because the protocol generates a large number of dominators for that particular scenario; this in turn increases queuing delay, thereby increasing end-to-end packet delay. The proposed genetic protocol outperforms the other dominating-set-based routing protocols by yielding the smallest packet delay across all scenarios, as shown in Figure 31.

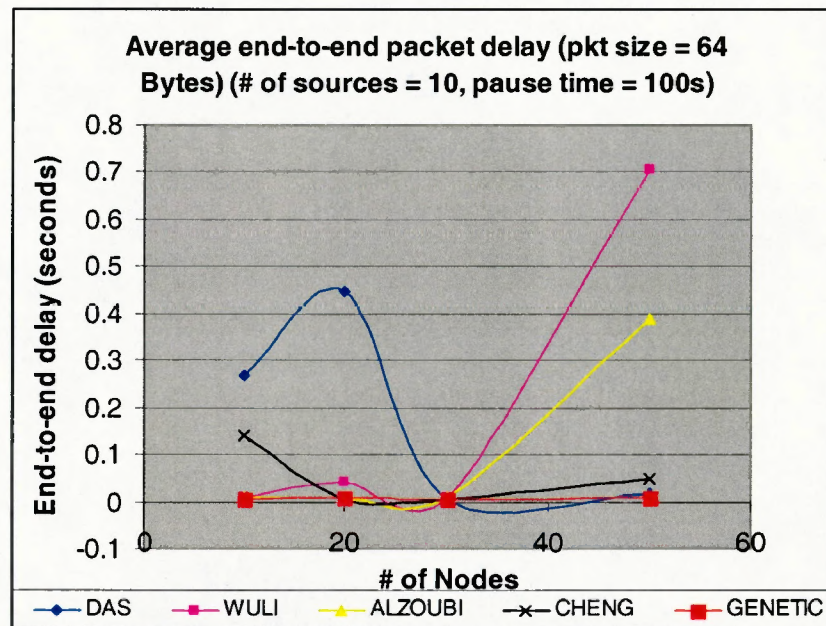


Figure 31 – Comparison of average end-to-end packet delay for 64 Byte packets with 10 traffic sources and pause time = 100 seconds

Throughput: Every dominating-set-based routing protocol exhibited improved performance compared to the 512 Byte scenarios. The higher throughput may be attributed to less congestion in the network, allowing most of the packets to be delivered. The throughput is lower for 50-node networks, as observed in the previous experiments with 512 Byte packets; this could be because of stale routes, longer delays in refreshing the topology and disconnections in large networks. The proposed genetic protocol outperforms the other dominating-set-based routing protocols for 20-node and 50-node networks, as shown in Figure 32.

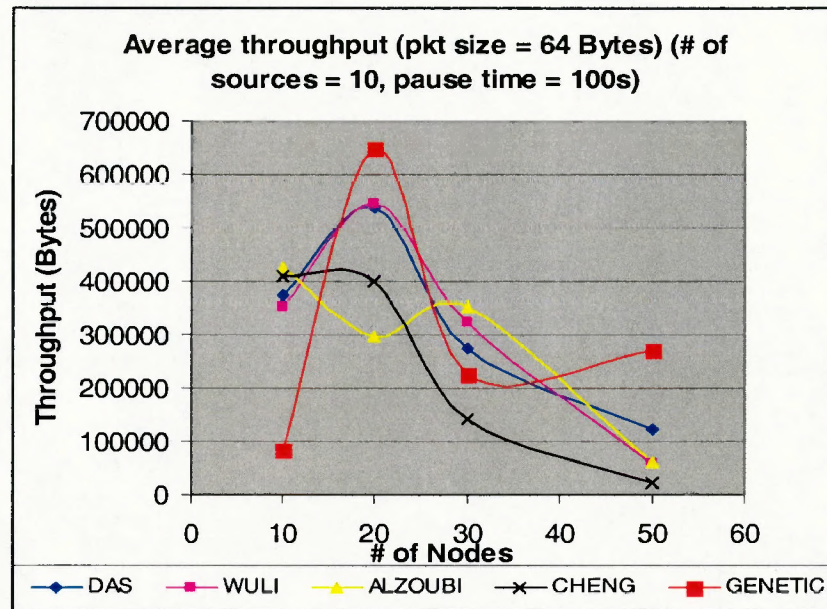


Figure 32 – Comparison of average throughput for 64 Byte packets with 10 traffic sources and pause time = 100 seconds

Packet delivery ratio: Every protocol exhibits improved performance with respect to this parameter compared to the 512 Byte scenarios. However, the proposed genetic protocol yields lower performance for 30-node and 50-node networks and is outperformed by the other protocols for these scenarios, as shown in Figure 33. This may be attributed to the mobility pattern or dominating set generated for these scenarios. As observed in our earlier simulations, the genetic protocol shows very good results for packet delivery; therefore, this case would be an exception rather than the norm. Alzoubi's protocol as well as Wu and Li's protocol performs best with respect to packet delivery ratio across a majority of the scenarios, as shown in Figure 33.

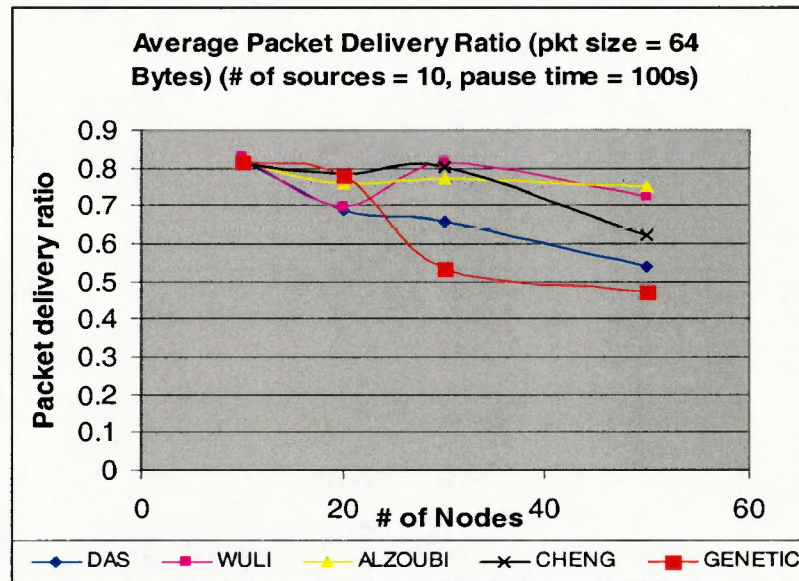


Figure 33 – Comparison of average packet delivery ratio for 64 Byte packets with 10 traffic sources and pause time = 100 seconds

Routing Overhead: This parameter exhibits similar trends as the 512 Byte scenarios across all protocols. As mentioned earlier, the routing overhead is being measured as the percentage of routing packets sent compared to the total number of packets sent in the network. From Figure 34, we observe that the routing overhead decreases as the network size increases. This may be attributed to the substantial increase in the number of data packets, and therefore the total number of packets in the network. On the other hand, the number of routing packets increases but not as much as the increase in the number of data packets.

Overall, every protocol yields a lower routing overhead than for the 512 Byte scenarios; the lowest overhead is 22% as compared to about 26% for the 512 Byte scenarios. Alzoubi's protocol as well as Wu and Li's protocol performs best with respect to routing overhead across a majority of the scenarios, as shown in Figure 34.

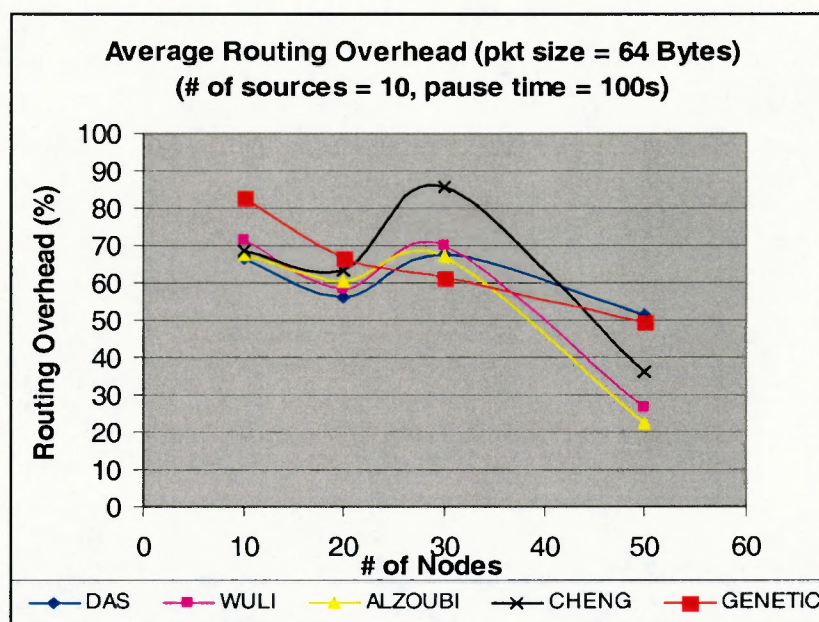


Figure 34 – Comparison of average routing overhead for 64 Byte packets with 10 traffic sources and pause time = 100 seconds

6.6 Experiments related to the Genetic Dominating-set-based Routing Protocol

This section describes the results of simulations conducted to test the proposed genetic dominating-set-based routing protocol further, with respect to:

1. Size of the dominating set or number of nodes in the virtual backbone
2. Network diameter

3. Modify the fitness function to select the nodes that have consumed the least power i.e. the nodes that have the most energy at the time of selection instead of the most stable nodes.

6.6.1 Size of the Dominating Set or Virtual Backbone

Since the size of the dominating set plays a significant role in determining the routing performance of the dominating-set-based routing protocol, we measure the average size of the dominating set or backbone generated for the proposed genetic protocol. The number of nodes in the dominating set is always less than half the number of nodes in the mobile ad hoc network, as shown in Figure 35. This implies that less than half the number of nodes in the network are involved in broadcasting packets; this greatly reduces the overhead present in protocols that use flooding (see Section 1.2) and thus improves end-to-end packet delay, throughput, packet delivery ratio and routing overhead, as we have seen in Sections 6.1-6.4.

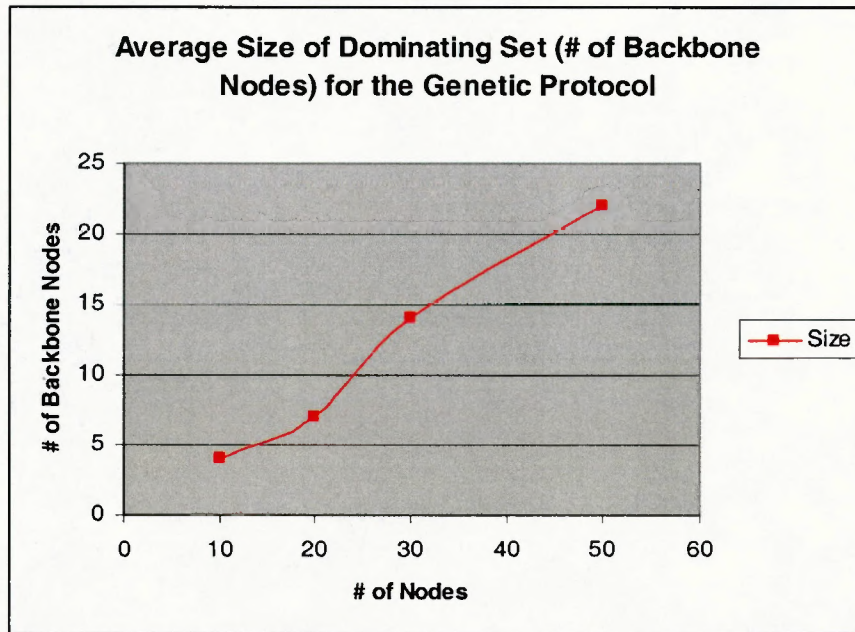


Figure 35 – Average size of Dominating Set generated by the genetic protocol

6.6.2 Network Diameter

The diameter of a network is a measure of the longest distance between any pair of nodes in the given network. This in turn measures the longest distance a packet must travel in the network, and therefore directly impacts the end-to-end packet delay. As we have seen in Sections 6.1-6.4, the proposed genetic protocol outperforms the other dominating-set-based routing protocols with respect to average end-to-end packet delay across a majority of the scenarios. For any size of network, the diameter is finite (see Figure 36) implying that the network is connected at all times. The value of the diameter is about half of the number of nodes in the network; from this, we can conclude that on the average, the greatest number of hops a packet has to travel is approximately equivalent to half the number of nodes in the network.

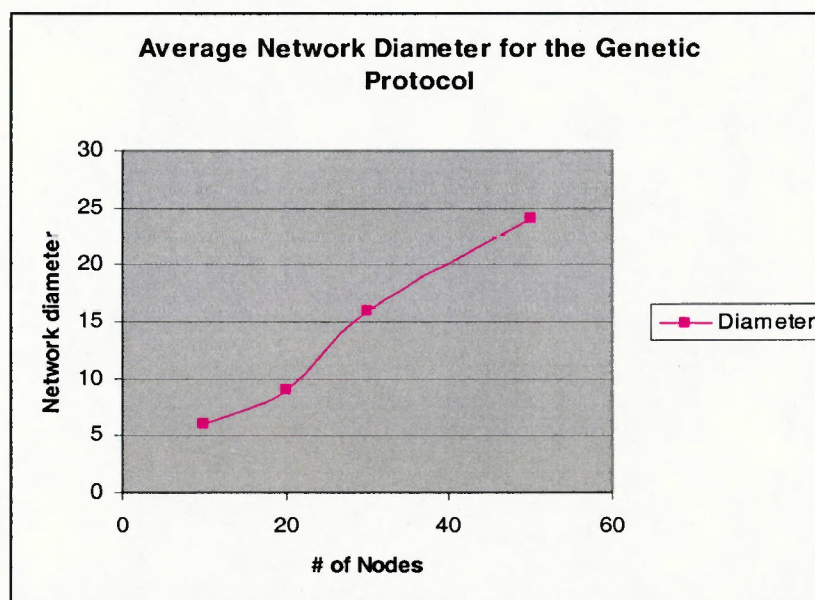


Figure 36 – Average network diameter generated by the genetic protocol

6.6.3 High-Energy Backbone Nodes

In order to experiment with the fitness function, and observe how different fitness parameters impact routing performance, we chose to select *node energy* to evaluate the fitness of the nodes. This implies that the nodes with the highest energy, or the least power consumption, at any given time are more likely to be selected as members of the dominating set or virtual backbone at that time. We compare and present the routing performance of a backbone generated based on energy as a fitness function with the backbone generated in our previous experiments with stability as a fitness function.

The simulation conditions were as described in the introduction to this chapter. Node energy was modeled using the C++ Energy Model class in ns-2. The initial energy of

each node was set to 100 Joules at the beginning of each simulation. The packet size was maintained at 512 Bytes for all simulations in this section. The indices used to measure and evaluate routing performance were the same as those used in the previous simulations: end-to-end packet delay, throughput, packet delivery ratio and routing overhead.

End-to-end packet delay: This performance index increases with an increase in the number of nodes since the packets have to travel across an increased number of hops. In certain scenarios, the average delay experienced by a packet may be less because the protocol yielded an efficient dominating set for that particular scenario. For example, in Figure 38, the delay for the 50-node network is less than that for the 30-node network. When the traffic is increased, and node mobility is decreased, the high-energy protocol yields a much lower delay than the high-stability protocol, as shown in Figure 40.

Overall, the high-energy version of the proposed genetic protocol outperforms the high-stability version with respect to average end-to-end packet delay across a majority of the scenarios. In particular, for large networks comprising 50 nodes, the high-energy genetic protocol exhibits a significantly smaller packet delay than the high-stability genetic protocol, as shown in Figures 37, 38, 39 and 40.

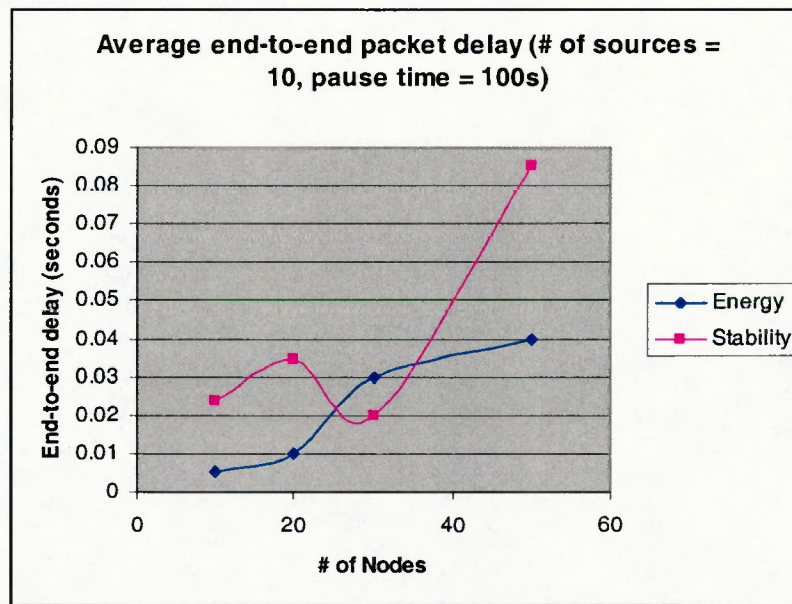


Figure 37 - Comparison of average end-to-end packet delay for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds

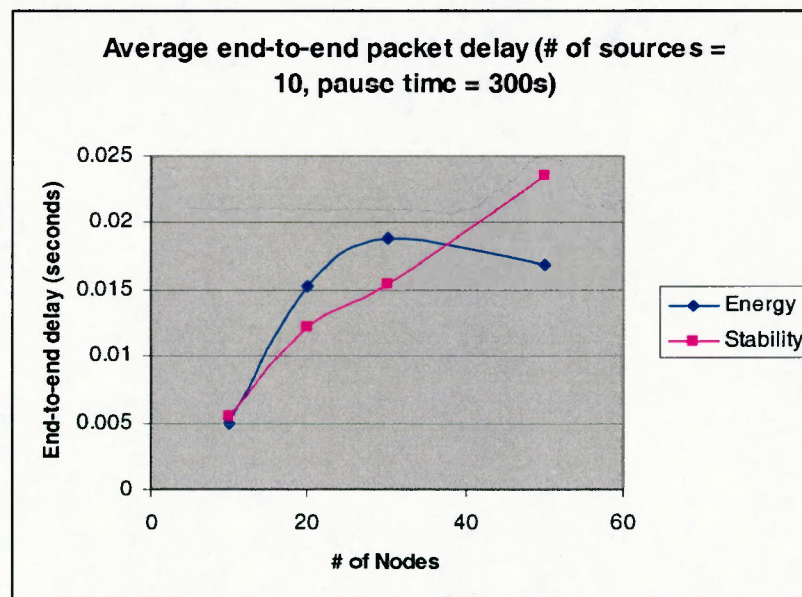


Figure 38 - Comparison of average end-to-end packet delay for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds

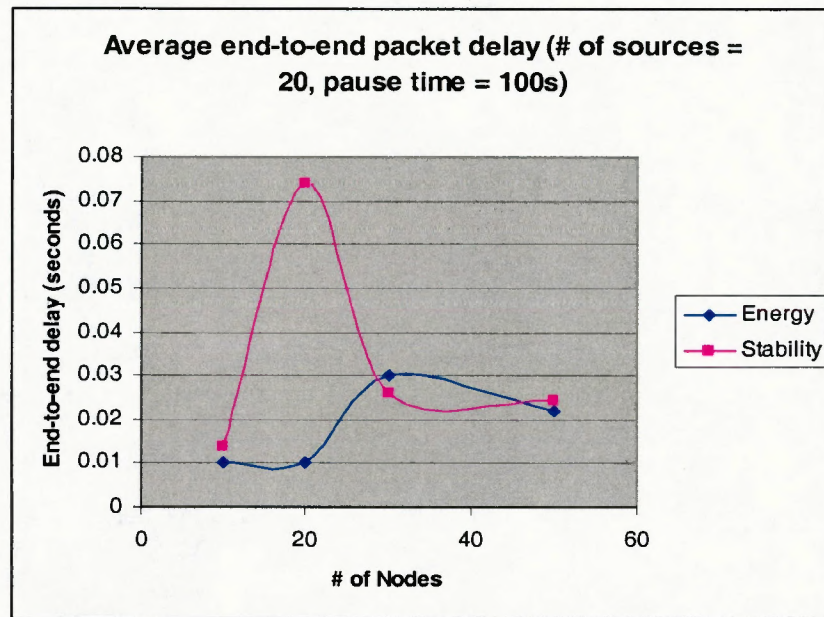


Figure 39 - Comparison of average end-to-end packet delay for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds

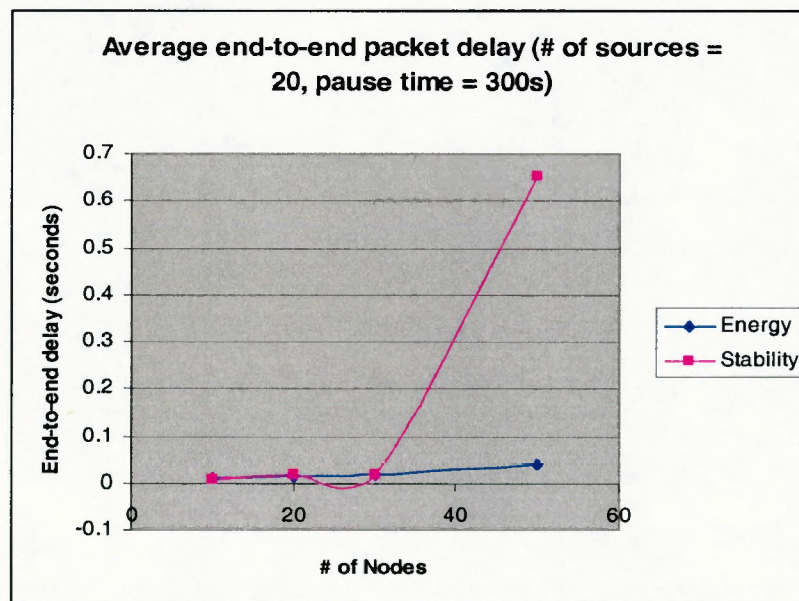


Figure 40 - Comparison of average end-to-end packet delay for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds

Throughput: As the network size and traffic load is increased, the throughput increases. If it is lower for 30-node networks than for 20-node networks in some scenarios, this could be attributed to the increased likelihood of stale routes due to topology changes and the delay involved in propagating topology changes throughout the network.

The throughput for the high-energy version of the proposed genetic protocol is significantly lower than that for the high-stability version, as can be seen in Figures 41, 42, 43 and 44. The high-stability protocol outperforms the high-energy protocol in all scenarios except when 20 traffic sources are used and node mobility is decreased. In this particular case, both protocols exhibit very similar trends with respect to average throughput, as shown in Figure 44.

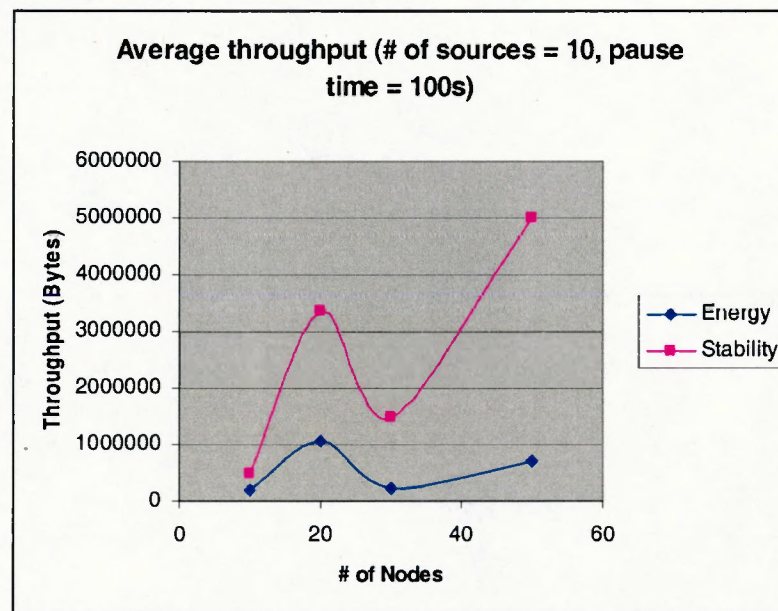


Figure 41 - Comparison of average throughput for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds

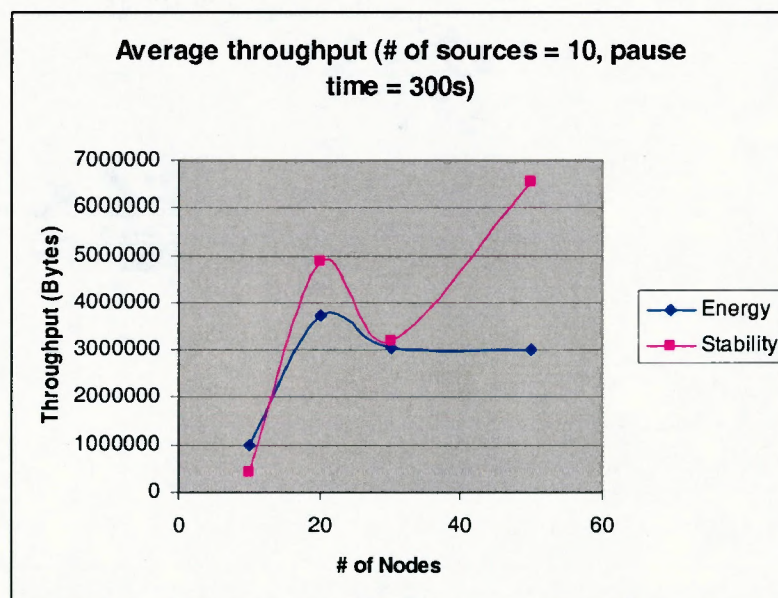


Figure 42 - Comparison of average throughput for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds

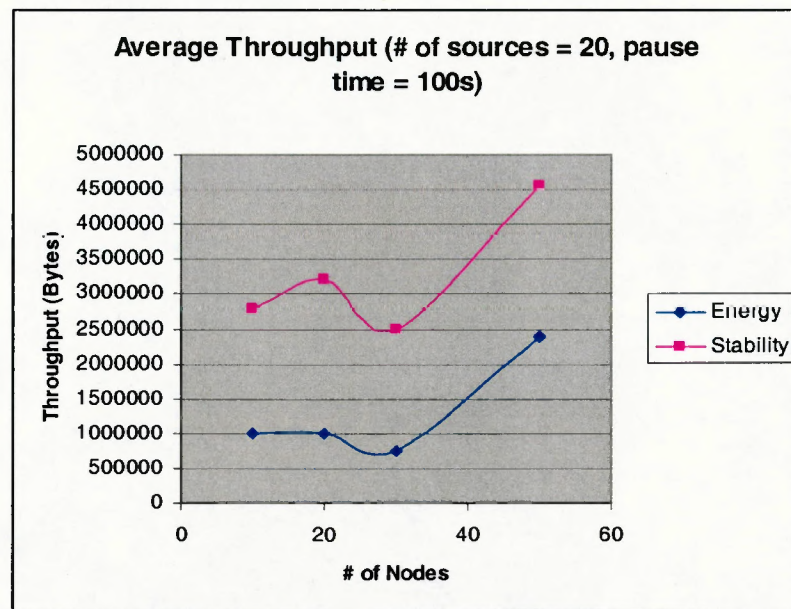


Figure 43 - Comparison of average throughput for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds

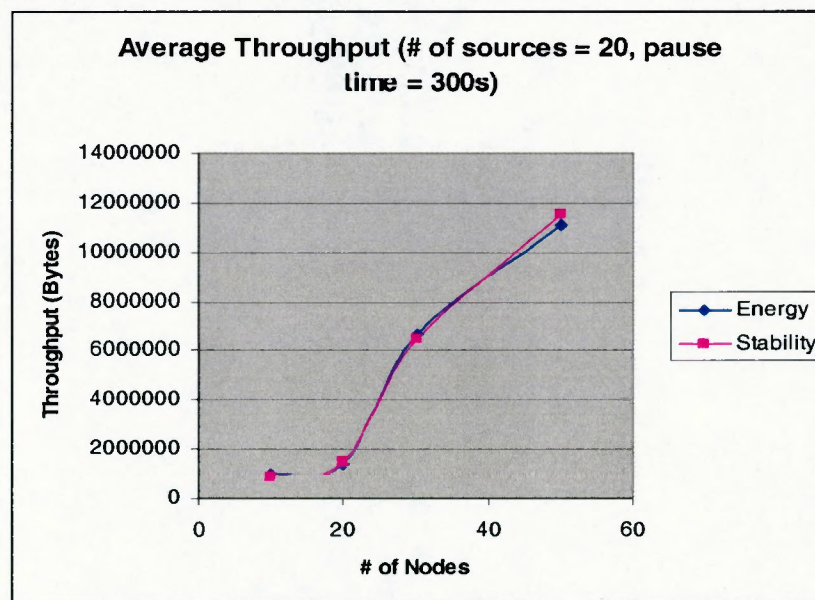


Figure 44 - Comparison of average throughput for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds

Packet delivery ratio: This parameter depends on the mobility pattern of the nodes in the given scenario, if the nodes are highly mobile this may result in many packets being sent that are dropped. Therefore, when the node mobility decreases, the packet delivery ratio improves considerably, as shown in Figures 46 and 48.

The high-energy version of the genetic protocol outperforms the high-stability version across a majority of the scenarios, as seen in Figures 45, 46, 47 and 48, although the high-stability genetic protocol performs better than the high-energy genetic protocol with respect to throughput. This may be attributed to differences in the actual numbers of packets being sent; also, the numbers of packets being dropped is more in the high-stability version, resulting in a lower packet delivery ratio.

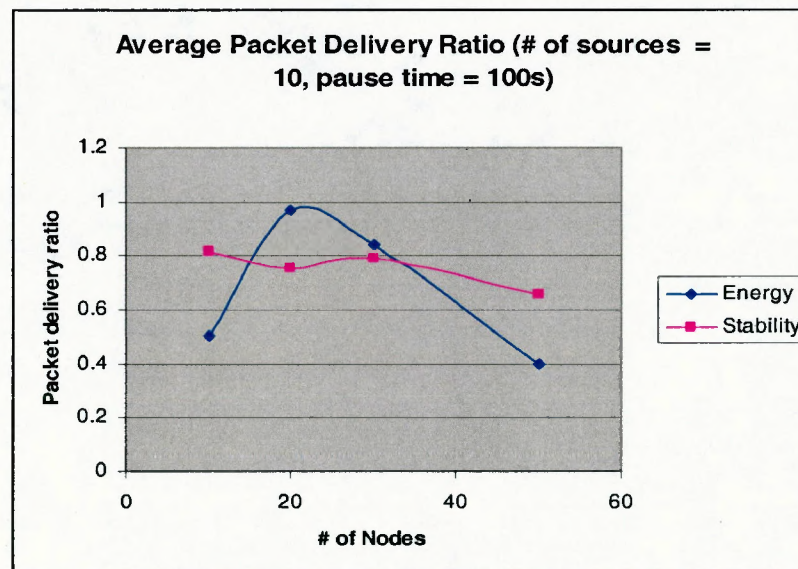


Figure 45 - Comparison of average packet delivery ratio for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds

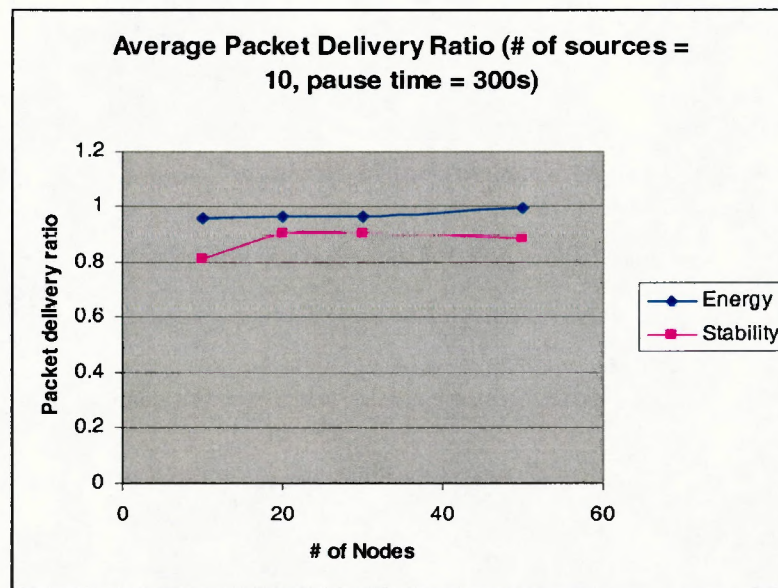


Figure 46 - Comparison of average packet delivery ratio for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds

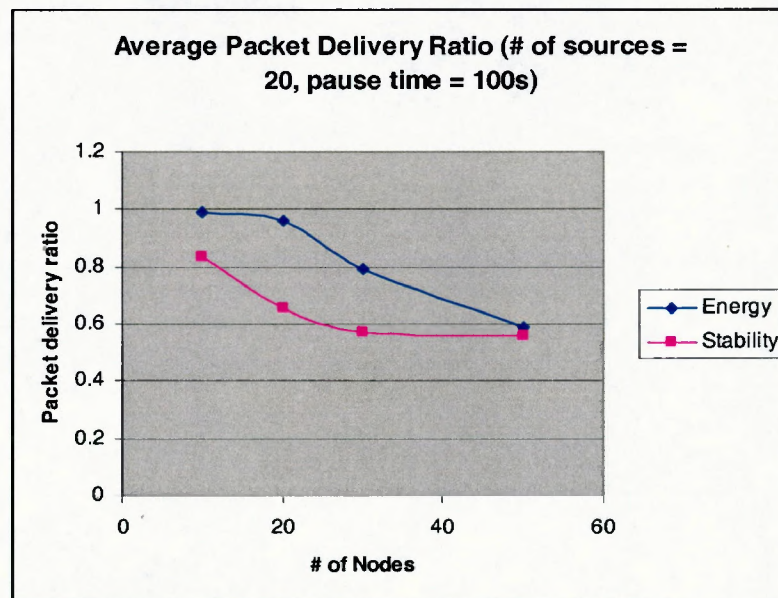


Figure 47 - Comparison of average packet delivery ratio for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds

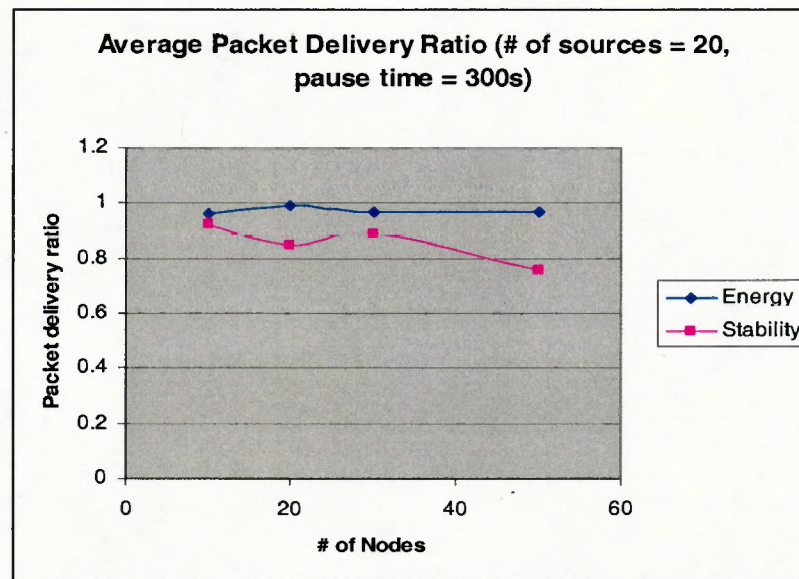


Figure 48 - Comparison of average packet delivery ratio for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds

Routing overhead: In this simulation study, the routing overhead is being measured as the percentage of routing packets sent compared to the total number of packets sent in the network, we observe that the routing overhead decreases as the network size increases. This may be attributed to the increase in the number of data packets, and therefore the total number of packets in the network. This implies that the number of routing packets increase although not as much as the increase in the number of data packets.

The high-energy version of the proposed genetic protocol performs better than the high-stability version with respect to routing overhead, across a majority of the scenarios, as shown in Figures 49, 50, 51 and 52.

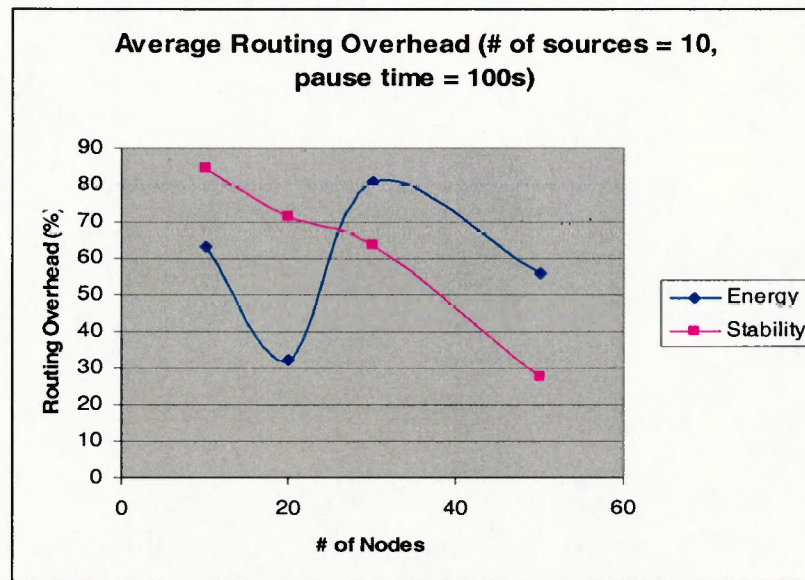


Figure 49 - Comparison of average routing overhead for the high-energy genetic protocol with 10 traffic sources and pause time = 100 seconds

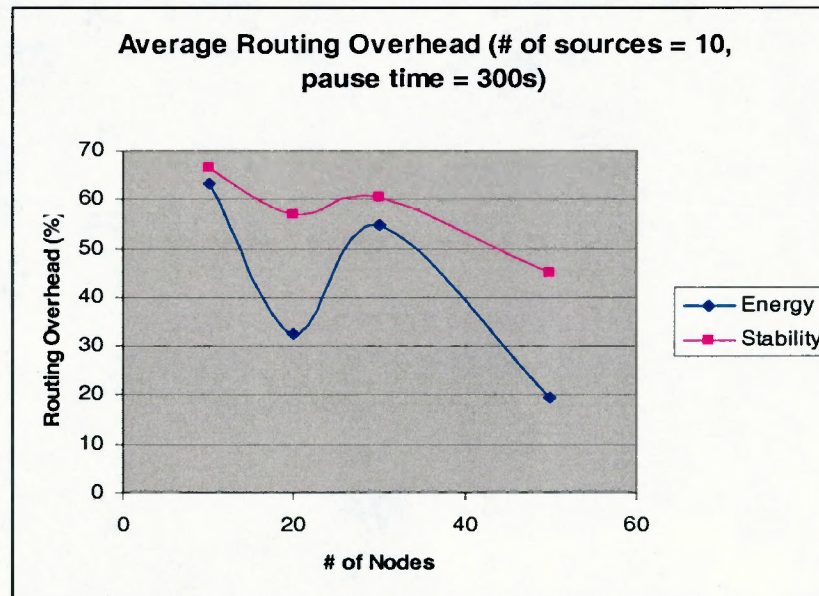


Figure 50 - Comparison of average routing overhead for the high-energy genetic protocol with 10 traffic sources and pause time = 300 seconds

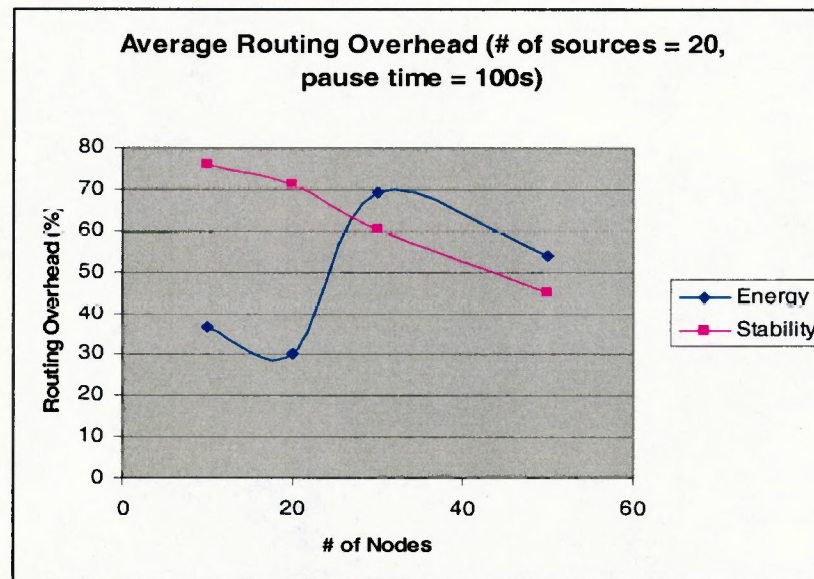


Figure 51 - Comparison of average routing overhead for the high-energy genetic protocol with 20 traffic sources and pause time = 100 seconds

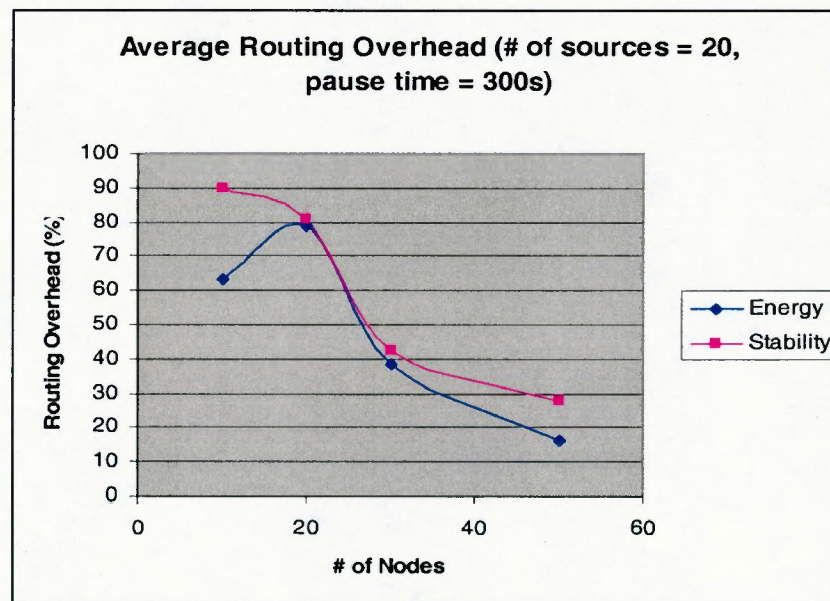


Figure 52 - Comparison of average routing overhead for the high-energy genetic protocol with 20 traffic sources and pause time = 300 seconds

In summary, the high-energy version of the genetic protocol performs well with respect to end-to-end packet delay, throughput, packet delivery ratio and routing overhead and outperforms the high-stability version of the genetic protocol with respect to the aforementioned performance indices across several scenarios. As described earlier, node energy is a very significant parameter in real-world mobile ad hoc networks that are comprised of battery powered mobile units and therefore, subject to strict resource constraints. This further highlights the importance of the finding that node energy may be incorporated into the genetic algorithm to obtain a competent level of routing performance. These experiments also demonstrate the flexibility of the fitness function in the genetic algorithm; it allows for different parameters to be explored while determining the fittest, or best, combination of backbone and non-backbone nodes in the network.

Chapter 7

An Evolutionary Dominating-set-based Routing Approach using Clusters of Nodes

Mobile ad hoc networks, or MANETs, tend to be clustered naturally into groups. For example, the clusters could be students in discussion groups in different parts of a room or soldiers in combat units, all parties equipped with mobile devices. Communication must take place both within the cluster (intra-cluster) as well as amongst clusters (inter-cluster). Instead of using a flat dominating-set-based routing strategy to communicate amongst all the mobile nodes, it would be more efficient to use different routing strategies for intra-cluster communication and inter-cluster communication.

In this chapter, we present a two-level hierarchical evolutionary dominating-set-based routing strategy for mobile ad hoc networks. The network is divided into groups of nodes, or clusters. Within each cluster, there exists a self-organizing, dynamic virtual backbone that is constructed and updated using the genetic algorithm described in Chapter 5. Between clusters, mobile nodes communicate via gateways, modeled using Base Stations. Here, the overhead of computing and refreshing the virtual backbone for the entire network would be greatly reduced, particularly as the network size increases, and this in turn should improve the routing performance significantly. We conduct detailed simulations using ns-2 to evaluate the routing performance of a

clustered evolutionary dominating-set-based routing strategy and compare this performance with the previous flat evolutionary dominating-set-based routing approach.

As before, our evaluations are based on the simulation of 10, 20, 30 and 50 nodes forming an ad hoc network, moving within a 670m X 670m grid for 500 seconds of simulated time. CBR traffic with a packet size of 512 bytes was used as data traffic in all simulations. We used 10 and 20 traffic sources with a sending rate of 4 packets/second across all scenarios. The speed of the mobile nodes was set to a maximum of 20 m/s. In all scenarios, the network was divided into clusters of 10 nodes each. Each cluster has a gateway associated with it, through which packets are routed between clusters.

The indices used to measure and evaluate routing performance were the same as those used in the previous simulations: end-to-end packet delay, throughput, packet delivery ratio and routing overhead.

7.1 End-to-end packet delay

With respect to average end-to-end packet delay, the clustered genetic dominating-set-based routing protocol outperforms the non-clustered genetic protocol across a majority of the scenarios, as shown in Figures 53, 54, 55 and 56. When 10 traffic sources are used, and node mobility is decreased, the non-clustered genetic protocol

yields a smaller delay than the clustered genetic protocol, as shown in Figure 54. This is very likely since there is a very slight difference in architecture between a flat 10-node network and a network comprising 10 nodes in a single cluster. When 20 traffic sources are used and node mobility is decreased, both the clustered as well as non-clustered genetic protocols perform comparably well; however, the clustered genetic protocol performs significantly better than the non-clustered genetic protocol for large 50-node networks, as shown in Figure 56. This re-emphasizes the fact that clustering is very useful, particularly for large networks.

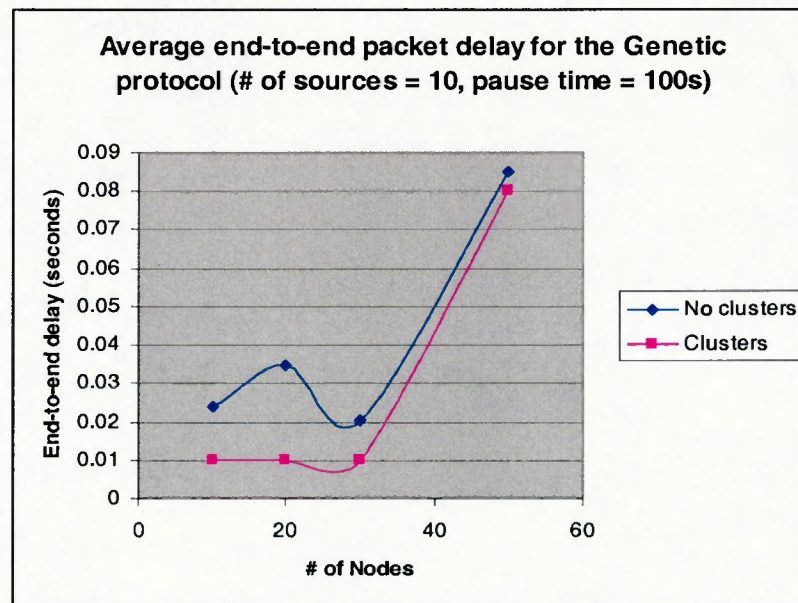


Figure 53 - Comparison of average end-to-end packet delay for clustering with 10 traffic sources and pause time = 100 seconds

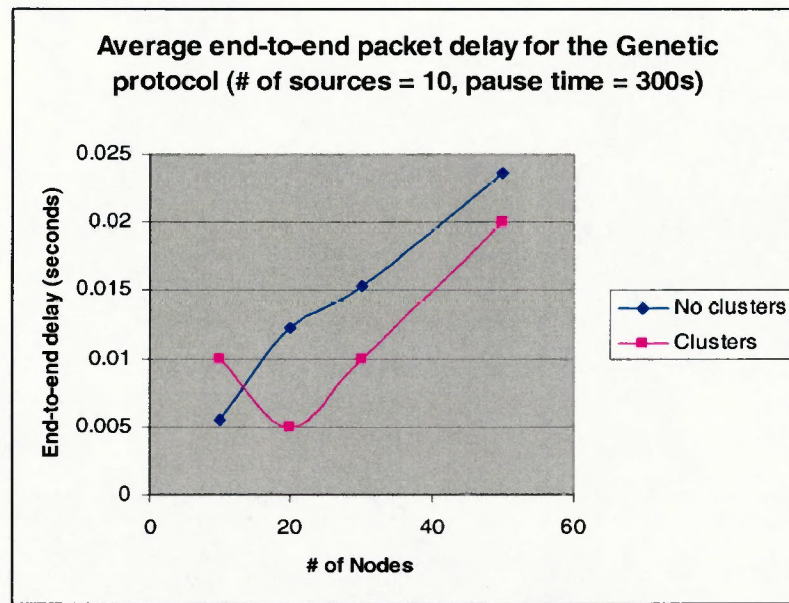


Figure 54 - Comparison of average end-to-end packet delay for clustering with 10 traffic sources and pause time = 300 seconds

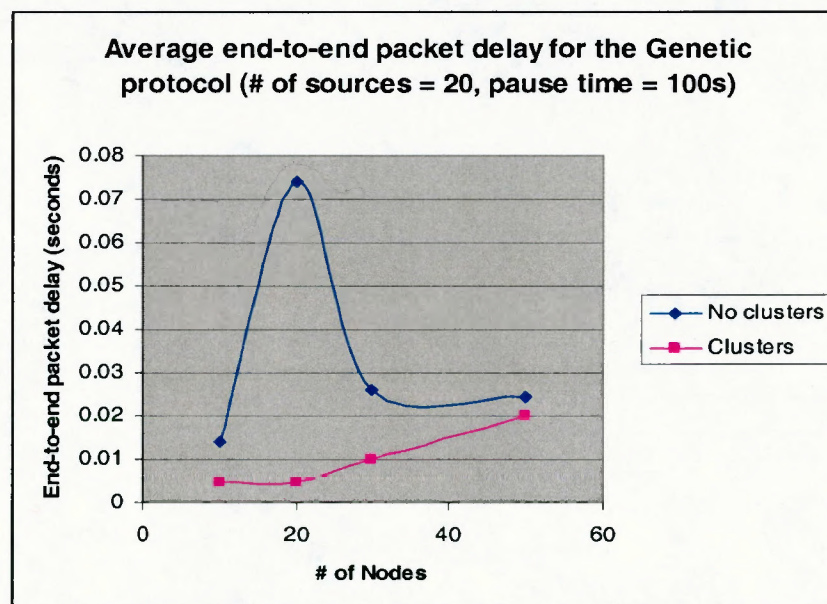


Figure 55 - Comparison of average end-to-end packet delay for clustering with 20 traffic sources and pause time = 100 seconds

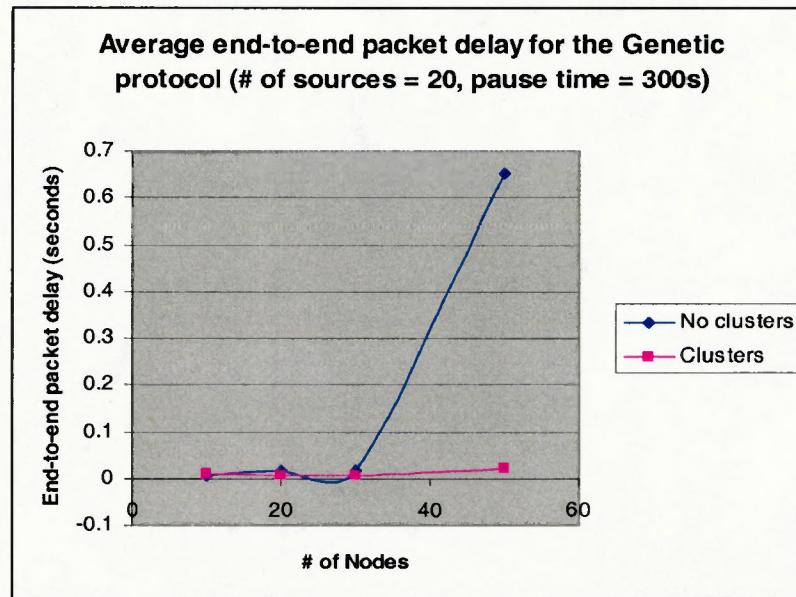


Figure 56 - Comparison of average end-to-end packet delay for clustering with 20 traffic sources and pause time = 300 seconds

7.2 Throughput

With respect to throughput, the clustered genetic protocol significantly outperforms the non-clustered protocol across all scenarios, as shown in Figures 57, 58, 59 and 60. This implies that clustering networks into smaller groups results in efficient construction of the virtual backbone, quick topology refreshing and less chances of stale routes; this in turn reduces packet loss and enhances throughput in the network.

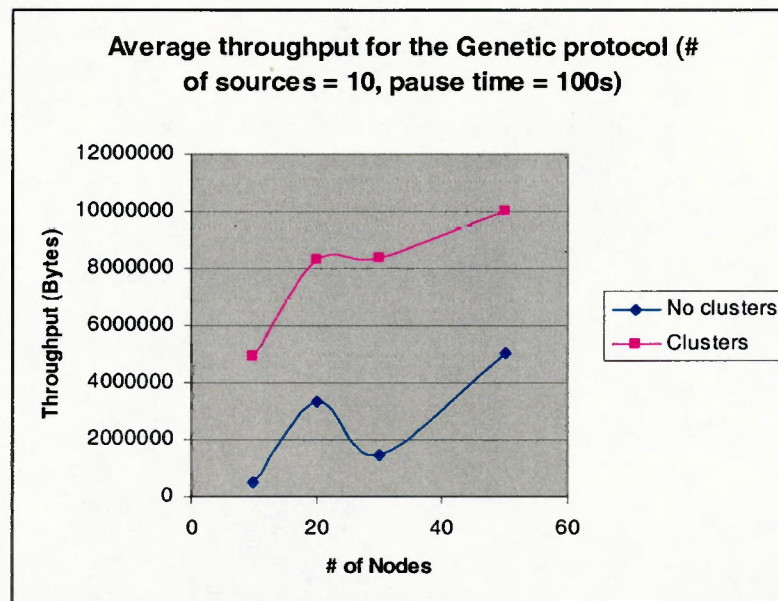


Figure 57 - Comparison of average throughput for clustering with 10 traffic sources and pause time = 100 seconds

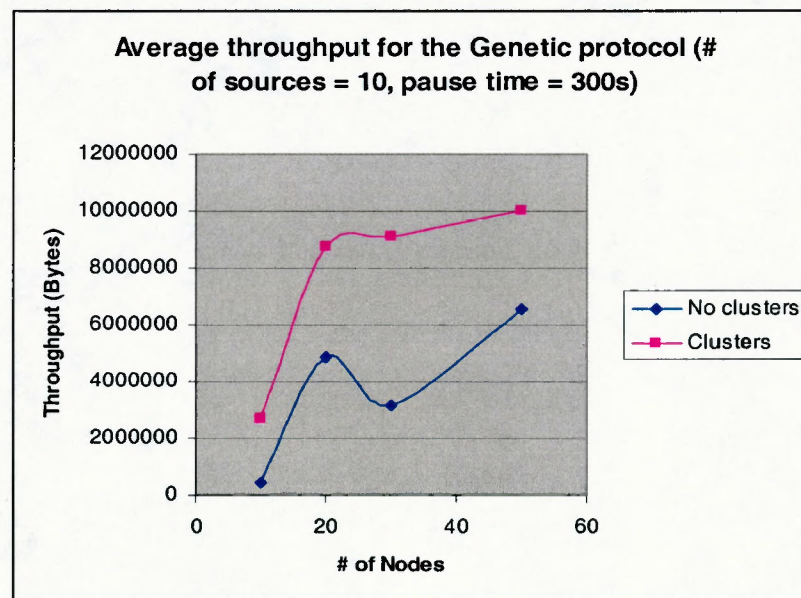


Figure 58 - Comparison of average throughput for clustering with 10 traffic sources and pause time = 300 seconds

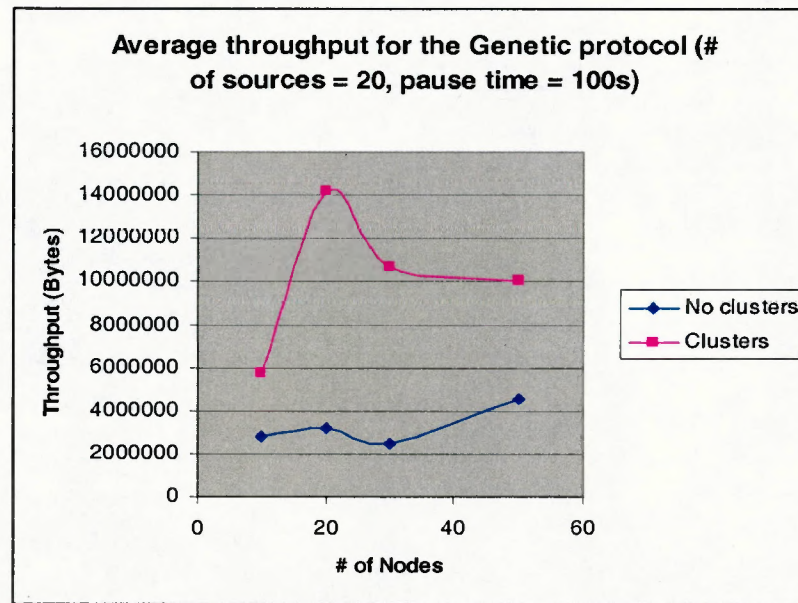


Figure 59 - Comparison of average throughput for clustering with 20 traffic sources and pause time = 100 seconds

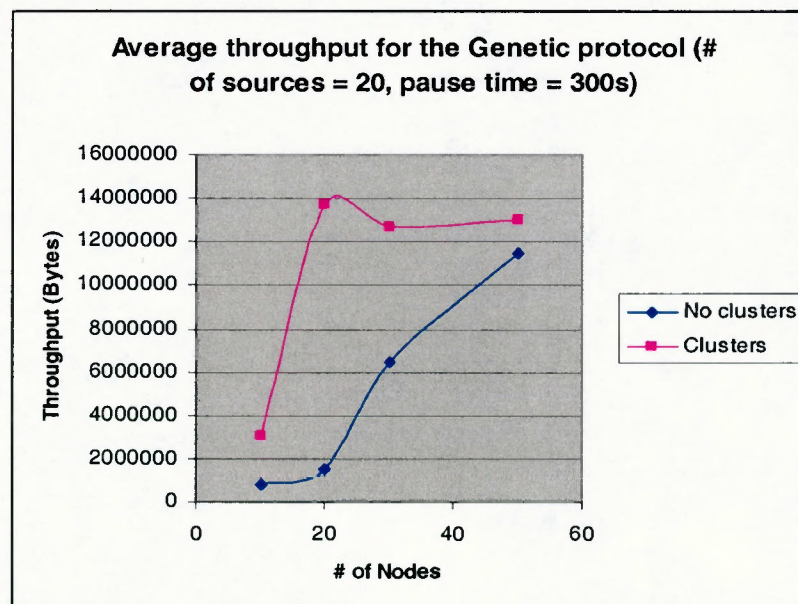


Figure 60 - Comparison of average throughput for clustering with 20 traffic sources and pause time = 300 seconds

7.3 Packet delivery ratio

With respect to packet delivery ratio, the clustered genetic protocol outperforms the non-clustered genetic protocol across a majority of the scenarios, as shown in Figures 61, 62, 63 and 64. This could be attributed to the efficient construction and refreshing of the virtual backbone within each cluster as well as efficient routing between the gateway nodes.

When the number of traffic sources is increased and node mobility is decreased, the performance of both the clustered and non-clustered protocols is comparable, as shown in Figure 64. When node mobility decreases, the nodes are more stable, there are less topology changes and the performance of the non-clustered protocol improves with respect to packet delivery. The non-clustered protocol outperforms the clustered protocol for networks comprising 10 and 20 nodes, and the clustered protocol performs better in larger networks. Once again, this emphasizes the advantages of using clustering to route efficiently, particularly in large networks where a large backbone must be constructed and updated regularly, and updates must reach all nodes in the network. This overhead is minimized by using a clustering approach.

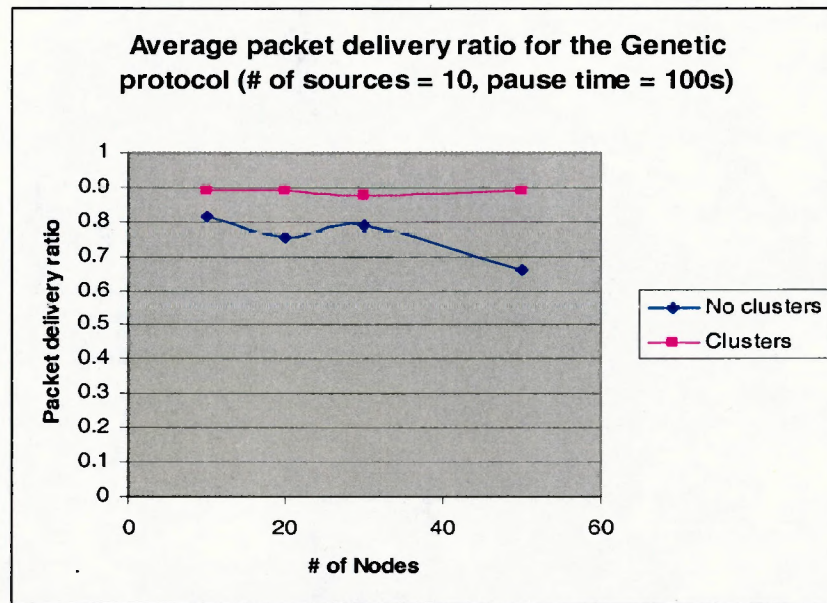


Figure 61 - Comparison of average packet delivery ratio for clustering with 10 traffic sources and pause time = 100 seconds

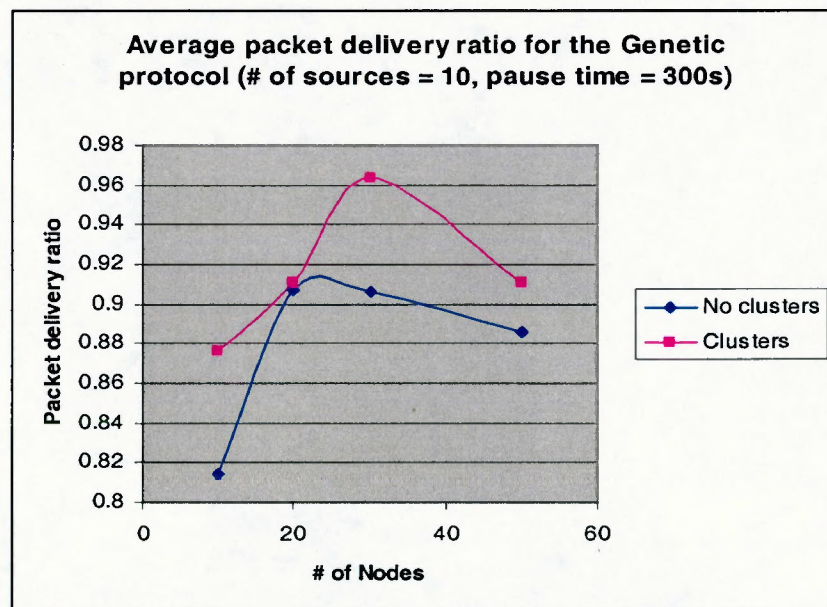


Figure 62 - Comparison of average packet delivery ratio for clustering with 10 traffic sources and pause time = 300 seconds

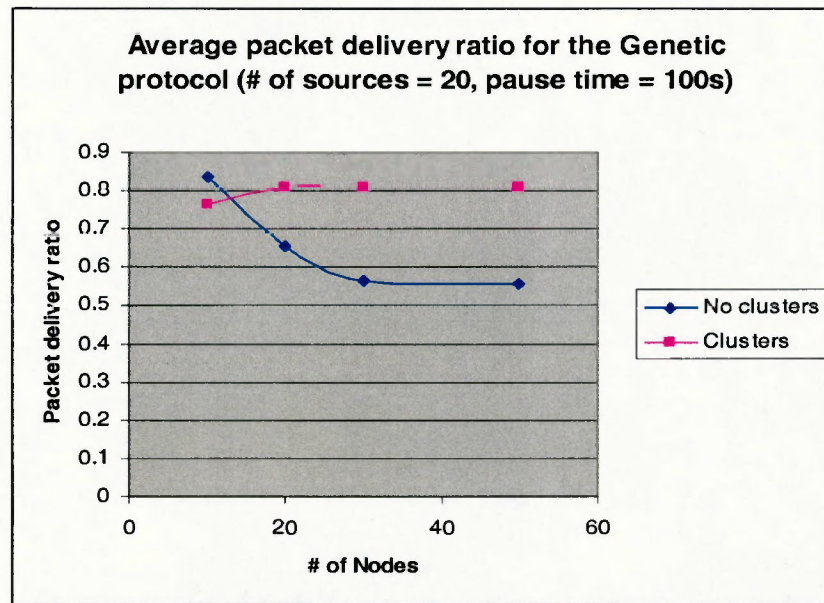


Figure 63 - Comparison of average packet delivery ratio for clustering with 20 traffic sources and pause time = 100 seconds

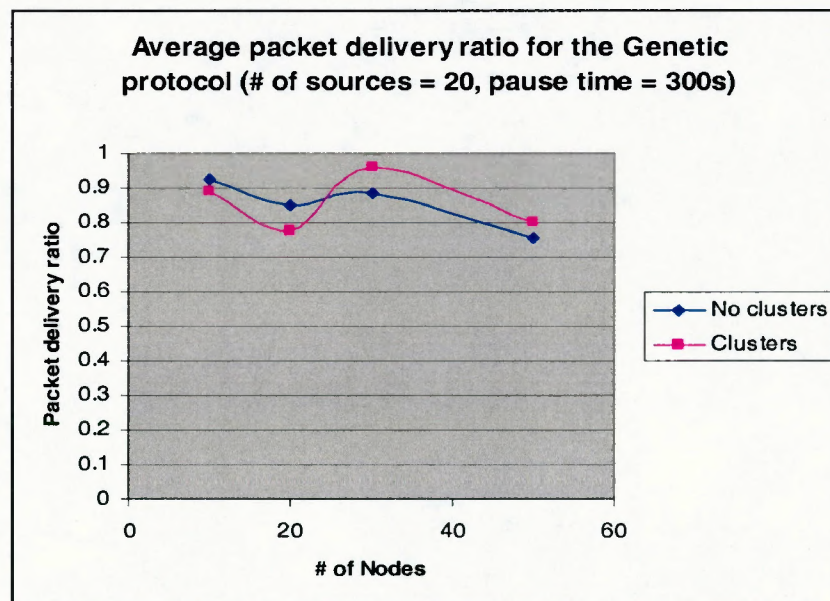


Figure 64 - Comparison of average packet delivery ratio for clustering with 20 traffic sources and pause time = 300 seconds

7.4 Routing Overhead

With respect to routing overhead, the clustered genetic protocol outperforms the non-clustered genetic protocol across a majority of the scenarios, as shown in Figures 65, 66, 67 and 68. The clustered genetic protocol yields lower overhead consistently. As described earlier, the routing overhead is measured as a ratio of the number of routing packets to the total number of packets sent during the simulation. This exhibits a decreasing trend because although the number of routing packets increases with an increase in network size, it does not increase by as much as the increase in the number of data packets. Therefore, the increase in the number of routing packets is offset by the substantial increase in the number of data packets.

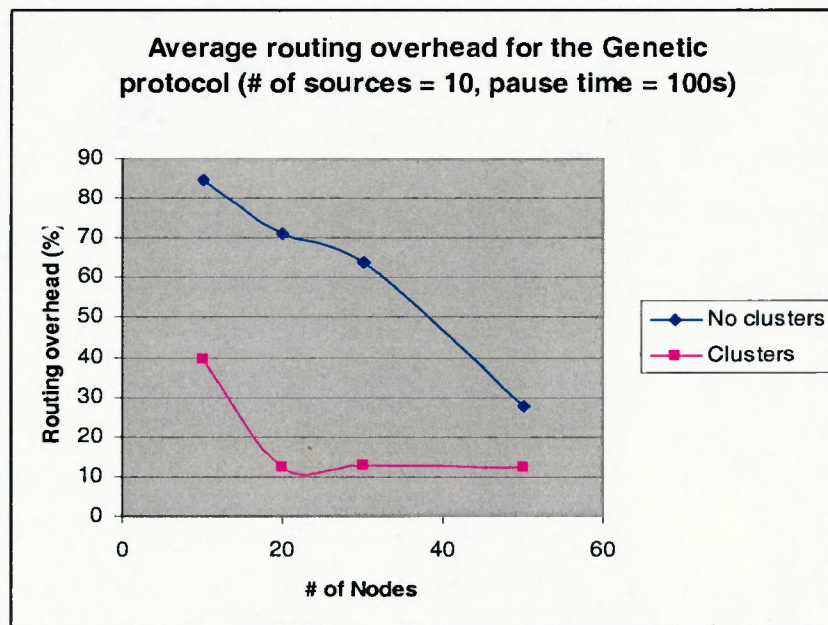


Figure 65 - Comparison of average routing overhead for clustering with 10 traffic sources and pause time = 100 seconds

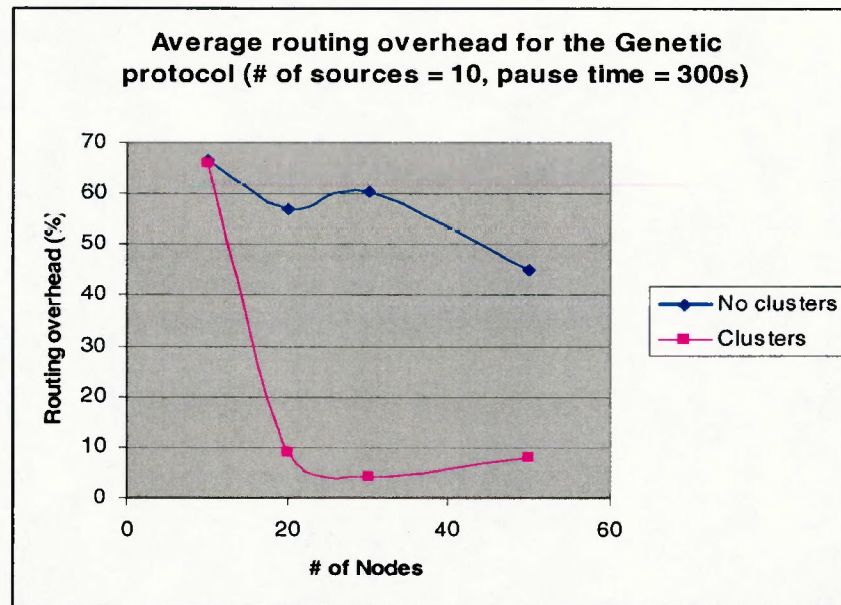


Figure 66 - Comparison of average routing overhead for clustering with 10 traffic sources and pause time = 300 seconds

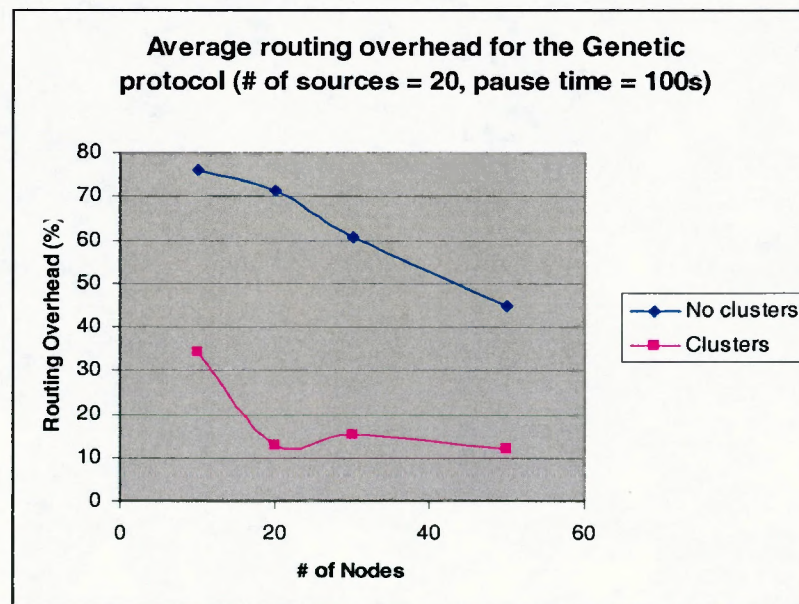


Figure 67 - Comparison of average routing overhead for clustering with 20 traffic sources and pause time = 100 seconds

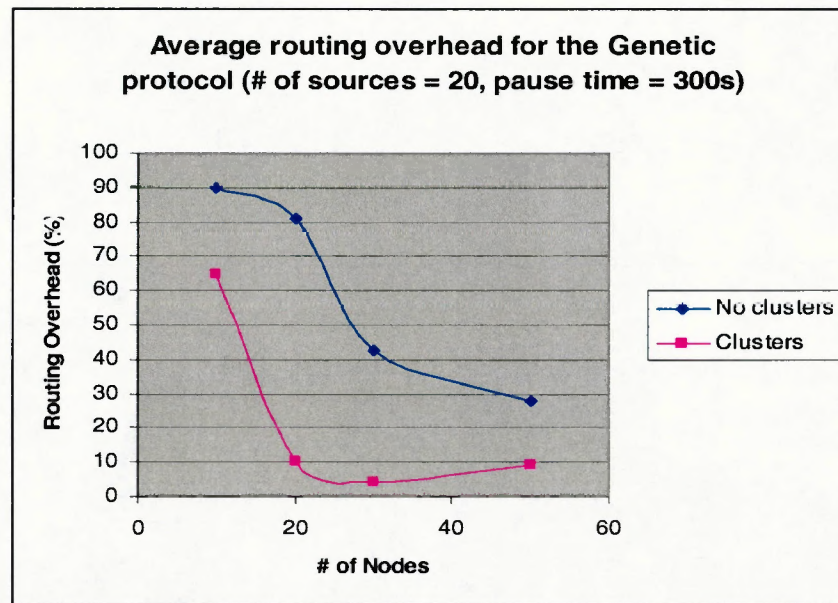


Figure 68 - Comparison of average routing overhead for clustering with 20 traffic sources and pause time = 300 seconds

In summary, the clustered genetic protocol performs well with respect to end-to-end packet delay, throughput, packet delivery ratio and routing overhead and outperforms the non-clustered genetic protocol with respect to the aforementioned performance indices across most scenarios, most significantly for large networks. This highlights the importance of clustering to improve the efficiency and routing performance of a dominating-set-based routing strategy, in particular for large networks. Since the concept of clustering in mobile ad hoc networks has direct real-world applications, such as combat units of soldiers communicating within and between units, the enhanced performance that clustering yields is an important finding that provides immense potential for future research and practical use.

Chapter 8

Summary and Conclusions

This chapter summarizes the work accomplished in this thesis, the objectives achieved, and conclusions drawn from the performance evaluation of the dominating-set-based routing protocols. This chapter also provides suggestions for future work.

8.1 Background

A mobile ad hoc network, or MANET, is a collection of mobile computers that form a temporary network without any pre-established, centralized infrastructure. Each mobile node behaves as a router and participates in the forwarding of packets sent from a source mobile node to a destination mobile node. This type of communication is known as *multihop* communication, and is one of the primary characteristics of a MANET. Other characteristics of MANETs that present challenges in the design of routing protocols for such wireless environments include: dynamic topology, limited energy and bandwidth resources and security vulnerabilities.

Existing routing protocols for ad hoc networks may be classified as *proactive* or *reactive*, based on how they react to topology changes in the network. Proactive, or table-driven, protocols attempt to maintain routes continuously, such that the route is already available when it is needed for forwarding a packet. In such protocols, routing

tables are exchanged among neighboring nodes every time a change occurs in the network topology. Therefore, a route between any source-destination pair is always available but the maintenance of consistent and updated routes requires the propagation of large amounts of routing information throughout the network. The control overhead in terms of network traffic and power consumption proves to be a serious limitation in MANETs, where bandwidth and power are scarce resources.

On the other hand, reactive, or on-demand, or source-initiated, protocols send control messages to discover a route between a given source-destination pair only when necessary. This reduces the control overhead substantially; however, the route discovery process generates a latency period that has a negative impact on routing performance.

Both proactive as well as reactive protocols employ “flooding” to disseminate control packets for topology updates and route discovery. Flooding is a broadcast mechanism that causes redundancy, contention and collision resulting in a high packet loss rate. The problem identified in this thesis involves the design of a robust and efficient routing protocol for mobile ad hoc networks that minimizes the flooding problem. This can be achieved by restricting broadcast to a subset of the mobile nodes in the ad hoc network, called the “virtual backbone”. This would minimize communication overhead and optimize the usage of critical resources in the network.

A virtual backbone may be constructed by approximating a *dominating set* that uses unicast to replace flooding. This backbone is a self-organized structure that is primarily used to compute and update routes and may also provide a backup route. Since this backbone is dynamic, it must be recomputed periodically and this presents a challenging problem in the design and implementation of the routing protocol. The dominating-set-based routing approach may be classified as a hybrid approach between the proactive approach and the reactive approach, wherein the nodes that constitute the dominating set maintain routing tables and the nodes outside the dominating set request for routes as required from the members of the dominating set.

8.2 Objectives Achieved

A literature search in the area of routing in mobile ad hoc networks using dominating sets was conducted. This provided valuable information on the existing approaches to dominating-set-based routing. To the best of our knowledge, these routing approaches have not been implemented, evaluated or compared with respect to network routing performance on a standard simulation platform. This study is the first to implement and analyze the routing performance of the protocols based on the dominating set approach using the network simulator ns-2. This thesis then proposed a new evolutionary approach to routing using dominating sets based on genetic algorithms.

In order to learn how to apply the genetic algorithm approach to wireless networks as well as learn how to use the network simulator ns-2, this thesis involved an in-depth

study of how genetic algorithms may be used to solve a contemporary problem in Bluetooth personal area wireless networks: the problem of Scatternet formation, or formation of multihop Bluetooth networks. The knowledge gained from this work was further utilized to accomplish the primary goal of this thesis: to design an evolutionary routing protocol for ad hoc wireless networks.

In this thesis, we introduce an evolutionary approach to construct a dominating set that avoids flooding and can be utilized to route packets efficiently through ad hoc wireless networks. This approach uses a genetic algorithm to find the best, or “fittest”, dominating set in the graph underlying the network. The fitness of the dominating set is evaluated by measuring the stability of the nodes. The proposed evolutionary approach takes advantage of the evolutionary nature of the underlying network: ad hoc wireless networks evolve as nodes move and change positions. This evolution of networks lends itself naturally to a genetic algorithm model.

8.3 Performance Evaluation

The performance results of our approach, obtained by conducting extensive simulations using ns-2, have been positive and encouraging. Through this performance evaluation, we demonstrate that the proposed genetic algorithm results in the construction of a robust virtual backbone that can route packets effectively.

We evaluated the performance of the proposed evolutionary (or genetic) routing protocol by measuring end-to-end packet delay, throughput, packet delivery ratio and routing overhead. We compared the performance of the genetic protocol with the performance of other known dominating-set-based routing protocols. The proposed genetic protocol performs best with respect to end-to-end packet delay across several different scenarios. With respect to throughput, the genetic protocol significantly outperforms the other protocols in large networks as well as in most scenarios when node mobility is decreased. The genetic protocol performs comparably well with respect to the percentage of packets delivered and routing overhead across most scenarios.

We also measured certain characteristics of the dominating set generated using the genetic algorithm: size of the dominating set (or number of nodes that comprise the virtual backbone) and network diameter. The size of the dominating set was always found to be less than half the number of nodes in the network, for all network sizes in the simulation study. This implies that flooding in the network is restricted to less than half the nodes in the network, resulting in improved routing performance. The diameter was found to be equal to approximately half the nodes in the network for all network sizes in the simulation study. From this we conclude that since the diameter is finite, the network is connected at all times and the greatest number of hops a packet needs to be travel to reach its destination is about half the number of nodes in the network.

Mobile node energy is a very significant parameter in real-world mobile ad hoc networks and sensor networks that are subject to strict power constraints. Therefore, it is important to incorporate node energy into the genetic algorithm, thereby transforming the algorithm into a resource-aware, adaptive algorithm. Further experiments with the evolutionary routing approach included modifying the fitness function to select the nodes that have the highest energy as members of the dominating set or virtual backbone. We evaluated and compared the routing performance of a backbone generated based on energy as a fitness function with the backbone generated in our previous experiments with stability as a fitness function. The high-energy version of the genetic protocol performs well with respect to end-to-end packet delay, throughput and routing overhead and outperforms the high-stability version of the genetic protocol with respect to the aforementioned performance indices across several scenarios. In particular, these experiments demonstrated the flexibility of the fitness function in the genetic algorithm; different parameters may be explored simultaneously while determining the fittest, or best, combination of backbone and non-backbone nodes in the network.

Mobile ad hoc networks may consist of naturally occurring clusters, or groups, such as combat units of soldiers who must communicate within each unit (or cluster) as well as between units. In this thesis, the phenomenon of node clustering while using an evolutionary dominating-set-based routing approach was investigated. Within each

cluster, packets are routed by a virtual backbone constructed using the proposed genetic algorithm. Each cluster has a gateway that routes packets between clusters. The clustered genetic protocol performs well with respect to end-to-end packet delay, throughput, packet delivery ratio and routing overhead and outperforms the non-clustered genetic protocol with respect to the aforementioned performance indices across most scenarios, most significantly for large networks. From these results, we conclude that clustering is a simple yet effective means to improve the efficiency and routing performance of a dominating-set-based routing strategy, in particular for large networks.

8.4 Future Work

Future research efforts could further investigate enhancements or modifications to the evolutionary dominating-set-based routing approach. The fitness function may be modified to include multiple network parameters and experiments conducted to observe the effect of these factors on routing performance. Future research efforts could also extend this work by measuring the characteristics of the dominating set, such as size and network diameter, for all the dominating-set-based routing protocols and comparing the results with those obtained for the evolutionary dominating-set-based approach. The clustering phenomenon may be studied further by observing the effect of clustering for all the dominating-set-based routing protocols in order to compare the effect of clustering using the evolutionary routing approach with the other approaches. Future work could also compare the performance of dominating-

set-based routing protocols with existing protocols for mobile ad hoc networks, such as DSDV, DSR, AODV and TORA.

References:

1. UCB/LBNL/VINT Network simulator ns-2 <http://www.isi.edu/nsnam/ns/>
2. The Bluetooth Special Interest Group
<http://www.bluetooth.com>
3. Bluehoc – Open Source Bluetooth Simulator
<http://www-124.ibm.com/developerworks/opensource/bluehoc/>
4. “Bluetooth – An Overview”,
Johnson Consulting
<http://www.swedetrack.com/images/bluet10.htm>
5. The Genetic Algorithms Archive, <http://www.aic.nrl.navy.mil/galist/>
6. IETF MANET Working Group, <http://www.ietf.org/html.charters/manet-charter.html>
7. Alok Aggarwal, Manika Kapoor, Lakshmi Ramachandran and Abhinanda Sarkar,
“Clustering Algorithms for Wireless Ad Hoc Networks”, *In Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 2000.
8. K.M. Alzoubi, P.-J. Wan and O. Frieder, "New distributed algorithm for connected dominating set in wireless ad hoc networks", *Proc. HICSS 2002*.

9. S. Basagni, R. Bruno and C. Petrioli, "A Performance Comparison of Scatternet Formation Protocols for Networks of Bluetooth Devices", IEEE PERCOM 2003.
10. S. Basagni, C. Petrioli and I. Chlamtac, "Configuring BlueStars: Multihop scatternet formation for Bluetooth networks", *IEEE Transactions on Computers, special issue on Wireless Internet, 2002. in press.*
11. Jan Beutel, "Wearable Computing – What It Is and Where It is Going", August 2001
12. Pravin Bhagwat and Srinivasa Rao, "On the Characterization of Bluetooth Scatternet Topologies", *Submitted for publication.*
13. J. Broch, D. B. Johnson, and D.A. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks", Internet draft, Dec. 1998
14. J. Broch, D.A. Maltz, D. B. Johnson, Y.-C Hu, J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", *Proceedings of MobiCom'98*, Dallas, TX, October 1998.
15. X. Cheng, D. Du, "Virtual Backbone-Based Routing in Multihop Ad Hoc Wireless Networks", 2001
16. I. Cidon and O. Mokryn, "Propagation and Leader Election in Multihop Broadcast Environment", *12th International Symposium on Distributed Computing (DISC98)*, Greece, September 1998.
17. S. Corson and J. Macker, "Mobile Ad Hoc Networking (MANET), IETF RFC 2501, Jan. 1999.

18. B. Das and V. Bharghavan, "Routing in Ad Hoc Networks using Minimum Connected Dominating Sets", *ICC '97*, Montreal, Canada, June 1997.
19. B. Das, R. Sivakumar and V. Bharghavan, "Routing in Ad Hoc Networks Using a Spine", *International Conference on Computers and Communications Networks '97*, Las Vegas, NV, September 1997.
20. B. Das, R. Sivakumar and V. Bharghavan, "An Improved Spine-based Infrastructure for Routing in Ad Hoc Networks", *IEEE Symposium on Computers and Communications '98*, Athens, Greece, June 1998.
21. B. Das, R. Sivakumar and V. Bharghavan, "The Clade Vertebrata: Spines and Routing in Ad Hoc Networks", *Proceedings of the International Symposium on Computer Communications (ISCC '98)*, 1998.
22. R. Dube, C.D. Rais, K. Wang and S. K. Tripathi, "Signal Stability-Based Adaptive Routing (SSA) for Ad Hoc Mobile Networks", *IEEE Personal Communications*, Feb. 1997.
23. M. L. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", San Francisco: W. H. Freeman, 1979.
24. S. Guha and S. Khuller, "Approximation Algorithms for Connected Dominating Sets", Tech. Rep. 3660, Univ. of Maryland Institute for Advanced Computer Studies – Dept. of Computer Science, Univ. of Maryland, College Park, June 1996.
25. Z.J. Haas and M.R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", Internet draft, Aug. 1998

26. Per Johansson and Johan Sorensen, "Ad Hoc IP Networks over Bluetooth", 2001.
27. P. Johansson, T. Larsson, N. Hedman, B. Mielczarek and M. Degermark, "Scenario Based Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks", *Proc. IEEE MOBICOM*, Seattle, Aug. 1999, pp. 195-206.
28. Ching Law, Amar K. Mehta and Kai-Yeung Siu, "Performance of a New Bluetooth Scatternet Formation Protocol", *In Proceedings of ACM Symposium on Mobile Ad Hoc Networking and Computing*, 2001.
29. Gy. Miklos, A. Racz, Z. Turanyi, A. Valko and P. Johansson, "Performance aspects of Bluetooth scatternet formation", *In Proceedings of the First Annual Workshop on Mobile Ad Hoc Networking and Computing*, 2000.
30. S. Murthy and J.J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks", *ACM Mobile Networks and Applications*, Oct. 183-197, 1996.
31. S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network", *Proc. MOBICOM*, Seattle, Aug. 1999, pp. 151-162.
32. Vincent D. Park and M. Scott Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", *Proceedings of IEEE INFOCOM '97*, , April 1997.
33. C. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector (DSDV) Routing for Mobile Computers", in *ACM SIGCOMM*

- Symposium on Communications, Architectures and Protocols, September 1994, pp. 234-244.
34. C. Perkins and E.M. Royer, "Ad Hoc On Demand Distance Vector (AODV) Routing", Internet draft, Aug. 1998
 35. E.M. Royer and C.K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", *IEEE Personal Communications*, Apr., 46-55, 1999.
 36. Theodoros Salonidis, Pravin Bhagwat, Leandros Tassiulas and Richard LaMaire, "Distributed Topology Construction of Bluetooth Personal Area Networks", *In Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communication Societies*, 2001.
 37. P. Sinha, R. Sivakumar and V. Bharghavan, "Enhancing Ad hoc Routing with Dynamic Virtual Infrastructures", *IEEE Infocom 2001*, Anchorage, Alaska, March 2001.
 38. I. Stojmenovic, "Dominating set based Bluetooth scatternet formation with localized maintenance", *In Proc. Of Workshop on Advances in Parallel and Distributed Computational Models at IEEE PDPS*, 2002.
 39. I. Stojmenovic, "Handbook of Wireless Networks and Mobile Computing", 2002.
 40. G. Tan, Al. Miu, J. Guttag et al, "Forming Scatternets from Bluetooth Personal Area Networks", MIT Technical Report, Oct 2001.
 41. J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks", *Proc. of the 3rd International Workshop*

on Discrete Algorithms and Methods for MOBILE Computing and Communications, 1999, Seattle, WA USA, pp.7-14.

42. J. Wu and H. Li, "On Calculating Power-Aware Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks", *International Conference on Parallel Processing (ICPP '01)*, Spain, September 2001.
43. G. Zaruba, S. Basagni and I. Chlamtac. "BlueTrees – Scatternet formation to enable Bluetooth-based personal area networks", *In Proceedings of the IEEE International Conference on Communications, ICC 2001, Helsinki, Finland*, June 2001.
44. H. Zhang, Jennifer C. Hou and Lui Sha, "Bluetooth Loop Scatternet Formation Algorithm", *IEEE International Conference on Communications*, May 2003.
45. Bin Zhen, Jonghun Park and Yongsuk Kim, "Scatternet Formation of Bluetooth Ad Hoc Networks", *In Proceedings of the 36th Hawaii International Conference on System Sciences*, 2003.