Swarm Intelligence Movement Control in a MANET based on Awareness of Traffic Condition

by

Hanin A. Almutairi

Bachelor of Computer Science, Taibah University, 2008

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisor: Przemyslaw R. Pochec, Professor, Computer Science
John M. DeDourek, Professor, Computer Science

Examining Board: Rodeny H. Cooper, Professor, Computer Science, Chair
Virendra Bhavsar, Professor, Computer Science
Maryhelen Stevenson, Professor, Electrical & Computer Engineering

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

August, 2015

ABSTRACT

Mobile Ad-hoc Networks (MANETs) are a set of dynamic wireless mobile nodes that perform without any centralized control or fixed infrastructure. Each mobile node in a MANET moves unpredictably, which causes rapid changes in the network topology. Frequent changes in the topology affect network performance, resulting in continuous disconnection between communication nodes. The proposed algorithms Slow Down Speed (SDS) and Ant System Node Control (ASNC) aim to control the movement of the mobile nodes based on awareness of traffic conditions. SDS is a simple self-organization algorithm, while ASNC is a complex self-organization algorithm adapted from ant swarming behavior in nature. The new proposed algorithms are simulated using Network Simulator 2 over Ad-hoc On-Demand Distance Vector (AODV) and Destination Sequenced Distance-Vector (DSDV) routing protocols. The results of SDS and ASNC show significant improvement in the performance of a MANET. In general, the reactive routing protocol, AODV, shows the best results for both SDS and ASNC.

DEDICATION

To my mother, father, husband, and daughter - I would not be who I am today without your constant love and support. I am sincerely thankful for the encouragement and help you have given me throughout my life to succeed and make my dreams come true. I have been blessed to have a family such as you.

ACKNOWLEDGEMENTS

In the name of Allah, the most gracious, the most merciful, all praise is due to Allah, Lord of the Worlds: a praise reaching his bounties and matching his provision. Blessings and peace of Allah be upon his servant and messenger Muhammad, our master and guardian the seal of the prophets and the leader of the messengers, and upon his entire family and companions just like the blessings and peace of Allah upon Ibrahim and his entire family and companions.

All praise is due to Allah for enabling me to complete my master's degree; may Allah teach me from that which benefits me, and benefit me with that which he has taught me, and increase my knowledge and accepted deeds.

Thanks go to King Abdullah bin Abdulaziz, may mercy of Allah be upon him, King of Saudi Arabia and custodian of the two Holy Mosques for giving me and the other Saudi students the opportunity to continue our studies aboard all over the world and providing us with full funds through his scholarship program.

I would like to thank the thanks also for the Ministry of Education in Saudi Arabia and the Saudi Arabian Culture Bureau in Canada for the support and help they have provided during my residency and study in Canada.

I would like to give thanks and my sincere gratitude to my supervisors Prof. Przemyslaw Pochec and Prof. John DeDourek for their continuing support throughout my thesis study, for their patience, motivations, and immense knowledge. Without their guidance and consistent help, this thesis would not have been completed.

Thanks also go to everyone who helped and guided me with kindness during my residency in Fredericton and during my study in the University of New Brunswick especially in the Department of Computer Science.

Table of Contents

List of Figures

List of Tables

Abbreviations

ABR	Associativity Based Routing
ALOHA	Areal Locations of Hazardous Atmospheres
ANSI	Ad-hoc Networking with Swarm Intelligence
AODV	Ad-hoc On-Demand Distance Vector
ARA	Ant-colony-based Routing Algorithm
ACO	Ant Colony Optimization algorithm
AS	Ant System
ASNC	Ant System Node Control
ABC	Artificial Bee Colony
AI	Artificial Intelligence
AWK	Aho, Weinberger and Kernighan
BCO	Bee Colony Optimization
BWAS	Best-Worst Ant System
CBR	Cluster Based Routing
CBR	Constant Bit Rate
CGSR	Cluster head Gateway Switch Routing
DSDV	Destination Sequenced Distance-Vector
DV	Distance Vector
DSR	Dynamic Source Routing
EAS	Elitist Ant System
FPTD	Forward Path Time Detection
FSR	Fisheye State Routing
FTP	File Transfer Protocol
GSR	Global State Routing
HSR	Hierarchical State Routing
LAN	Local Area Network
LAR	Location Aided Routing
MMAS	Max-Min Ant System
MANET	Mobile Ad-hoc Network

NAM	Network AniMator
OSPF	Open Shortest Path First
OTCL	Object Oriented Tool Command Language
PDR	Packet Delivery Ratio
PAR	Power-Aware Routing
PSO	Particle Swarm Optimization
PRNET	Packet Radio Network
AS_{rank}	Rank-Based Ant System
RERR	Route error
RREP	Route replay
RREQ	Route request
RIP	Routing Information Protocol
RWP	Random WayPoint model
SARA	Simple Ant Routing Algorithm
SDS	Slow Down Speed
SSR	Signal Stability Routing
STAR	Source Tree Adaptive Routing
SURAN	Survivable Adaptive Radio Networks
SI	Swarm Intelligence
TCP	Transmission Control Protocol
NS2	Network Simulator 2
Tcl	Tool Command Language
TORA	Temporally-Ordered Routing Algorithm
TSP	Travelling Salesman Problem
UDP	User Datagram Protocol
UNB	University of New Brunswick
WRP	Wireless Routing Protocol
ZHLS	Zone based Hierarchical Link State
ZRP	Zone Routing Protocol

Chapter 1

Introduction

1.1 Background

A mobile ad-hoc network (MANET) is a wireless network where mobile nodes move unpredictably causing rapid changes in the network topology [1] [2]. The new proposed algorithms aim to control the movements of mobile nodes in a MANET in order to establish connections between the communication nodes and improve the network performance. Controlling the movement of the mobile nodes to move in an area between the communication nodes instead of moving randomly creates a transmitting channel that forwards data packets through the network. This channel made of forwarding nodes is called a mobile medium where the performance of a MANET depends on how long the mobile medium remains in place and how many packets it can carry [3]. The connection between the other mobile nodes in a MANET is not important at this point. This thesis only considers provision of a full connection between the communication nodes by controlling the movement of the mobile nodes in a MANET.

1.2 Problem Description

High dynamic-node mobility is a significant problem experienced in Mobile Ad-hoc Networks (MANETs) [4]. Wireless mobile nodes in a MANET communicate with each other, by forwarding data, without infrastructure or centralized control. Each mobile node moves in random directions and at random speeds, which causes a rapid disconnection between the communicating mobile nodes and changes in the network topology [2] [4]. The highly dynamic mobility of the nodes also affects routing tables and causes continuous

loss or changes in the forwarding paths that are used to transmit the data packets through the network [4]. Furthermore, losing packets or not transmitting them at all, because of constant mobile node movements, significantly decreases the network performance in a MANET. These many issues in a MANET make it an interesting research area for investigation.

1.3 Objectives

To address the issue of the unreliability of the movement of mobile nodes, this master's thesis aims to control the mobile node movements in a MANET by adapting a Swarm Intelligence algorithm inspired by the behavior of insects in nature. The thesis proposes dynamic-movement node-control algorithms that control the mobile node movements based on awareness of the traffic condition of the network. The information about the traffic condition illustrated in accessing the forwarding paths in the routing protocol makes it possible to know which mobile nodes are active forwarding nodes and which are not.

The first of two proposed algorithms, Slow Down Speed (SDS), is a self-organization algorithm where the mobile node movements are controlled based on knowing simple information about the traffic condition. The other algorithm, Swarm Intelligence, is a complex self-organization algorithm called Ant System Node Control (ASNC). ASNC is adapted from an Ant Colony Optimization algorithm (ACO) in order to control the mobile node movements in a MANET with an awareness of the network traffic condition. Two different MANET routing protocols have been used in the experiments for comparison purposes; one of them is Ad-hoc On-Demand Distance Vector (AODV) and the other is Destination Sequenced Distance-Vector (DSDV). The experiments have been done

through simulation to control the mobile node movements in a MANET and the results used to measure the network performance over different performance metrics.

1.4 Related Works

Many studies have been proposed to implement new routing protocols for a MANET, adapting different Swarm Intelligence algorithms including: Ant-colony-based Routing Algorithm (ARA) [5], Simple Ant Routing Algorithm (SARA) [6], Ad-hoc Networking with Swarm Intelligence (ANSI) [7], AntHocNet hybrid algorithm [8], AntNet routing algorithm [9], and HOPNET routing algorithm [10]. However, few researchers have focused on the problem of controlling mobile node movement in a MANET and the existing work in this area is still in early stages.

At the University of New Brunswick (UNB) different related work studies have been done in the area of controlling the dynamic movement of wireless mobile nodes in a MANET. One of the studies on MANETs "The Self-organizing Mobile Medium Ad Hoc Network" outlines a scheme for controlling the movement of mobile nodes in a M2ANET based on an attraction/repulsion paradigm [11]. Another study "The Advantage of Moving Nodes in Formation in MANETs and M2ANETs" describes a novel group mobility model, a towing formation and shows its advantage in controlling the movement of mobile nodes in MANET [12]. Another existing study is the "Linear Node Movement Patterns in MANETs", which investigated the impact of four different linear movement patterns similar to those observed in some man-made objects [13].

This master's thesis proposes two different dynamic-movement node-control algorithms in a MANET. The proposed algorithms control the mobile nodes based on awareness of the routing protocol information about the traffic condition in the network. The first algorithm

is a self-organization algorithm while the other one is an Ant Colony Optimization algorithm. These algorithms were implemented and tested in a network simulator using two different routing protocols, AODV and DSDV.

1.5 Thesis Organization

This master's thesis consists of nine chapters. Chapter 1 is an introduction that includes the background of MANETs in addition to the description of the problem, followed by the objective of this work, related work and the organization of the thesis. Chapter 2 explains the concept of MANET in general terms including the history, applications, characteristics, and mobility models. Chapter 3 gives brief information about the routing protocol categories in MANETs, then focuses on the utilized AODV and DSDV routing protocols. Chapter 4 illustrates Swarm Intelligence in nature and the different existing Swarm Intelligence algorithms, such as the Ant Colony Optimization (ACO) and the Bee Colony Optimization (BCO). Chapter 5 introduces the Network Simulator NS2 and the modifications that have been done within it, along with the simulation environment and the performance metrics. Chapter 6 describes the proposed self-organization algorithm, called Slow Down Speed algorithm (SDS), as well as the results and analysis of experimentation. Chapter 7 describes the proposed Swarm Intelligence algorithm, called Ant System Node Control (ASNC), in addition to the results and analysis of the experiments conducted. Chapter 8 compares the performance of SDS and ASNC from different aspects including PDR, distance covered and active forwarding nodes in addition to visualization maps that show the position and amount of pheromones. Chapter 9 concludes the thesis and provides suggestions for future work.

Chapter 2

Mobile Ad-Hoc Networks

2.1 Mobile Ad-Hoc Networks

A Mobile Ad-hoc Network (MANET) is an IEEE 802.11 framework that can be defined as a set of self-organizing wireless nodes (laptops, smart phones, sensors, etc) that communicate randomly with each other to provide a network with no fixed infrastructure or a central node to control [1] [2]. A MANET is a stand-alone network that is attached to the Internet or a cellular network via one or multiple points. In a MANET nodes move randomly and are free to join or leave the network at any time, which leads to unpredictable topology that changes rapidly [4]. Nodes in a MANET are able to detect the presence or absence of other nodes in the network and the network performance is based on communication and sharing of packets. They use several techniques to become aware of neighboring nodes, one of which is broadcasting "Hello" messages [14]. These types of networks are capable of being deployed with limited nodes and are suitable for implementation when a fixed infrastructure is impossible or too expensive.

Mobile ad-hoc networks provide a supporting environment for the users to access information and communicate anywhere using any device at any time. MANETs are fully symmetric environments with all nodes having the same capabilities and responsibilities [15]. On the other hand, nodes are asymmetric when it comes to transmission and radio ranges, battery life, movement speed, and processing capacity [15]. When two nodes are in radio range a connection is established between them in a peer-to-peer form. MANETs are also known as multi-hop networks, which take several wireless hops to deliver a packet

to its destination [4]. Nodes of this network function as routers, or end-user devices; they discover and maintain routes to other nodes and contain one or multiple hosts during transmission. Nodes forward packets via other nodes, which are not in the packet's source transmission range to reach the packet's destination.

Considering the topology of a MANET, keeping a node connecting to another and finding the path through the network are really challenging problems. The forwarding path between the source node and the destination node in MANETs is usually the optimal path, but it is difficult to make the nodes follow this optimal path in such networks. For this purpose a number of algorithms, known as Swarm Intelligence Algorithms inspired from insect behavior such as ants in a colony, bees in a colony and fireflies, were developed. Insects in nature group together making swarms to find food, shortest paths to sources of food, or new locations in a primitive way; these simple intelligent methods can be applied to control the movement of random nodes in MANETs.

2.2 History of MANETs

In the early 1970's the concept of mobile ad-hoc networks started as a military application, which is useful under disorganized or hostile environments. The advantages of the network include flexibility, mobility, self-creation, and self-organization. The first generation of MANET, called PRNET (Packet Radio Network), was used on trails to improve network capabilities in battlefields in combination with ALOHA (Areal Locations of Hazardous Atmospheres). In the 1980s, a project called SURAN (Survivable Adaptive Radio Networks) aimed to enhance the ad hoc network scalability and survivability in the battlefield by providing small devices with low cost and low power [16] [17] [18].

The next generation came up with notebook computers in the 1990s, which used commercial ad hoc networks as a type of mobile node. In addition several ideas about a collection of mobile nodes were proposed during conferences. Finally, the IEEE 802.11 adopted the ad hoc networks and applications in different areas that were then implemented by the research community [18].

Because of the low cost, small size and powerful devices, MANETs grew fast. Soon, the great potential of mobile ad-hoc networks became widely known as a research area outside the military field, although it is still recent technology in its early stage that needs more attention to be fully understood.

2.3 MANET Applications

Despite all the difficulties, MANETs attract several real world applications that are increasing with evaluation of wireless technology and portable devices. The applications vary from large-scale mobility network applications to small and limited power network applications. Considering the architecture of MANETs, such as lack of infrastructure, mobility and multi-hop routing, MANETs can be applied almost anywhere, which helps MANET applications to become more popular.

Typical present and future applications for mobile ad-hoc networks and their services can be found on the battlefield. Military equipment uses MANET applications to help maintain information sharing between soldiers for tactical operations, between vehicles, and with military information headquarters. These airplanes, tanks and warships move rapidly at high speed, transferring critical information that need a quick, efficient, reliable, and secure communication network like a MANET. More applications for MANETs have been designed for industrial and commercial use, as well as for emergency services and disaster

relief efforts where MANET applications are the only possible solution. Examples include search and rescue operations, policing and firefighting, earthquake or flood situations, and support for doctors and nurses in hospitals [4] [19].

Other MANET applications for distributed computing can be found in the field of education to share information, e.g. university and campus settings, conference rooms and facilities, and virtual classrooms. Even in home and office networks, MANETs have found their way through to provide entertainment applications, for instance multi-user games, outdoor Internet access, wireless P2P networking, and Robotic pets. The applications of MANET services extend to cover more important areas, such as sensor network applications and context-aware services, to provide services in call-forwarding, mobile workspace, location-specific services, time-dependent services, and tourist information.

2.4 MANET Characteristics

The following are the most important characteristics of MANETs:

- **Scalability**

MANETs have limitations in power, bandwidth and routing, which cause scale limitation at implementation. Also, the mobility of nodes regularly causes changing in the scale of MANETs as they rapidly join or leave the network range. This mobility makes it hard to predict the exact scale of a MANET or how many nodes there are in the network, which may range from a few nodes to hundreds or even thousands of nodes. On the other hand, the scale of wired networks does not change over usage as it is predefined during the design. Still, the scope of MANETs can cover sensors in small areas of a few meters or extend to cover WLAN areas in classrooms, offices, and conference rooms up to hundreds of meters, as in military applications [17] [18] [19]

- **Control**

As there is no infrastructure, MANETs have no base stations to control or switches to process the power, unlike wired networks. Therefore, MANETs are defined as a distributed control network where each node makes its own decision for the benefit of the entire network [17] [18] [19].

- **Mobility and Dynamic Topology**

MANET nodes are free to join or leave the network at any time by moving at random with different speeds. This mobility leads to rapid unpredictable changes in the network topology. Also, transmission power, reception parameters and structure all impact the topology of MANETs [15].

- **Routing Complexity**

The complexity of routing increases in MANETs due to the dynamic topology, which causes constant changing in the routing tables or the source routes since every node in a MANET is a mobile router. Routing packets between two nodes is a complex task because the route may contain multiple hops. In addition, it is better to use reactive routing protocols instead of proactive protocols in MANETs. Another challenge in MANETs is multicast routing; considering the random movement of nodes the multicast tree is no longer static, whereas in wired networks the routing complexity comes from IP address changing, subnets, link status, and control congestion [15] [18] [19].

- **Bandwidth and Link Capacity**

MANETs and other wireless networks have significantly less capacity than wired networks due to spectrum constraints and the nature of the medium. The changing environment of

MANETs produces variable bandwidth delays in links that can cause an increase in congestion [18] [19].

- **Coverage**

MANETs, like other wireless networks, do not require cables; they can be built rapidly because there is no wired infrastructure as they are wireless networks - this is one of the biggest advantages of MANETs. The coverage is expanded dynamically by adding more nodes to the network. This flexibility in MANETs comes with disadvantages and costs; for example the coverage of MANETs is inconsistent and affected by the medium, physical layer and environmental conditions [18].

- **Energy Constraints**

Power efficiency is a primary concern in distributed control networks like MANETs, considering the difficulties of adding tasks required by these types of networks. Nodes in MANETs are often powered with batteries while in wired networks nodes get the power from electricity, which means the power supply is almost infinite and power is not considered a problem. Nodes that derive their power from batteries face several problems, and every effort such as transmission and processing must be minimized to save power [15] [18] [20].

- **Reliability**

The reliability of MANETs suffers from several effects, much as any other wireless network. These effects can be topology changes, variable link capacity, power and energy issues. MANETs have to deal with previous issues plus control rapid random movement of nodes to enhance reliability in the delivery of packets and avoid packet loss or data

transmission errors. The same thing applies to routing protocols, which must also deal with these issues to preserve reliable delivery of packets in MANETs [15] [18] [19].

- **Security**

MANETs are a wireless medium; they cannot be easily secured by physical means from security risks such as threats to confidentiality, unauthorized data manipulation or denial of service attacks [4] [17]. Physical security measures are designed to secure networks in general and wired networks in particular. Furthermore, the absence of a centralized server to manage and monitor the traffic in a dynamic large scale MANET makes a security attack difficult to detect. Therefore, MANETs are exposed to many more security threats, and the available security solutions for wired networks cannot be applied directly on MANETs. However, both wired networks and wireless networks face security issues that are possible to avoid [15] [18] [19].

2.5 Mobility Models

Mobility Models are designed to simulate the movement of mobile nodes in a simulated ad-hoc network. Simulating the locations, velocities and movement directions of the mobile nodes gives researchers the ability to test new algorithms and methods in order to improve the network performance, and determine if it is useful to implement or not, e.g. testing a new routing protocol. Controlling the movement of the mobile nodes in a MANET is a possible research area where the mobility models can be used to mimic the movement of real mobile nodes.

The Mobility Models utilized in the network simulation are divided into two types, Trace Models that are used in real life systems to observe mobility patterns over a long period, and Synthetic Models which are aimed to simulate the behavior of mobile nodes with

realistic characteristics in networks that are difficult to implement on a wide scale, such as MANET. Unlike the Trace Models, the Synthetic Models do not need to create a trace to model the mobile nodes [21]. My proposed dynamic-movement node-control algorithms have been simulated using the Network Simulator NS2, which utilizes a common random Synthetic Model, called Random Waypoint Model, to mimic the locations, speeds and directions of the mobile nodes in MANET. The Random Waypoint Model is preferred for simulating MANETs over other Mobility Models, due to its simplistic features, although sometimes MANETs may use more complex Mobility Models. Examples of other available Mobility Models are:

- Gauss-Markov Model
- Smooth Random Mobility Model
- A Boundless Simulation Area Model
- City Section Model
- Reference Point Group Model
- Markovian Model
- Column Mobility Model
- Nomadic Community Model
- Pursue Model

The Random Waypoint Model used to model the movement of the mobile nodes in order to control them is illustrated in more detail in Section 2.5.1.1.

2.5.1 Random Based Mobility Models

Random Based Mobility Models give the mobile nodes freedom to move randomly without any conditions. Each mobile node is chosen individually with a random position, speed and

movement destination, which makes them perfect models for the purpose of experimental study [22].

The commonly used Random Waypoint Model is discussed in Section 2.5.1.1, and the two types of the Random Model, the Random Walk Model and the Random Direction Model are discussed in Section 2.5.1.2 and Section 2.5.1.3 respectively.

2.5.1.1 Random Waypoint Model

In 1996, D. Maltz and D. Johnson presented the Random Waypoint Model (RWP) [23] as a simple, realistic wireless transmission-channel model added to the NS2 network simulator. The model was designed to provide an accurate simulation of the MAC and physical layer of the IEEE 802.11 standard. During their experiments, Maltz and Johnson compared four wireless ad-hoc network routing protocols, Ad Hoc On-Demand Distance Vector Routing (AODV), Dynamic Source Routing (DSR), Destination Sequenced Distance-Vector (DSDV), and Temporally-Ordered Routing Algorithm (TORA) utilized RWP model in a network with 50 mobile nodes. As a result of the experiments, AODV and DSR showed the best performance, followed by DSDV, while TORA had the worst performance [24].

In the NS2 simulator, different mobility scenarios can be generated using the Random Waypoint model. The mobile nodes are distributed randomly over the simulation area, then each mobile node selects a destination position randomly to move toward, along with a constant velocity that is uniformly distributed between [V_{min}, V_{max}], where V_{min} is the minimum velocity and V_{max} is the maximum velocity. The velocity and destination for a mobile node are totally independent and not related to the other nodes in any way. When a node reaches its destination it stops for a specific period of time, called "pause time"

defined by T_{pause}, if the T_{pause} is equal to 0 then the node continues moving without any pause time to the next random destination. This process is repeated until the run time of the simulation ends [25].

Based on the changes in the velocity and the pause time parameters, the ad-hoc network topology may be relatively stable or highly dynamic. The NS2 simulator uses the Setdest tool from the Monarch Group Model to generate a trace for the mobile node in the Random Waypoint Model.

2.5.1.2 Random Walk Model

The Random Walk Model, also known as Brownian Motion, is a special condition of the Random Waypoint Model with zero pause time [22]. In the Random Walk Model, the mobile nodes are independent and random, just like the Random Waypoint Model except that the node velocity and destination change for each time interval *t*. During the simulation run, the mobile nodes follow the same random process of the Random Waypoint Model [26]. The previous destination and velocity information for mobile nodes are not used for future decisions.

2.5.1.3 Random Direct Model

The Random Direct Model is another special condition of the Random Waypoint Model presented by Royer, Melliar Smith and Moser (2001) [27]. In the Random Direct Model, the mobile node pauses for a specific period of time T_{pause} when it reaches the boundary of the simulation area. After the boundary is reached, the mobile node changes its direction, between 0 and 180 degrees, and then continues moving until it reaches the boundary again [22]. The whole process is repeated until the end of the simulation run time.

In the Random Waypoint Model, the mobile nodes tend to move through the center of the simulation area toward the destination, and cluster near the center, rather than moving toward the boundaries. In the Random Direct Model, each mobile node chooses a random direction to move along until it reaches the boundary [25].

Chapter 3

Mobile Ad-Hoc Routing Protocols

3.1 Routing in Mobile Ad-Hoc Networks

Routing protocols in MANETs have to deal with the random movement of nodes, power limitations, low bandwidth, and high error rates. Considering the mobility of MANETs, the nodes have to change their routing table during their movement at high speed or it will become very complex to transmit a packet to its destination [15] [18] [19]. Generally, routing protocols are categorized into Proactive (Table-Driven), Reactive (On-Demand Driven) and Hybrid Protocols [28]. All the routing protocol categories have been improved to support a higher quality of service, provide more security solutions and spread over larger areas (more scalability). Each category is illustrated in more detail in Sections 3.1.1, 3.1.2 and 3.1.3.

3.1.1 Proactive Routing Protocols

Proactive routing protocols, also known as table-driven protocols, are the same as Internet routing protocols, e.g. Routing Information Protocol (RIP), Distance Vector (DV) and Open Shortest Path First (OSPF) [24]. These protocols maintain and update routing information for the whole network. They send control packets to update the routing table, but the congestion increases as the network becomes more dynamic because of these control packets. Nodes in proactive routing are responsible for taking charge of information storing, broadcasting and propagating in one or more routing tables and tracking the changes in the network topology. Routes of proactive protocols are always available with

full use of traffic signals and power even if there is no need for it [28] [29]. Examples of the proactive protocols in MANET ordered by the date of implementation include:

- DSDV- Destination Sequenced Distance-Vector, 1994
- WRP- Wireless Routing Protocol, 1996
- CGSR- Cluster head Gateway Switch Routing, 1997
- GSR- Global State Routing, 1998
- FSR- Fisheye State Routing, 1999
- HSR- Hierarchical State Routing, 1999
- ZHLS- Zone based Hierarchical Link State, 1999
- STAR- Source Tree Adaptive Routing, 2000

The experiments for the dynamic-movement node-control utilized two types of routing protocols, one of which is the proactive routing protocol DSDV. The DSDV was chosen due to the great potential it has as one of the best existing routing protocols for MANETs. Also, because of the limitation in the available MANET routing protocols in the Network Simulator NS2, which has been used to simulate the presented dynamic-movement node-control algorithms. As a result, only the DSDV routing protocol from the proactive category was used in these experiments, and the other protocols are beyond the scope of this thesis.

3.1.1.1 DSDV Routing Protocol

Destination Sequenced Distance Vector (DSDV) routing protocol is a proactive protocol for MANET, developed in 1994 by C. Perkins and P. Bhagwat [30]. DSDV adapted the Routing Information Protocol (RIP) or Distributed Bellman Ford algorithm to prevent routing loops by limiting the number of hops and finding the shortest routes between the

source and the destination. DSDV added a sequence number generated by the destination node to the routes; if the sequence number is even, then the route is still available and good to use. On the other hand, if the sequence number is odd that mean the routes will no longer be available [31] [32]. This routing protocol is suitable for dynamic mobile nodes in networks with no infrastructure, such as MANETs.

Each node in DSDV has its own routing table, which contains the destination, the metric (hops count), the next hop, and the sequence number [33]. Each node continuously maintains route information in its own routing table. The packets are transmitted between two nodes through the route with the latest sequence number, and the other routes are discarded. If two routes have the same sequence number in the routing table, then the packets will be transmitted through the shortest route, which has the smallest metric number [32] [34].

The routes between the mobile nodes need to be updated rapidly because of changes in the topology of the MANET. To update the routing tables each node advertises its own routing table to the neighboring nodes when routes are broken or when new routes have been found. Updating the routes may be delayed until the best routes are found, to reduce the number of duplicated route sequence numbers in the routing table. Advertising the routes causes a delay in time and waste of bandwidth [33] [34]. This delay causes DSDV to take a long time to detect the first forwarding path in the network compared with other routing protocols, such as AODV. This leads to a decrease in packet delivery ratio and network performance in general.

3.1.2 Reactive Routing Protocols

Reactive protocols in MANET, also known as on-demand routing protocols, attempt to reduce the constant need for updating the routing tables that track the topology changes in the network by creating routes only when required [28] [29]. A route-by-route discovery procedure is established when a source ($node_x$) decides to send a packet to a destination ($node_y$). During the packet transmission, a route maintenance procedure is set up to maintain the route until it is no longer desired or it becomes inaccessible. Finally, a route deletion procedure is performed to destroy that route [35]. In reactive protocols, the route discovery procedure causes long delays in transmission, but they are more efficient at signaling and consuming power. Some of the existing reactive routing protocols, in date order, are:

- DSR- Dynamic Source Routing, 1996
- ABR- Associativity Based Routing, 1996
- TORA- Temporally-Ordered Routing Algorithm, 1997
- SSR- Signal Stability Routing, 1997
- PAR- Power-Aware Routing, 1998
- LAR- Location Aided Routing, 1998
- CBR- Cluster Based Routing, 1999
- AODV- Ad Hoc On-Demand Distance Vector Routing, 1999

The performed dynamic-movement node-control experiments utilized one of these reactive routing protocols, AODV. The AODV routing protocol demonstrates high performance in MANETs compared with other protocols. The reasons behind using AODV instead of another reactive routing protocol are the same reasons mentioned in Section 3.1.1.

3.1.2.1 AODV Routing Protocol

Ad-hoc On-Demand Distance Vector Routing (AODV) is an on-demand routing protocol that finds the route only when a source node requires a route for packet transmission [36]. The dynamic nature of AODV makes it suitable for networks that do not have a fixed infrastructure such as MANET. AODV uses the sequence number for the route in the routing tables to update the routes and avoid the problem of routing loops. The routing table contains information about the destination, the next hop, hop counts, the sequence number, and the route expiry time [29] [33] [37].

The source node floods the network with a route request (RREQ) packet, called broadcasting, to establish a connection. A temporary route in the reverse path will be created by the RREQ for each passing node until it reaches the destination node. When RREQ finds a route leading eventually to the destination node, a route replay (RREP) packet is unicast backward to the source node via the same temporary route that is used to transmit the RREQ packet. The chosen route is the route with the minimum number of hops to transmit the data packet through from the source node to the destination. If a path between two nodes breaks, both nodes inform their end nodes about the broken link by sending route error (RERR) packets. Then, the end nodes delete the corresponding entries from their table. Each RREQ has a sequence number to avoid repeating RREQs through the nodes they have already passed. Also, the RREQ has a limited number of retransmissions, known as “time to live” [28].

AODV routing protocol is simple, with no extra traffic for communication; it does not require complex calculations or large storage capacity. Also, it has an efficient technique to establish and maintain routes with minimum time delay [35]. In a large dense network,

e.g. 40 mobile nodes, AODV is able to detect the first forwarding path from the first second of run time. Because of this, a network that uses the AODV protocol performs much better than other networks using other protocols such as DSDV and DSR [29] [33].

3.1.3 Hybrid Routing Protocols

Hybrid routing protocols collect sets of nodes in zones inside the network topology then partition the whole network into zones. Each zone uses a proactive approach to maintain routing information and a reactive approach to route packets between different zones. If the route to a required destination node is in the same zone the route will be established without delay, but if the destinations are in other zones then route discovery and maintenance are needed [28] [35]. There are two available protocols of hybrid routing protocols:

- ZRP- Zone Routing Protocol
- ZHLS- Zone-Based Hierarchical Link State

The hybrid approach is suitable for routing in large networks because these types of protocols provide a better balance between communication overhead and delay based on the dynamics and the size of the zone [28].

Chapter 4

Swarm Intelligence

4.1 Self-Organization

W. Ross Ashby formulated the original principle of the system of self-organization in 1947, although it was not published until 1962. In his paper, Ashby explained the fundamental concept of organization, and the integration of machines, and how they lead to what he referred to as a system of self-organization [38]. Later, in 1999, Bonabeau et al. defined the concept of self-organization relating to "Swarm Intelligence". Self-organization can be defined as a set of dynamic spontaneous global structures that appear out of the local interactions between lower-level components of an initially disordered system [39] [40]. Spontaneous means that the process is not controlled by any agent inside or outside the system. Based on purely local information, an agent chooses which rules the process and its initial conditions should follow. This is distributed across all the agents in the system which follow the same process in parallel. As a result of this, the organization is very robust, and self-repairs any damage or perturbations. Any elimination or replacement of individual agents will not affect the system. In such a complex system, the agents directly interact locally with other agents near them, by visual contact or chemical contact, and affect the remote agents indirectly by changing the environment around them. The agents are goal-directed and select one outcome over another during the evolution of the system [40] [41]. Self-organization can be found in many different areas of science, e.g. physics, chemistry, biology, and telecommunication. Swarming in groups of animals, flocks of

birds, schools of fish, and colonies of insects are common examples of self-organization in nature.

Implementing a swarm system with self-organization features in wireless networks is a real challenge, but not impossible. Bonabeau et al. (1999) present self-organization in swarms through positive feedback, negative feedback, fluctuations and multiple interactions. Positive feedback as a simple behavior may be presented as a recruitment and reinforcement [39]. For example, the dancing of bees, or laying of pheromone and following these trails of ants, are kinds of recruitment for their fellow species to reach a food source. On the other hand, negative feedback represents a way of balancing positive feedback and stabilizing the collective pattern. Fluctuations, such as random searches and errors, are important for self-organization in swarms in order to create creativity, innovation and discover new solutions. Finally, multiple interactions appear from sharing and spreading information between agents in the swarm [39] [41].

Individuals in swarms behave in a simple way that collectively create more complex behaviors. They interact locally with each other and with the environment around them to create a self-organization system able to solve complex problems [40] [41].

4.2 Swarm Intelligence

Swarm Intelligence (SI) is an artificial-intelligence approach to problem-solving using algorithms based on the self-organized collective behavior of social insects such as ants, bees, birds, fish, wasps, and termites that follow very basic rules [39]. Beni and Wang introduced the expression "Swarm Intelligence" in 1989 in the context of cellular robotic systems [42]. Scientists and engineers have studied this field of swarm behavior for years to analyze the intelligence behind how insects achieve a goal, and the interest continues to

grow. The individual insects are not intelligent but with cooperation they make smart colonies able to solve complex tasks. Insects communicate with each other directly, or indirectly through changes in their environment, forwarding information about locations, paths, or food. Sharing these experiences about the best places they found and the shortest paths they can take makes a perfect SI system of simple agents. Natural swarm systems made up of millions of flexible individual agents following simple rules demonstrate complicated group behavior. Such systems can be found in ant and bee colonies, flocks of birds, and schools of fish [43] [44].

Swarm intelligence solves problems within complex systems made up of many agents that are capable of cooperation. The failure in one of the agents affects neither the SI system performance nor the number of the agents. SI based on the nature of insects provides a basis on which it is possible to explore collective (or distributed) problem-solving methods without centralized control or the provision of global models such as MANETs [44] [45] [46]. Different Swarm Intelligence algorithms are applied to optimally solve problems in the real world, i.e. Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC).

Swarm Intelligence algorithms can be used in MANET to control the movement of the nodes in order to follow an optimal forwarding path during the network run time. Nodes using an SI algorithm have the same ability to connect as other nodes, except that they move dynamically and are called agents. The agents follow very simple rules, although there is no centralized control structure dictating how individual agents should behave. These interactions between agents lead to "intelligent" global behavior, unrecognized by the individual agents [47] [48] [49].

4.2.1 Particle Swarm Optimization

The Particle Swarm Optimization (PSO) algorithm was introduced to solve nonlinear function optimization based on swarm intelligence by Kennedy, Eberhart and Shi in 1995 [50]. Particle Swarm Optimization is an Artificial Intelligence (AI) technique related to both genetic algorithms and evolutionary programming to model group movement behavior inspired by flocks of birds and schools of fish. An example is birds flying randomly searching for food in different places, until one of the birds is close to the food source and can actually smell it. The birds then know which bird is nearest to the food and they also know how far away the food is. At this point, the bird shares the information about the position of the food with the other birds in the flock. Transference of the correct information will enable the flock to follow the bird closest to the food to reach the food [51] [52]. In this example, the flock represents the swarm solution and the food source represents the optimal solution.

The Particle Swarm Optimization algorithm represents the swarm as a population, while the "n" particles represent the candidate solution. The population following a few simple mathematical formulas to move around in a D-dimensional search space is represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. Each particle position represents a possible solution and each particle with a weight value has its own velocity representing a direction to the next position. The i-th particle is represented as $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, while the velocity of i-th particle is also represented in a vector as $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The particles move according to the local best-known position and toward a better best-known position that other particles found in the search space. Eventually, the whole swarm moves toward the best position until a solution is found, although it is not guaranteed to be the optimal

solution [51] [52]. The new velocity, and the new position of the i-th particle added to its new velocity, are respectively determined according to the following expressions:

$$V_i^{k+1} = \chi\left(\omega V_i^k + c_1 r_{i1}^k\left(P_i^k - X_i^k\right) + c_2 r_{i2}^k\left(P_g^k - X_i^k\right)\right) \quad (4.1)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (4.2)$$

$i = 1, 2, \ldots, N$ - N the population size

X_i^k- the position of particle i at time k

P_i^k- the best position particle i has found up to time k

P_g^k- the best position any particle (global) has found up to time k

χ- constriction factor to control velocities

ω- inertia weight

c_1- positive constant, called cognitive parameter

c_2- positive constant, called social parameter

r_{i1}& r_{i2}- uniformly distributed random numbers between [0,1]

The value of the constriction factor χ is given in [51] [52] as:

$$\chi = \frac{2}{|2-\varphi-\sqrt{\varphi^2-4\varphi}|} \quad (4.3)$$

$$\varphi = c_1 + c_2, \varphi > 4 \quad (4.4)$$

The concept of Particle Swarm Optimization (PSO) is to take advantage of the collective memory of the entire swarm along with the individual memories. The swarm learns from the performance and experience of individuals in the swarm, and the individuals in the swarm learn from their own and each other's experiences. PSOs can search very large spaces for candidate solutions and use these on optimization problems that are partially irregular, noisy and change over time. Also, it is a fast algorithm and requires less memory

compared with other optimization algorithms, and is easy to implement [52]. A PSO algorithm can be used in many different areas such as neutral network training, machine study, model classification and function optimization.

4.2.2 Ant Colony Optimization

Ant Colony Optimization (ACO) is a type of optimization algorithm modeled on the actions of an ant colony using a swarm intelligence method. In 1996 Marco Dorigo, Vittorio Maniezzo and Alberto Colorni implemented the Ant Colony Optimization based on their observation of ant behavior in nature [53]. ACO methods are useful in problems that need to find paths to goals. The first ACO algorithms aimed to solve the NP-complete Travelling Salesman Problem [TPS] with the goal of finding the shortest round trip between a series of cities [53] [54] [55]. In nature, ants without vision (almost blind) explore the environment around the colony searching randomly for a food source. When an ant finds food, it leaves a chemical pheromone trail on the path taken returning to the colony carrying that food. The other ants start following that pheromone trail to reach the food source [56] [57]. Pheromone is not permanent, it evaporates over time and distance. The pheromone evaporates from the longer paths faster than the shortest one and in this way ants always find the shortest path to the food source. Artificial 'ants' as simulation agents locate optimal solutions by moving through a parameter of space representing all possible solutions to solve a problem [58] [59] [60].

4.2.2.1 Ant System

Colorni, Dorigo and Maniezzo (1996) developed the first and simplest Ant Colony Optimization algorithm, called Ant System (AS) [53]. The Ant System algorithm was inspired by the parallel search over different paths for real ants in nature. Real ants take

several ways to solve their problems based on the data and information stored in their memory about previous obtained results [53] [61]. Ant System transition probability was modified to include heuristic information. Also, a tabu list was added to the Ant System as a memory capacity. In the Ant System a set of agents (Ants) follows local decisions to move through nodes (places) to solve a problem [62] [63]. The probability rule used by the ant to move from place i to place j during iteration is:

$$p_{ij}^{k} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{k \notin tabu_k} \tau_{ik}^{\alpha}\eta_{ik}^{\beta}} & if\ k \notin tabu_k \\ 0 & otherwise \end{cases} \quad (4.5)$$

τ_{ij}- the amount of pheromone trail

η_{ij}- the visibility $(0 \leq \eta_{ij} \leq 1)$

α- the impact of trail defined by the user $(0 \leq \alpha \leq 1)$

β- the attractiveness defined by the user $(0 \leq \beta \leq 1)$

$tabu_k$- tabu list contain the visited cities by ant_k during current iteration

The visibility value represents the heuristic information, which can be calculated by using the following expression:

$$\eta_{ij} = \frac{1}{d_{ij}} \quad (4.6)$$

d_{ij}- the distance between place i to place j

Each ant leaves a trail called pheromone τ after completing a solution to attract the other ants. This pheromone trail τ can be modified by other ants moving along the same path or it can just evaporate over time [63]. The pheromone τ_{ij} laid on the connected edge between place i and place j is updated by using the following equation:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(\mathrm{t}) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad (4.7)$$

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{4.8}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & if\ ant_k\ used\ \ path(i,j)\ in\ its\ tour \\ 0 & otherwise \end{cases} \tag{4.9}$$

ρ- the evaporation coefficient defined by the user $(0 \leq \rho \leq 1)$

m-the number of ants

$\Delta\tau_{ij}^{k}$- the pheromone quantity laid by ant-k on the path (i,j)

Q- constant, often 1

L_k-the tour length of the solution obtained by ant-k

1- Initialize the parameters value

- The pheromone trail τ_{ij}
- The visibility η_{ij}

2- For each ant_k in state i do //construction

Repeat

- Choose the next state j to move into //using the probability rule
- Add the chosen state j to the tabuk for the antk
- Until ant_k complete its solution

3- For each ant move through $path(i,j)$ do //pheromone update

- Compute the pheromone quantity $\Delta\tau_{ij}$
- Update the pheromone trail matrix

4- If not (end test) go to step 2 //terminating condition

Figure 4.1: Ant System Algorithm Pseudocode [53]

The ant agents use the pheromone trail information to control movement through the search operation to find the optimal solution for the problem [60] [61] [63]. The full pseudocode for the ant system algorithm is given in (Figure 4.1).

4.2.2.2 Rank-Based Ant System

The Rank-Based Ant System (AS_{Rank}) is an improvement of the Elitist Ant System (EAS) algorithm introduced by Bullnheimer in 1997 [64]. Each iteration in the Rank Based Ant System has a number of tours approximately equal to the number of ants illustrating the solutions found by these ants. These tours are ranked and have a weight factor ω usually equal to 25% of the number of ants based on previous experiments. The best tour has a weight factor of (ω-1), the second tour has a weight factor of (ω-2), the third ranked tour has a weight factor of (ω-3), etc. In the Rank-Based Ant System, pheromone is deposited only by the best ants that have a weight factor of (ω-1) and by one other elite ant [61] [64]. The amount of pheromone deposited depends on the tour rank and the quality of the solution. The pheromone update expression is given by:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sigma\Delta\tau_{ij}^{e} + \sum_{\mu=1}^{\sigma-1}(\sigma-\mu)\Delta\tau_{ij}^{\mu} \tag{4.10}$$

$\Delta\tau_{ij}^{e}$ and $\Delta\tau_{ij}^{\mu}$can be defined as follow:

$$\Delta\tau_{ij}^{e} = \begin{cases} Q/L^{gb} & if\ path(i,j) \in G_{best} \\ 0 & otherwise \end{cases} \tag{4.11}$$

e- the elitist ants

L^{gb}- the tour length of global-best solution

$$\Delta\tau_{ij}^{\mu} = \begin{cases} Q/L_{\mu} & if\ the\ \mu th\ best\ ant\ travels\ path(i,j) \\ 0 & otherwise \end{cases} \tag{4.12}$$

4.2.2.3 Max-Min Ant System

Thomas Stutzle and Holgar Hoos introduced the Max-Min Ant System (MMAS) in 1996 to improve the Ant System Algorithm [65]. One of the improvements was binding the pheromone value on the edges to avoid search stagnation (all ants follow the same path) [66]. The Max-Min Ant System has two pheromone bounds, upper bound $\tau_{max}$and lower bound τ_{min} to reach pheromone thresholds. At the beginning of the Max-Min Ant System, the upper bound (Max) pheromone value is initialized, to make the most advantage of the algorithm [66] [67]. The upper bound $\tau_{\max}(t)$ is given by using the following equation:

$$\tau_{\max}(t) = \frac{1}{(1-\rho)\,\mathrm{Cost}_{\mathrm{opt}}(t)} \tag{4.13}$$

$\mathrm{Cost}_{\mathrm{opt}}(t)$- the optimal solution value for a specific problem.

On the other hand, the lower bound $\tau_{\min}$ value is given by:

$$\tau_{\min}(t) = \frac{\tau_{\max}(t)\left(1-\sqrt[n]{P_{best}}\right)}{(\lambda-1)\sqrt[n]{P_{best}}} \tag{4.14}$$

$$\text{if } \begin{cases} P_{best} = 1 & \tau_{\min}(t) = 0 \\ P_{best} \approx 0 & \tau_{\min}(t) > \tau_{\max}(t) \end{cases} \tag{4.15}$$

P_{best}- the probability of creating the global-best solution defined by the user

n- the number of decision points

λ- the average number of edges at each decision point

The probability rule used by the Max-Min Ant System to move from place i to place j during iteration is the same rule used by the Ant System algorithm (Equations 4.5 and 4.6):

$$p_{ij}^{k} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{k \notin tabu_k}\tau_{ik}^{\alpha}\eta_{ik}^{\beta}} & if\ k \notin tabu_k \\ 0 & otherwise \end{cases}$$

$$\eta_{ij} = \frac{1}{d_{ij}}$$

d_{ij}- the distance between place i to place j

Another improvement identified how to update pheromone; this is the main difference between the Ant System and other Ant Colony Optimization algorithms [66] [68]. In Ant Colony Optimization each ant leaves a certain amount of pheromone trail on every edge it passes. However, to find the best solution in the Max-Min Ant System, the pheromone trail is updated and placed during iteration only by the best ant, the ant with best local solution [66].

1. Initialize the parameters value
    - The pheromone trail τ_{ij}
    - The heuristic value η_{ij}
2. For each ant_k do //main loop
    - Place ants on the starting node // randomly or in predefined order
    - Constructs solutions for each ant by adding nodes // using the probability rule
    - Boosting algorithm performance constructed a better solution than the initial solution // optional step
    - Find ant with the best solution
    - Update the pheromone trails by evaporate trails or reinforce trails
3. End for loop

Figure 4.2: MMAS Algorithm Pseudocode [65]

The pheromone update is implemented by using the following expression:

$$\tau_{ij}(t+1) = \left[(1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{best}\right]_{\tau_{min}}^{\tau_{max}} \quad (4.16)$$

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{L_{best}} & path(i,j) \in G_{best} \\ 0 & otherwise \end{cases} \quad (4.17)$$

L_{best} is the tour length found by the best ant, which can be equal to L_{ib}-length of iteration best- the best solution at the current iteration or L_{gb}-length of global best- the best found solution since the beginning of the algorithm [68] [68]. Max-Min Ant System one of the best Ant Colony Optimization algorithms so far. For full algorithm pseudocode check Figure 4.2.

4.2.2.4 Best-Worst Ant System

The Best-Worst Ant System (BWAS) algorithm was designed to improve the performance of the Ant Colony Optimization algorithm [70]. The Best-Worst Ant System uses the same transition probability rule as the Ant System algorithm (Equation 4.5), which is given by:

$$p_{ij}^{k} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{k \notin tabu_k}\tau_{ik}^{\alpha}\eta_{ik}^{\beta}} & if\ k \notin tabu_k \\ 0 & otherwise \end{cases}$$

At the beginning of the algorithm, a pheromone value is initialized for each edge. With regards to pheromone updating, a pheromone trail is deposited only by the global best ant over the edge of the best route so far. The pheromone is updated at the end of each iteration by using the following function:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs} \quad (4.18)$$

The pheromone trail value in Best-Worst Ant System also evaporates over time from the edges as all the other Ant Colony Optimization algorithms. The evaporation happens

immediately after the ant chooses the edge for its tour. The evaporation equation is given by:

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0 \quad (4.19)$$

The difference with Best-Worst Ant System is that the pheromone evaporates only from the worst route [70] [71]. Also, the pheromone trail on each row of the pheromone matrix is subject to mutation to introduce diversity in the search based on a probability called p_{mut}.

4.2.3 Bee Colony

There are thousands of kinds of insects that behave in different patterns but they are all capable of solving complex tasks. They are robust, flexible, organized and adaptive to changes in their environment. Bees in colonies are self-organized and follow very simple rules of behavior, like all other individual insects in nature.

Karl Von Frisch 1973 investigated bee behavior more closely, and came up with an excellent understanding of the language of communication between bees in a colony [72]. Frisch did not believe the biologists' idea that all insects are color-blind, so he undertook experiments. He started testing the ability of bees to see color, and the more he tested the more he discovered interesting facts. One of the important facts he discovered was the communication tool used among bees, which he named the Waggle dance. The Waggle dance is a figure-eight dance pattern starting with a straight waggle run then a turn to the right back to the starting point, and again another straight waggle run then a turn but to the left this time and back to the starting point. Bees repeat these steps a few times to inform each other about the direction and distance of the location of the food source. The length of the straight waggle run shows the distance to the food source, where the angle of the dance relative to the sun shows the direction to it [72] [73].

In nature, bees fly out of the hive looking for flowers to collect nectar and carry it back to the hive where they store it. When a bee finds a flower full of nectar she attracts other bees, nestmates, to follow her by dancing. Dancing is the bees' way to share information about the source of the nectar and each hive has its own dance floor area to distinguish them from other hives. Bees will follow the dancing bee to the source of the nectar and the dancing bee will begin collecting the nectar until she is full. Afterwards, she returns to the hive, hands over the nectar to the storer bee and flies out again to collect more nectar. There is no obvious reason to explain why the bees will decide to follow a specific bee dancer rather than another, but the number of attracting bees always correlates with the quality of the nectar source[73] [74].

4.2.3.1 Bee Colony Optimization

Bee Colony Optimization (BCO) is a new direction for those studying in the field of swarm intelligence, and the interest of research applies to a variety of areas, such as computing and engineering. The Bee Colony Optimization was proposed first in 2003 by Panta Lucic and Dušan Teodorovic to solve the Traveling Salesman problem [74]. The Bee Colony Optimization algorithm is a population-based algorithm inspired by bees' behavior in nature. The algorithm is designed to solve complex optimization problems by creating artificial bees called multi-agents that search for solutions to problems by behaving partially like and partially different from the bees in nature [74] [75].

At the beginning of the Bee Colony Optimization algorithm each agent, or artificial bee, is initially located in the hive to start exploring the search space with a series of predefined moves to find a solution.

Set parameter values

B= number of bees in the hive;

It= number of moves during one forward iteration;

Start; //all bees are in the hive

1. Set every bee to an empty solution;
2. For every bee (i=0; i<B; i++) do: //the forward pass
   i. Set Counter=1; // for constructive moves in the forward pass;
   ii. Evaluate all possible constructive moves;
   iii. According to evaluation, choose one move using the roulette wheel;
   iv. Counter=Counter+1; if Counter<=It Go to step ii.
3. All bees return to the hive; //the backward pass;
4. Sort the bees by their objective function value;
5. Every bee decides randomly whether to continue its own search and become a recruiter or to become a follower;
6. For every follower, choose a new solution x_i from recruiters using the roulette wheel;
7. If solutions are not completed go to Step 2;
8. Evaluate all solutions to find the best one and update $x=x_i$;
9. If the stopping condition is not met go to Step 2;
10. Output the best result x found.

Figure 4.3: BCO Algorithm Pseudocode [74]

One or more partial solutions are generated at each iteration during the algorithm search. Each agent generates only one partial solution for the problem and continues to search to find a new partial solution or improve an existing one. This phase is called the forward pass and during the next phase, referred to as the backward pass, all the agents, or artificial bees, go back to the hive, share information directly about the quality of the partial solution i.e. the object function value and make a decision [74] [75] [76].

Based on a probability function, the bee decides to continue with its own exploring if she has a high-object function value and dances to attract other bees, or nestmates; otherwise the bee will give up on the created partial solution and become a follower. The bees will go through a second forward pass and expand the previous discovered partial solutions, after which they will return to the hive during a second backward pass. The agents keep performing the two previous phases iteratively until feasible solutions are created or until they reach a stop condition, then the best solution will be chosen as the solution for the current problem [76] [77]. See Figure 4.3 for a full BCO algorithm pseudocode.

4.2.3.2 Artificial Bee Colony

The Artificial Bee Colony algorithm (ABC) is an optimization algorithm developed by Karaboga (2007) that simulates honeybee behavior [78] [79] [80]. The Artificial Bee Colony has two types of honeybees: employed bees that fly to a food source they visited before, and unemployed bees. The unemployed bees are divided into two types: scout bees, that search for new food sources around the hive randomly, and onlooker bees, that wait in the hive for the information that will come from the employed bees to establish a food source and make decisions. All of them share one specific purpose, to search the area around the hive randomly to find a food source and collect nectar [78] [79] [80].

1. Initialize the population of solutions x_i, $i = 1, \ldots, SN$ // the food source number
2. Evaluate the fitness fit_i population
3. Set cycle=1
4. repeat
5. For each employed bee {

Produce new solutions $v_i$for the employed bees by using (4.22) and evaluate them;

Apply the greedy selection process for the employed bees;}

6. Calculate the probability values P_i for the solutions x_i by (4.20)
7. For each onlooker bee {

   Produce the new solutions v_i for the onlookers from the solutions x_i

   selected depending on P_i and evaluate them

   Apply the greedy selection process for the onlookers}

8. If the scout abandons a solution, then replace it with a new random
9. produced solution x_i by (4.23)
10. Memorize the best solution achieved so far
11. cycle = cycle + 1
12. until cycle = MCN // the maximum cycle number

Figure 4.4: ABC Algorithm Pseudocode [78]

The position of the food source is saved in the employed bees' memories and when a new food source is discovered they compare between the two sources based on the amount of nectar. If the amount of nectar in the new food source is higher than the previous source, the employed bees will memorize it and forget about the old one. Otherwise they keep the position of the one that is already there in their memories. By dancing the waggle dance,

the employed bees attract other bees to follow them and share information with the onlookers about the location of the new source they found, such as direction, distance and amount of nectar. Immediately, onlookers evaluate this information and again they notice the difference in the amounts of nectar by comparing the two sources, just like the employed bees did. The onlooker bees prefer food sources with a lot of nectar, forget the position of the low nectar sources and only remember the preferred one [80] [81] [82]. In the ABC algorithm the artificial onlooker bee uses the following probability value p_i to choose a food source:

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{4.20}$$

fit_i- the fitness value of the solution i

SN- the number of food source

The fit_i value of the ith employed bee is obtained as follows:

$$fit_i = \begin{cases} \frac{1}{1+f_i} & if\ (f_i \geq 0) \\ 1 + abc(f_i) & if\ (f_i < 0) \end{cases} \tag{4.21}$$

The next expression produces a candidate food position from the old one in memory:

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{4.22}$$

ϕ_{ij}- random number between [-1,1]

k $\in \{1,,2,\dots,SN\}$- random index and different from i

j $\in \{1,,2,\dots,D\}$- random index

The employed bees carry the nectar to the hive and those whose food source has been abandoned can become a scout and start to search for a new food source, or they can be follower bees and follow the employed bees after watching the waggle dances. The ABC

algorithm produces a random position to replace the abandoned source x_i *limit* with control parameter represented in operation (4.23).

$$x_i^j = x_{min}^j + rand[0,1]\left(x_{max}^j - x_{min}^j\right) \quad (4.23)$$

The Artificial Bee Colony algorithm initializes and randomly distributes the food source positions. The employed bees are also initialized and distributed randomly between the food sources. Each employed bee is assigned to one food source so the number of food sources matches the number of employed bees. The food source positions represent possible solutions for the problem and the nectar amount, or fitness value, represents the quality of that solution. The ABC algorithm has three control parameters, which are: SN, the number of food source equal to the number of employed or onlooker bees, and the value of *limit and MCN*, the maximum cycle number [82] [83]. A full pseudocode for the ABC algorithm is given by Figure 4.4.

4.3 The proposed Self-Organization Movement Control

The movement of the mobile nodes in MANETs is random and the performance depends on the density, distribution and movement of the mobile nodes. Because of the changing nature of MANET topology, the network suffers from the rapid disconnection between the nodes, which causes a significant decrease in the delivery ratio of the network. Controlling the movement of nodes in MANETs and keeping them in the transmission range to improve the network performance is always a problem. The Slow Down Speed algorithm (SDS) is a self-organization control algorithm proposed in this Section 4.3 as a dynamic-mobile node-movement control solution based on observation of network traffic conditions. It aims to control the mobile node movements and extend the length of time that the forwarding path lasts and therefore reduce the number of changes in the routing table.

At the beginning, the mobile nodes start moving randomly at different speeds. The mobile nodes represent the agents, and they have no control over each other or over the network. During the random movement, the network traffic is observed waiting for the routing protocol to detect the forwarding path, which is used in the SDS algorithm as the local information. Based on that local information, the agents decide which rule they should follow to transmit the packets from the source node to the destination node. When a few mobile nodes agents take a place in between the source and the destination, the routing protocol finds a forwarding path, and the nodes are divided according to this local information into forwarding nodes, and source and destination communicating nodes. At this time, each active forwarding agent that is actually a part of the forwarding path used to transmit the packets over the network decides to follow a simple rule to slow down. This simple self-control behavior presented in changing the speed of the active forwarding agent based on the local information of the traffic condition results in extending the life time of the forwarding path for a longer period. The slower the agents the longer the forwarding path remains, and the longer the time the forwarding path remains live, the higher the packet delivery ratio.

This simple SDS self-organization behavior of the agents in a random movement network creates a more complex MANET that is actually able to solve the problem of the rapid disconnection of the source from the destination. It also decreases the changes in the routing table in addition to the changes in the MANET topology, resulting in a higher packet delivery ratio to network performance.

4.4 The proposed Ant Colony Optimization Movement Control

MANET is a high dynamic network where all the mobile nodes are moving in random directions at random speeds. This rapid movement of the mobile nodes causes changes in the MANET topology, which increase the disconnection between the source node and the destination node. The continual disconnection affects network performance and decreases packet transmission rates. In order to improve MANET performance, a dynamic-movement node-control algorithm based on Swarm Intelligence has been proposed to control the mobile node movement according to the node's awareness of the traffic condition. In this thesis the proposed Ant System Node Control algorithm, ASNC, is implemented to control the mobile node movements in MANET according to the traffic condition in the network. The ASNC is adapted from the Ant Colony Optimization algorithm, and more specifically the Ant System algorithm. A variety of related Swarm Intelligence (SI) studies and algorithms for MANETs have been proposed by other researchers where they adapted different SI algorithms, such as the Ant Colony Optimization and the Bee Colony Optimization algorithms. Examples of related work where the researchers adapted the Ant Colony Optimization algorithm to evaluate routing protocols for MANET include: Ant-colony-based Routing Algorithm (ARA) [5], Simple Ant Routing Algorithm (SARA) [6], Ad-hoc Networking with Swarm Intelligence (ANSI) [7], AntHocNet hybrid algorithm [8], AntNet routing algorithm [9], and HOPNET routing algorithm [10].

At the start, the mobile nodes are moving randomly in MANET without any control. The mobile nodes act like Ant-agents, they explore the simulation area randomly until the routing protocol detect the first forwarding path between the source and the destination.

The source node represents the ant colony while the destination node represents the position of the food source. At this time, each active forwarding node, ant-agent, leaves a pheromone trail on its position, and the deposited pheromone trails show the forwarding path that the packets transmit through from the source to the destination. The other mobile nodes, ant-agents, will start moving towards these pheromone trail positions deposited by the forwarding ant-agents. Each time a forwarding path detected in MANET, the active forwarding ant-agents deposited new pheromone trails on the forwarding positions. The pheromone trails are not permanent, they evaporate over time from the far forward positions faster than the closest forward positions to the source and the destination. The ant-agents take several positions to solve the problem of the rapid disconnection between the source and the destination in MANET. In this way, the packets always find a forwarding path from the source to the destination to transmit in MANET.

In the ASNC, the ant-agents follow local decisions to move from forward position to another forward position using the Ant System probability rule (Equation 4.5). The deposited pheromone in a forward position will be modified by other ant-agents, mobile nodes, moving toward the same forward position, otherwise the pheromone evaporates from the forward position over time using the Ant System update equation (Equation 4.7). The pheromone trail deposited based on the traffic condition of MANET control the ant-agents movement during the simulation to find a forwarding path between the communication nodes, the source and the destination, and improve the network performance.

Chapter 5

Simulation Environment

5.1 Introduction to Network Simulator 2

The Network Simulator 2, also known as NS2, was developed by researchers at Berkeley University in 1989 under UNIX environment to simulate different types of wired, e.g. LANs, and wireless, e.g. MANETs, networks [84]. NS2 is an open source simulation tool, which means that the source code is available to be developed and improved by the users for free. This simulator is an object-oriented event-driven network simulation tool where the events are not executed on a fixed time interval. Each event has a time stamp showing the time the event will occur, which has to be greater than the current simulation clock to insure that the simulation always goes forward in time. The events are stored in a list, called an event list, and the simulation executes these events by retrieving and removing them in order according to the time of occurrence [84] [85].

NS2 is a popular simulation tool in the network research area, which provides a variety of ways to design network topologies with different scenarios, traffic generators such as FTP and CBR, and protocols including TCP and UDP to study their behaviors. Over years, several developers and programmers have improved the first NS2, the REAL network simulator, since its inception in 1989, and the development still continues [85] [86].

5.2 Network Simulator 2 Architecture

NS2 was written in two languages, C++ and Object-oriented Tool Command OTCL both of which are linked together using TclCL. NS2 runs on a UNIX environment or Linux without any complications, however, it is possible to run it on other platforms, such as

Windows via Cygwin (UNIX emulator) and Mac [84] [86]. In order to write a scripting file for NS2, programmers have to use a scripting language called Tool Command Language, or Tcl. The Tcl was developed by J. Ousterhout in 1988 as dynamic typing where every operation is a command and everything in the script can be treated as a string [87]. NS2 files are executed using "ns" command in the terminal followed by a space and the name of the scripting file as follows: ns filename.tcl

The simulation then starts and the results can be visualized via graphical and interactive tools, such as the animation tool Network AniMator (NAM), which visualizes the mobile node movements and how the packets are transmitted through the network (Figure 5.1). Xgraph (Figure 5.2) is another tool used to plot graphs, e.g. throughput graphs [84] [85].

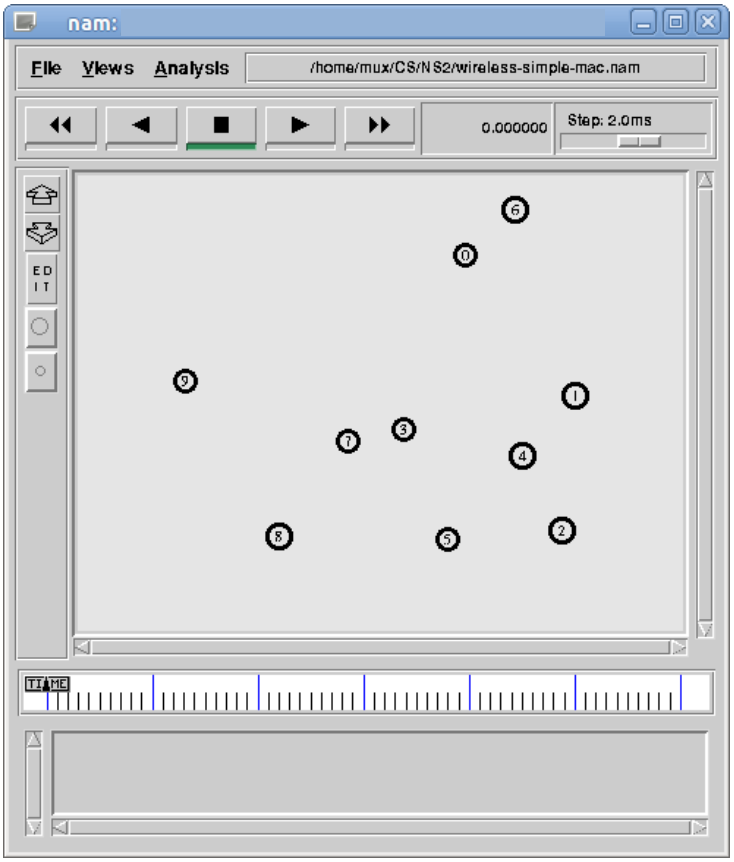

Figure 5.1: Network AniMator

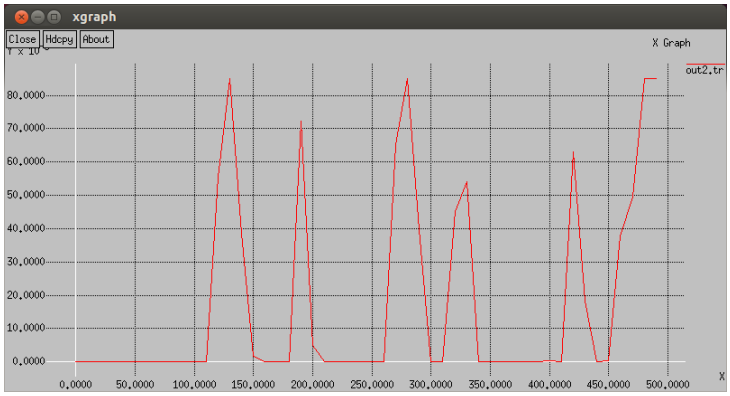

Figure 5.2: Xgraph

The random distributed mobile node positions, velocities and movement directions are initialized in NS2 using a Setdest command or a Setdest tool [84]. The Setdest tool uses the Random Waypoint Model to automatically generate a large number of mobile node positions along with their destinations and speeds in the following syntax:

setdest -v 1 -n $numnodes -p $pt -M $maxspeed -t $simtime -x $maxx -y $maxy

where:

$numnodes, mobile node total numbers in the simulation

$pt, pause time

$maxspeed, the maximum speed allowed for the mobile nodes

$simtime, simulation run time

$maxx, the maximum X coordinate for the simulation area

$maxy, the maximum Y coordinate for the simulation area

It is also possible to create a trace file, a text-based output, to analyze and process the simulation results, such as calculating the delay, delivery ratio and throughput, in order to measure the network performance. AWK language, developed in 1970 and named after the

authors' family names A. Aho, P. Weinberger and B. Kernighan, is used to extract and calculate the data from the NS2 trace file [88].

5.3 Modification on Network Simulator 2

The dynamic-movement node-control algorithms utilize the NS2 to simulate the experiments using the MANETs. In the experiments, the proposed SDS and ASNC algorithms were performed based on the awareness of the traffic conditions during the simulation run time to control the random movement of the mobile nodes. The algorithms are written in Tcl, which does not provide the necessary information about the network routing tables. To be able to extract the routing table information the two routing protocols, AODV and DSDV, were modified in NS2. It was also important to get the right position for a mobile node at a specific time during the simulation run; unfortunately NS2 did not provide timely X and Y coordinate information for a mobile node, so that had to be modified as well. John DeDourek, professor at UNB, had modified NS2 to extract the correct positions of the mobile nodes during the simulation run and the modification in Section 5.3.3 is a modification based on his work. As NS2 files are written in C++ and linked with OTCL, all the modifications have been done using C++ language.

5.3.1 AODV

AODV, one of the reactive routing protocols available for MANET in NS2, has been used in this study on dynamic-movement node-control algorithms. Each entry in the AODV routing table contains information about the packet destination, route sequence number, hop counts needed to reach the destination, next hop, route expiry time, and flags. These variables are declared in the AODV routing table header aodv_rtable.h, which is located in the aodv folder. A new function, called ForwardPaths that takes a routing table of type

aodv_rt_entry as an argument, has been added to aodv.cc and declared in the AODV header aodv.h as follows: AODV::Forwardpaths(aodv_rt_entry *rt).

This function opens a text file to write the routing table entry information that was actually used to transmit a packet over a forwarding path. The writing text file contains the node id *(index)*, current time *(CURRENT_TIME)*, destination *(rt->rt_dst)*, next hop *(rt->rt_nexthop)*, hop counts *(rt->rt_hops)*, route sequence number *(rt->rt_seqno)*, route expiry time *(rt->rt_expire)*, flags *(rt->rt_flags)*, X coordinate *(xpos)*, and Y coordinate *(ypos)* respectively, (Figure 5.3).

```
0 39.522581 1 6 7 4 49.522581 1 50.000000 400.000000
6 39.528863 1 4 6 4 49.528863 1 215.988352 553.086271
4 39.535225 1 11 5 4 49.535225 1 392.469308 575.056250
11 39.541407 1 16 4 4 49.541407 1 487.443696 381.596208
16 39.547189 1 10 3 4 49.547189 1 674.974626 229.450740
10 39.553352 1 14 2 4 49.553352 1 920.019073 247.328439
14 39.559373 1 1 1 4 49.559373 1 948.287665 360.825409
0 49.554501 1 6 8 4 49.522581 1 50.000000 400.000000
6 49.555518 1 15 7 6 59.555518 1 223.229006 534.432421
15 49.561780 1 9 6 6 59.561780 1 370.423781 663.037535
9 49.567722 1 11 5 6 59.567722 1 487.386541 581.945456
11 49.573983 1 16 4 6 59.573983 1 514.728394 393.969527
16 49.579846 1 10 3 6 59.579846 1 692.729822 241.066723
10 49.586028 1 14 2 6 59.586028 1 919.231459 263.574320
14 49.592069 1 1 1 6 59.592069 1 947.039449 370.528963
0 59.250000 1 6 8 6 69.250000 1 50.000000 400.000000
6 59.255618 1 15 7 6 69.255618 1 230.233047 516.388149
15 59.261560 1 9 6 6 69.261560 1 358.888243 659.614267
9 59.267722 1 11 5 6 69.267722 1 498.349725 580.591536
11 59.273823 1 16 4 6 69.273823 1 541.103990 405.930579
16 59.279886 1 10 3 6 69.279886 1 709.893276 252.295570
10 59.285848 1 14 2 6 69.285848 1 918.470101 279.278628
14 59.291669 1 1 1 6 69.291669 1 945.832872 379.908827
```

Figure 5.3: AODV forwarding Paths

To determine the position of the forwarding mobile node the following code was used directly from C++ rather than from Tcl:

*MobileNode *m;*

*m = (MobileNode *) (Node::get_node_by_address(index));*

m-> update_position();

xpos = m-> X();

ypos = m-> Y();

The ForwardPaths function is called from inside another function in aodv.cc, the forward function, that takes a routing table entry, a packet and a delay as arguments as follows:

*AODV::forward(aodv_rt_entry *rt, Packet *p, double delay)*

More specifically, only the forwarding paths between the stationary communication nodes, the source (node 0) and the destination (node 1), with no errors or dropped packets, are considered. This has been accomplished by adding an 'if' condition to the forward function. These improvements made it possible to extract all the needed information about the traffic conditions to control the mobile node movements using the SDS and ASNC algorithms. During the simulation run time, the forwarding paths are written in a text file using C++, and are read by the Tcl scripting file simultaneously.

5.3.2 DSDV

The DSDV proactive routing protocol is the other MANET routing protocol that has been used in the experiment on dynamic movement node control algorithms. DSDV routing table entries contain information about the destination, next hop, metric (hop count), and route sequence number, which are all declared in rtable.h, the routing table header located in the dsdv folder. As in AODV, a new function, ForwardPaths, has been added to dsdv.cc and declared in the DSDV header dsdv.h. This function takes a routing table of type rtable_ent as an argument as follows: DSDV_Agent::Forwardpaths(rtable_ent *prte)

It has the same functionality as the ForwardPaths function in AODV, which writes the routing table entry information that was actually used to transmit a packet over a forwarding path into a text file. The text file contains the node id *(myaddr_)*, current time *(now)*, destination *(prte->dst)*, next hop *(prte->hop)*, metric *(prte->metric)*, route sequence number *(prte->seqnum)*, X coordinate *(xpos)*, and Y coordinate *(ypos)* respectively, (Figure 5.4).

```
0 113.509981 1 4 5 50.000000 400.000000
4 113.522182 1 8 4 212.600653 506.324689
8 113.535139 1 9 3 415.836549 567.635846
9 113.544255 1 18 2 642.309652 562.812912
18 113.553071 1 1 1 784.181263 432.474439
0 161.000000 1 4 5 50.000000 400.000000
4 161.005718 1 8 4 221.989592 498.800995
8 161.011900 1 9 3 445.757732 547.925253
9 161.018102 1 16 2 664.548081 555.877133
16 161.024004 1 1 1 803.291464 497.048610
0 373.800000 1 5 5 50.000000 400.000000
5 373.805678 1 17 4 261.289707 465.267385
17 373.811741 1 18 3 488.268180 511.776907
18 373.817781 1 9 2 578.221025 548.211809
9 373.823844 1 1 1 781.465488 555.781600
0 483.900000 1 4 5 50.000000 400.000000
4 483.906038 1 8 4 255.058131 396.164606
8 483.912040 1 18 3 460.251491 474.815848
18 483.918081 1 9 2 569.351448 487.938356
9 483.924003 1 1 1 759.674239 581.987879
```

Figure 5.4: DSDV forwarding Paths

To determine the position of the forwarding mobile node the following code was used directly from C++ rather than from Tcl:

node_-> update_position();

xpos = node_-> X();

ypos = node_-> Y();

The ForwardPaths function is called from inside another function in dsdv.cc, the forward packet function that takes a packet argument as follows:

*DSDV_Agent::forwardPacket (Packet * p)*

Particularly, the forwarding paths between the stationary communication nodes, the source (node 0) and the destination (node 1), with no errors or dropped packets are considered, accomplished by adding an 'if' condition to the forward packet function.

The improvements in the DSDV routing protocol in NS2 made it possible to extract all the needed information about the traffic conditions to control the mobile node movements using the SDS and ASNC algorithms. During the simulation run time, the forwarding paths are written in a text file from the C++, and read by the Tcl scripting file simultaneously.

5.3.3 Mobile Nodes

The mobile node files in NS2 have been modified to extract the correct positions of the mobile nodes during the simulation run at a specific time. The C++ files that have been modified are *mobilenode.cc* and the header *mobilenode.h* in the *ns-allinone-2.35/ns-2.35/common* directory. Two inline functions, called *Xp* and *Yp*, have been added to the mobile node header in order to get the updated X and Y coordinates:

inline double Xp { update_position(); return X_;}

inline double Yp { update_position(); return Y_;}, where:

update_position() is an existing function in NS2 that updates the position of the mobile nodes

*X_* binding variable in NS2

*Y_* binding variable in NS2

Two OTcl commands have been created in the command function that take two arguments to return the values of the updated X and Y coordinates. Within the command function, in *mobilenode.cc* an "if/then" condition has also been added to compare the arguments and the names of the OTcl commands, and return *TCL_OK*.

To invoke the new OTcl commands from Tcl script *Xp* and *Yp* function names have been used instead of *set X_* and *set Y_*, for example:

set Xpos [$node_($i) Xp]

set Ypos [$node_($i) Yp]

instead of :

set Xpos [$node_($i) set X_]

set Ypos [$node_($i) set Y_]

The modification insures a return of the correct position of the mobile nodes at a specific time in order to control the movements of the mobile node in the implementation of the proposed algorithms.

5.4 Simulation Study

In this thesis, a Random Movement scenario has been simulated using NS2, in addition to self-organization node-control and Swarm Intelligence node-control algorithm scenarios using MANETs. For comparison purposes, two different routing protocols, AODV and DSDV, were used in the simulation. The network performance was measured based on different MANET performance metrics.

5.4.1 Simulation Environment

The simulated experiments were done in a square simulation area of 1000x1000 meters with different densities of mobile nodes: 5, 10, 20, 30, and 40 nodes. The mobility nodes

were randomly distributed using the Random Waypoint Model with 250 m transmission range and an average speed of 4 m/s or 9 mph, which is equal to speed of riding a bicycle. The source and destination communication nodes are stationary, with the source node located at (50, 400) and the destination node located at (950, 600). The Constant Bit Rate (CBR) was used to generate the data packets, and the size of each packet was 512 bytes (Table 5.1). The link data rate was 1 Mbps, and the buffer size for each node was 50 packets. Each simulated experiment was run three times for 500 seconds with the average of the measurements calculated.

Table 5.1: NS2 Parameters

Parameters	
Simulator	NS-2.35
Channel Type	Channel / Wireless Channel
Network Interface Type	Phy/WirelessPhy
Mac Type	Mac/802.11
Radio-Propagation Type	Propagation/Two-ray ground
Interface Queue Type	Queue/Drop Tail
Link Layer Type	LL
Antenna	Antenna/Omni Antenna
Maximum Packet in ifq	50
Area (n * n)	1000 x 1000m
Source node location	(50, 400)
Destination node location	(950, 600)
Source Type	CBR over UDP packetSize_ 512 interval_ 0.05
Simulation Time	500 s
Routing Protocol	AODV and DSDV

5.4.2 Performance Metrics

The performance of the proposed dynamic-movement node-control algorithms was evaluated using NS2. Different performance metrics were used to evaluate the simulation results. The performance metrics are illustrated in more detail in sections 5.4.2.1, 5.4.2.2 and 5.4.2.3.

5.4.2.1 Packet Delivery Ratio

Packet Delivery Ratio, PDR, is the number of the packets received by the destination divided by the number of the packets sent by the source. As long as the mobile nodes stay connected to each other and transmit packets between the source and the destination the PDR results will be high. The higher the PDR results the higher the mobile node connectivity. The proposed SDS and ASNC algorithms aim to control the mobile node movements and increase the packet transmission rate by keeping the nodes connected as much as possible. The best way to measure the performance of these algorithms in MANET is by comparing the results using PDR. AWK has been used to extract and calculate the PDR results from the NS2 trace file.

5.4.2.2 End-to-End Delay

End-to-End Delay, or packet delay, is the average transmission time for the data packet from when it is sent from the source until it reaches the destination. The delay time includes all the possible delays that may happen during the transmission through MANET, such as any delays caused by buffering, queuing, retransmission, or propagation. End-to-End delay has been used to measure the network performance during the experiments. The delay results extracted and calculated from the trace file using AWK.

5.4.2.3 Forward Path Time Detection

Forward Path Time Detection, FPTD, represents the time when the routing protocol detects the first forwarding path. Due to the high dynamic nature of MANETs, some routing protocols take a longer time than other protocols to find a route and forward the packets from the source to the destination. This delay in finding the first forwarding path affects network performance and PDR. The FPTD is one of the performance metrics to measure the performance of the proposed SDS and ASNC algorithms.

Chapter 6

Study of Self-Organization Movement Control

6.1 Slow Down Speed Algorithm

The proposed SDS (Slow Down Speed) algorithm is a self-organization algorithm that aims to control mobile node movements in MANET based on awareness of the network traffic. The experiments have been done in NS2 using the RWP model. The mobile nodes move to random destinations at random speeds, with an average speed of 4 m/s. The mobile nodes, or agents, move in a restricted area of 1000x1000 meters, while the source and destination communication nodes are stationary and located at (50,400) and (950,600) coordinates respectively. The stationary nodes communicate with each other using CBR traffic generators over UDP to generate and send packets. The SDS experiment has five different scenarios: Random, Speed/2, Speed/6, Speed/10, and Speed Zero. Each scenario is run for different mobile node densities: 5, 10, 20, 30, and 40. Each mobile node density is run over two routing protocols, AODV and DSDV, three times each and the average results taken. The simulation run time is 500 seconds for each experiment.

- **Random Scenario**

The Random scenario case is the basis of all the experiments, in which all the mobile nodes are moving randomly, using RWP model, toward the destinations with random but constant speeds. The speed for each mobile node is calculated based on the speed equation:

$$S_i = D_i/T \tag{6.1}$$

where D is the distance between the current position and the next position for node i and T is a constant time period to reach the destination.

In this scenario there is no control over the movements of the mobile nodes during the entire 500 seconds of the simulation run time. The source communicates with the destination when a forwarding path is available, and because of the dynamic movement of the mobile nodes the forwarding path may not last for a long time, or may be broken. In some cases, e.g. low mobile node density, the routing protocols are not able to detect a forwarding path, which means no packets are transmitted through the network.

- **Speed/2 Scenario**

At Speed/2 scenario, the mobile nodes start moving randomly at different speeds without control over each other or over the network, as in the Random scenario. During the simulation the network traffic is observed waiting for the routing protocol to detect the first forwarding path, which is used in SDS algorithm as local information to control the movement of the mobile nodes. The mobile nodes continue moving until a few nodes take a position in between the source and the destination. The mobile nodes then become forwarding nodes, mobile medium, to forward data packets through the forwarding path that has been detected by routing protocol. The packets are transmitted between the source and the destination through these forwarding nodes, mobile medium, and the network performance is measured based on how many packets the mobile medium carries and how long the nodes stayed connected. The information about the detected forwarding paths are written to a text file directly from the routing protocol C++ files in NS2. The text file is read by the SDS Tcl script at the same time it is written by the C++ code to get the forwarding path information. Based on the forwarding path information in the text file, the active forwarding nodes that transmit packets through the detected forwarding path decides to follow a simple rule. While the forwarding nodes decide to slow their speed down by

half, the other mobile nodes are not affected by the forwarding nodes decision and they continued moving randomly. For example, if the forwarding nodes are nodes 5, 3, 9, and 6 moving randomly at speed 3.2, 2.5, 4, and 1.8 m/s respectively, then the speed of the forwarding nodes after slowing down by half become 1.6, 1.25, 2, and 0.9 m/s for nodes 5, 3, 9 and 6 respectively. Slowing down the speed of the active forwarding nodes is a self-organization control behavior that the forwarding nodes follow according to awareness of traffic condition in MANET. Slowing down the mobile node speeds by half results in extending the life time of the forwarding path, which means that the source and the destination stay connected for a longer period of time. This also decreases the changes to the routing tables along with the changes in MANET topology compared with the Random scenario. In this scenario of slowing down the speed by half, the network performance and PDR increase in general.

- **Speed/6 Scenario**

Speed/6 scenario is the same as Speed/2 scenario where the mobile nodes move randomly in the beginning of the experiment waiting for the routing protocol to detect the first forwarding path. When the information about traffic condition becomes available for the mobile nodes in MANET the forwarding nodes take a decision to divide their speed by six. Decreasing the speed of the forwarding nodes results in an increase in the forwarding path life time and the number of packets transmitted between the source and the destination. Slowing down the speed of the forwarding nodes also decreases the number of changes in the routing table. The network performance is measured based on how many packets the mobile medium can carry during the simulation run time.

- **Speed/10 Scenario**

In the Speed/10 scenario decrease the random speed of the forwarding nodes in MANET is decreased by 10, which is slower than the speed in Speed/2 and Speed/6 scenarios. The same technique used in Speed/2 and Speed/6 scenarios has been used in Speed/10 scenario. The traffic condition is observed, waiting for the moment when the routing protocol detects the first forwarding path. Based on the information of the forwarding path the forwarding nodes are determined and then the forwarding nodes decide to follow a simple self-control action, which is slow down speed. The decision of slowing down the speed of the forwarding nodes does not affect on the other mobile nodes in MANET. The other mobile nodes remain moving randomly at random speeds without any self-organization control. Slowing down the speed extends the life time for the forwarding path resulting in the transmission of more packets through the mobile medium and increasing the performance of the network.

- **Speed Zero Scenario**

In Speed Zero scenario the active forwarding nodes stop moving once the forwarding path is detected by the routing protocol. Based on information about the traffic condition coming from C++ routing protocol files in NS2, the forwarding nodes take a decision to set their own speed to zero. Setting the speed of the forwarding nodes at speed zero means that the nodes are no longer moving but they stay still on the forwarding position until the end of the simulation run time. Keeping the forwarding nodes on the forwarding positions insure that a high number of packets are transmitted between the source and the destination from when the first forwarding path is detected, for the rest of the simulation run time. A high PDR results in a high performance of the network.

6.2 SDS Results and Analysis

The proposed SDS algorithm scenarios were tested using two different routing protocols, AODV and DSDV, with different mobile node densities: 5, 10, 20, 30, and 40. The average result for three simulation runs has been taken for each mobile node density during the 500 seconds of the simulation run time. The performance metrics PDR, End-to-End Delay and FPTD have been used to measure the performance of the network.

6.2.1 Performance of SDS over AODV

AODV experiment results tested measurement of the performance of the SDS algorithm in MANET. The analysis of the PDR results are provided in Section 6.2.1.1, followed by the End-to-End Delay results in Section 6.2.1.2 and finally the result of the FPTD in Section 6.2.1.3.

6.2.1.1 PDR for SDS

Slowing down the speed of the forwarding nodes in the experiments using AODV routing protocol in an area of 1000x1000 meters improved the PDR in general (Figure 6.1).

For mobile node density of five, a very low density, the AODV routing protocol was not able to detect any forwarding path during the entire simulation run time, resulting in zero PDR for the five different SDS scenarios: Random, Speed/2, Speed/6, Speed/10, and Speed Zero. Mobile node density of 10 shows PDR of 2% for the Random scenario without any control over the mobile nodes, increasing to 4%, 10% and 30% for Speed/2, Speed/6 and Speed/10 scenarios respectively, and reaching a high PDR of 90% when the forwarding nodes stop moving in Speed Zero scenario. The 20 mobile node density is the most interesting case in SDS, demonstrating the best benefits of the dynamic-movement mobile medium control in MANET.

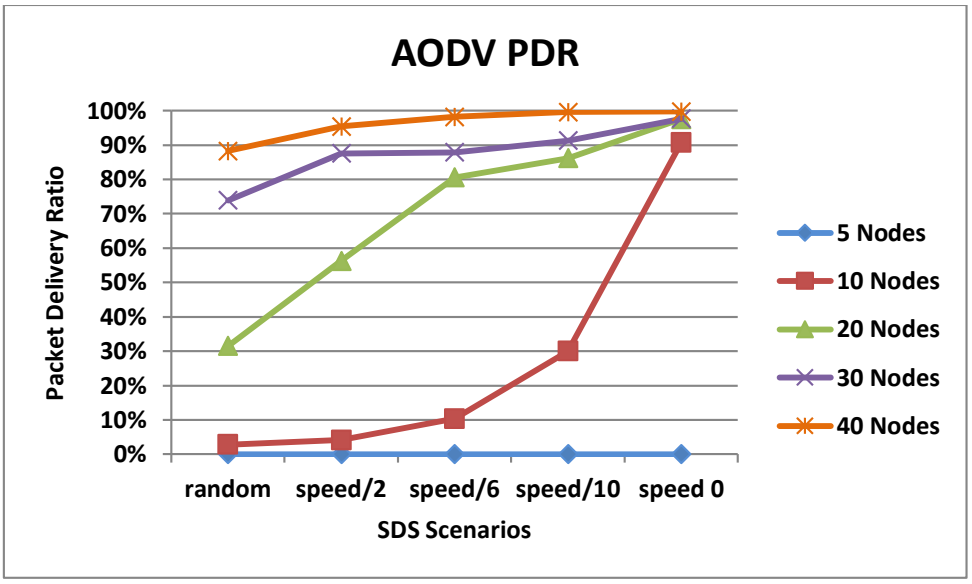

Figure 6.1: PDR for SDS over AODV

The PDR for the 20 mobile node density in the Random scenario is 31%, rises to 56% in Speed/2 scenario and increases to 80% and 86% in Speed/6 and Speed/10 scenarios, finally reaching 98% in Speed Zero scenario. At a mobile node density of 30 the PDR is high at any speed, starting at 74% in Random speed scenario where there is no self-control over the forwarding nodes, then increasing to 87% in Speed/2 and Speed/6 scenarios, followed by 91% in Speed/10 scenario and 98% in Speed Zero scenario. The PDR at high density of 40 mobile nodes in 1000x1000 simulation area ranges between 88% and almost 100% resulting in a high PDR as expected regardless of the speed.

Reducing the forwarding node speeds in a low mobile node density in an area of 1000x1000 meter results in a zero delivery ratio (Figure 6.2). With a low mobile node density of 10 nodes the PDR increased gradually as the speed was slowed down, to reach 90% at Speed Zero.

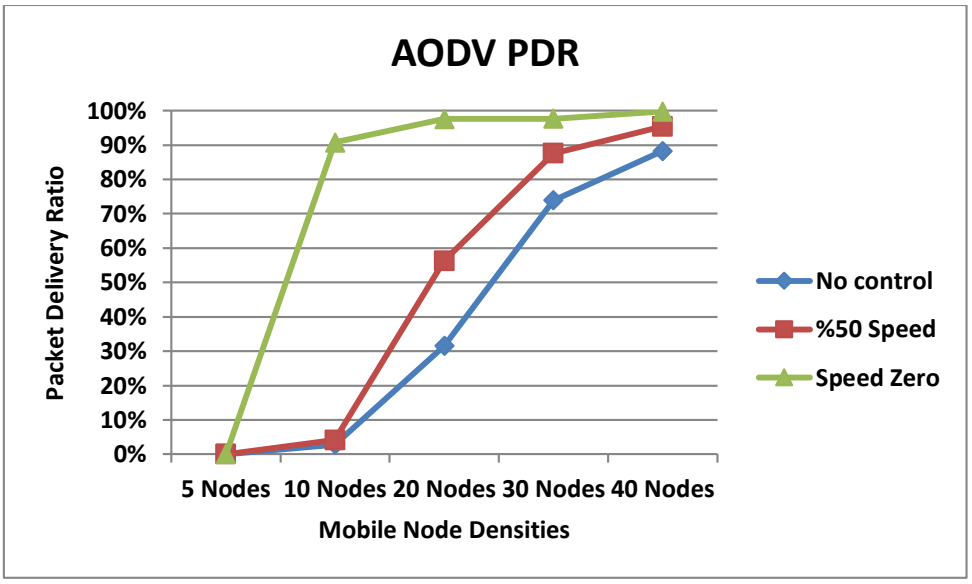

Figure 6.2: PDR2 for SDS over AODV

The performance improved significantly with reducing the forwarding nodes speed by half compared with the original random speeds without control. The PDR is very low at the low mobile node densities, then increases dramatically relative with reducing the speeds, to reach almost 100% when the forwarding nodes stop moving. The network with high mobile node densities, 30 and 40 nodes, perform very well as expected regardless of the speed. The lower the speed of the forwarding nodes the higher the PDR, which results in better network performance.

6.2.1.2 End-to-End Delay for SDS

The very low mobile node density, five nodes, shows no packet delay during the entire simulation run time because the AODV routing protocol was not able to detect any forwarding path and the PDR was zero percent for the five SDS scenarios (Figure 6.3).

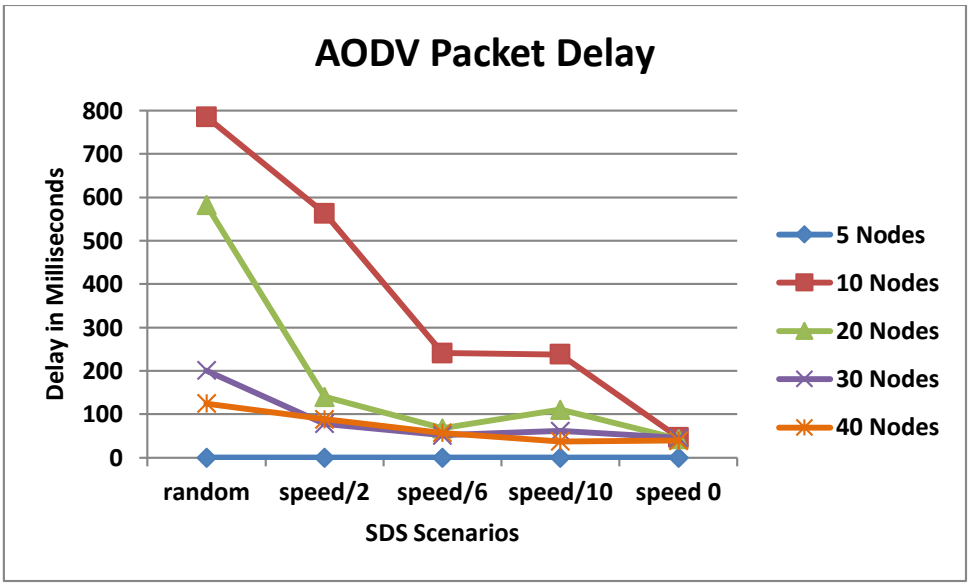

Figure 6.3: Packet Delay for SDS over AODV

For 10 mobile node density the packet delay starts at 785 ms for the Random scenario then decreases to 563 ms, 240 ms and 237 ms by reducing the speed to half, 6% and 10% respectively.

The packet delay for 20 mobile node density decreases significantly from 582 ms in Random scenario to 140 ms by reducing the speed to half, and reaches 42 ms when the forwarding nodes stop moving in Speed Zero scenario. At high mobile node densities of 30 and 40 nodes the packet delay diminishes with reducing the speed and range from 201 ms to 37 ms. As in PDR the 20 mobile node density shows the most beneficial performance for SDS in packet delay. The low and high mobile node densities also performed as expected in the network demonstrating a low packet delay when the PDR is high and vice versa.

6.2.1.3 FPTD for SDS

The experiments of SDS for the five mobile node density shows that no forwarding paths were detected by AODV. While in the 10 mobile node density the AODV routing protocol took an average 190 seconds out of 500 second simulation run time to detect the first forwarding path (Figure 6.4).

The FPTD for the mobile node density of 20 nodes was 25 seconds, reducing by the increase of the nodes to 8 second and 1.2 second for 30 and 40 mobile node densities respectively. Detecting the forwarding paths early increase the PDR and the network performance.

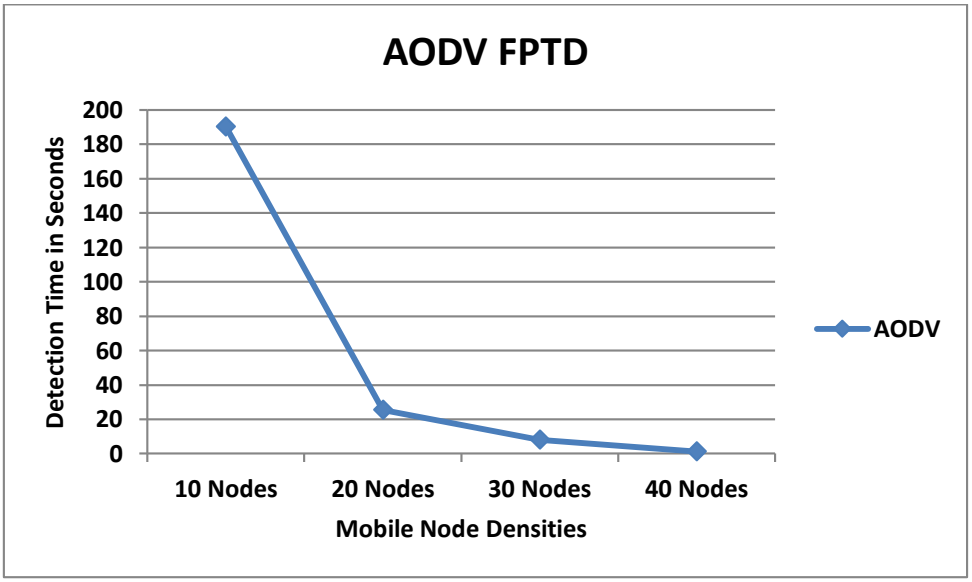

Figure 6.4: FPTD for SDS over AODV

6.2.2 Performance of SDS over DSDV

The performance of SDS experiments running DSDV routing protocol has been measured in terms of PDR, End-to-End Delay and FPTD. The analysis of the PDR results is given in

Section 6.2.2.1, then Section 6.2.2.2 provides the result of the End-to-End Delay, and the result of FPTD illustrated in Section 6.2.2.3.

6.2.2.1 PDR for SDS

The experiments of slowing down the speed for the forwarding nodes using DSDV perform similar to AODV. DSDV did not detect any forwarding path for a mobile node density of five in an area of 1000x1000, resulting in zero PDR for all scenarios: Random, Speed/2, Speed/6, Speed/10, and Speed Zero (Figure 6.5). The 10 mobile node density shows a very low PDR, less than 10%, for all the scenarios except the Speed Zero scenario where the delivery ratio increases to 38%. As in AODV experiments, the 20 mobile node density in DSDV experiments shows an improvement in network performance.

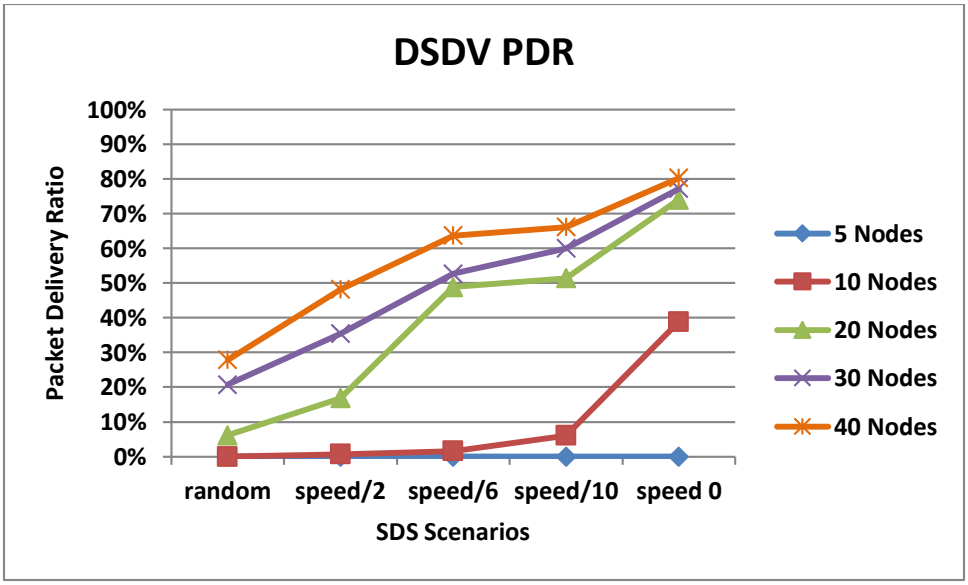

Figure 6.5: PDR for SDS over DSDV

The PDR in the Random scenario is 6%, increasing significantly with reduction in speed to reach 17%, 48% and 51% for Speed/2, Speed/6 and Speed/10 scenarios respectively, and ending up at 74% when the forwarding nodes stop moving in Speed Zero scenario.

These results demonstrate the best benefits of using the self-organization node control proposed in the SDS algorithm. At 30 mobile node density the PDR starts high at 20% in the Random scenario compared with 0.01% and 6% in the Random scenario at 10 and 20 mobile node densities respectively. The PDR then increases gradually as the speed slowed down to reach 35% when the speed is slowed by half, followed by 53% at Speed/6, 60% at Speed/10 and 77% at Speed Zero. The high mobile node density, 40 nodes, perform well at any speed with 28% PDR at the original speed without control and 48% at half speed to range eventually between 64% and 80%.

Reducing the speed by half for the forwarding nodes in different densities of 5, 10, 20, 30, and 40 increases the delivery ratio in line with the increase in the number of nodes (Figure 6.6).

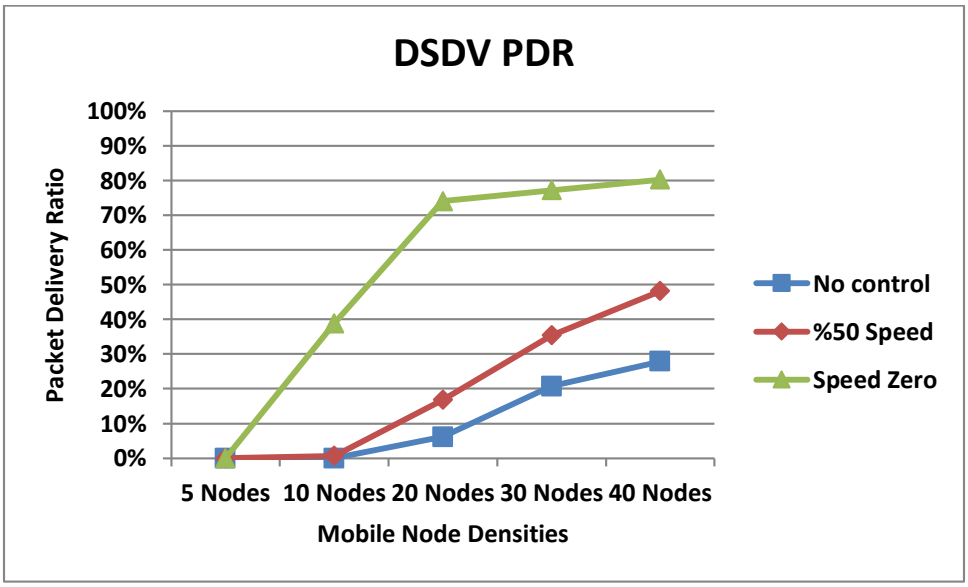

Figure 6.6: PDR2 for SDS over DSDV

Compared with the original random speed of the nodes the network performance improved by 20% when slowing down the speed by half, while in stopping the forwarding nodes the PDR goes higher to range between 74% and 80% for the 20, 30 and 40 node densities.

6.2.2.2 End-to-End Delay for SDS

As in AODV, DSDV experiments detect zero forwarding paths at very low density of five nodes resulting in zero packets delivered, and delays for all the SDS scenarios (Figure 6.7). The 10 mobile node density in Random scenario has a 760 ms packet delay, reducing to 610 ms when the speed was slowed by half, 481 ms at Speed/6, 79 ms at Speed/10, dropping to 36 ms at Speed Zero. Packet delays in 20 mobile node density decrease from 89 ms to 38 ms along with the node speeds. In the high node densities, 30 and 40 nodes, the packet delay is less than100 ms for high speeds and more than 40 ms for low speeds. The lower the packet delay and the node speeds, the higher the PDR.

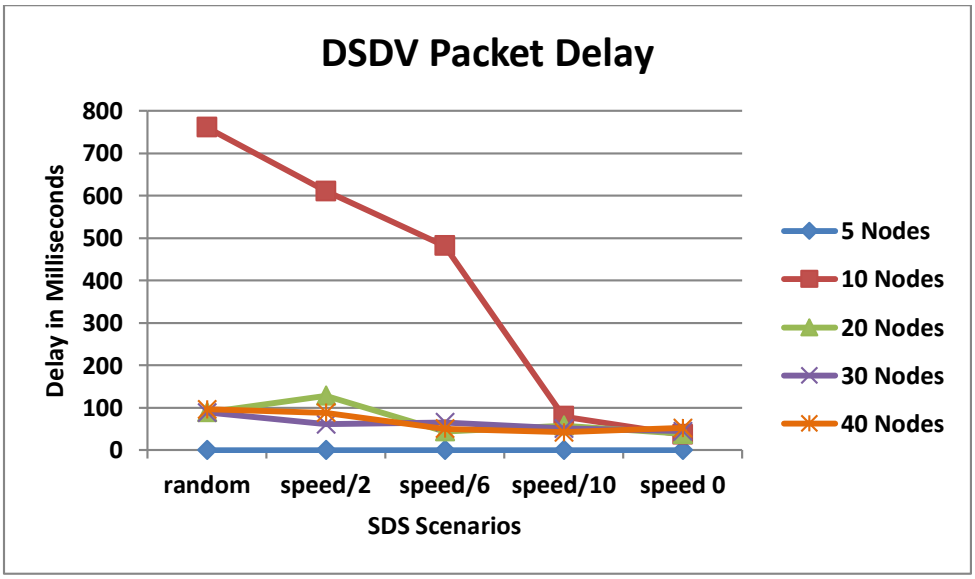

Figure 6.7: Packet Delay for SDS over DSDV

6.2.2.3 FPTD for SDS

Due to the proactive nature of DSDV, detecting forwarding paths in SDS experiments over DSDV takes a long time compared with the experiments over AODV (Figure 6.7). DSDV displays a poor ability to detect the routes quickly in high dynamic mobility networks and it takes more time in advertising good routes and recovering broken ones. At low mobile node density, five nodes, no forwarding paths have been detected by the DSDV routing protocol during the 500 second simulation run time (Figure 6.8).

In 10 mobile node density it took an average of 187 seconds to detect the first forwarding path, while it took 106 seconds to detect the first forwarding path in 20 mobile node density compared with 25 seconds for AODV with the same node density. The FPTD for high mobile node densities is 65 seconds for 30 nodes and 69 seconds for 40 nodes, which is around 60 seconds more than FPTD in AODV experiments.

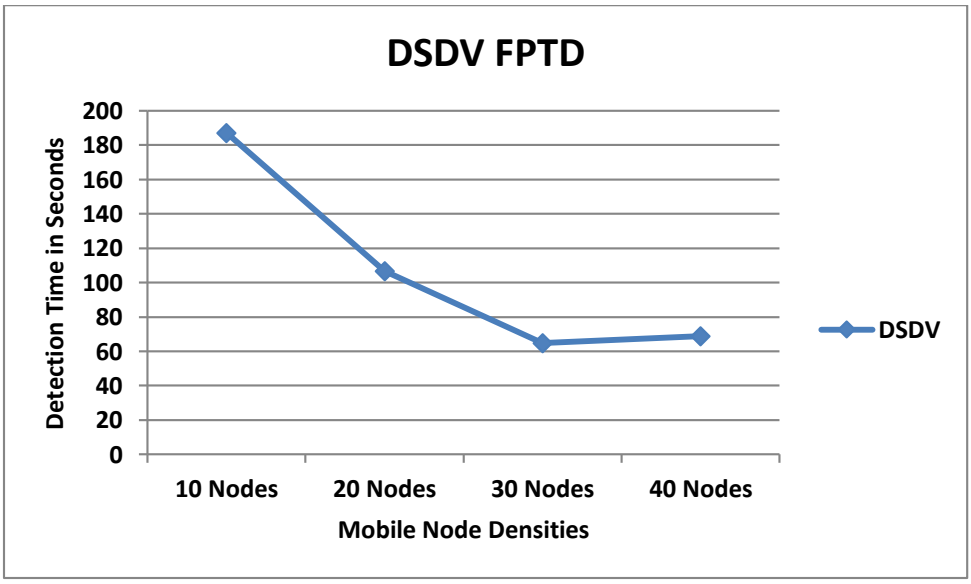

Figure 6.8: FPTD for SDS over DSDV

6.3 SDS Summary

The new proposed SDS algorithm is a dynamic-movement node-control paradigm for a self-organizing MANET. This algorithm takes advantage of the traffic condition in the network to control the forwarding nodes that create the mobile medium and uses it to transmit data packets between communication nodes. The control mechanism is based on slowing down the speed of the forwarding nodes in MANET, which increases the network performance. The lower the speed of the active forwarding nodes the higher the PDR. The implemented SDS algorithm improves the network performance for different mobile node densities running two different routing protocols, AODV and DSDV. The low mobile node densities show low PDR and increase relative to the increase in the node densities. In general, AODV experiments show a very good performance and give higher PDR than DSDV experiments. AODV routing protocol is a better choice for ad-hoc networks than DSDV due to the potential it has to deal with high dynamic mobility networks. The proactive DSDV routing protocol takes a long time to advertise the new routes and identify the broken ones. This delay causes DSDV a long time to detect the forwarding paths, which affects the performance of the network and reduces the PDR. On the other hand, the reactive AODV routing protocol is able to detect the forwarding paths in a noticeably shorter time compared with DSDV. For high mobile node densities, e.g. 40 nodes, AODV is able to detect a forwarding path at the first second of the simulation run time. This is because AODV has an efficient technique to establish and maintain routes quickly without any extra traffic for communication. The best mobile node density that shows the noteworthy benefit of the proposed SDS algorithm is a density of 20 nodes.

Chapter 7

Study of Ant Colony Optimization Movement Control

7.1 Ant System Node Control Algorithm

The new proposed Ant System Node Control algorithm, ASNC, is a Swarm Intelligence algorithm that aims to control the mobile node movement in MANET based on awareness of the traffic condition in the network. ASNC is adapted from the Ant System algorithm, which is one of the existing Ant Colony Optimization (ACO) algorithms used to control the random movement of mobile nodes. ASNC experiments simulated in NS2 utilized the RWP model in a restricted area of 1000x1000 meters. The mobile nodes, representing the ant agents, move toward random destinations at random speeds with an average speed of 4 m/s. In the experiments based on ASNC, two stationary nodes, the source and destination nodes located at (50,400) and (950,600) coordinates respectively, are communicating with each other using the CBR traffic generator over UDP. The experiments run over two different ad-hoc network routing protocols, AODV and DSDV, for five mobile node densities: 5, 10, 20, 30, and 40 nodes. Each mobile node density has been simulated three times for 500 second simulation run time and the average results taken.

- **ASNC Mechanism**

At the beginning of each experiment, the mobile node moves randomly to explore the simulation area without any control. The traffic of the network is observed during the entire simulation run time waiting for the utilized routing protocol to detect a forwarding path. The control mechanism starts when the first forwarding path is detected by the routing protocol. At this moment the information about the forwarding path becomes available

including the active forwarding nodes and their positions. Each forwarding node deposits a pheromone trail τ on its own position to mark it as a good place to be and to transmit data packets between the source and destination nodes. The amount of the deposited pheromone trail τ on a new forwarding position is initially equal to one. These pheromone trails demonstrate the forwarding path that has been used to transmit data packets between the communication nodes. For the ant agents the source node represents the ant colony while the destination node represents the food source in the simulation. The pheromone trail attracts other mobile nodes to move toward the forwarding postpositions where the pheromone is deposited rather than moving randomly. In order to move toward the pheromone trails each mobile node follows a local action based on the distance between its current position and the pheromone trail position as well as how far the trail is from the source and the destination. The closest pheromone trail position to the current position of the mobile node with the highest pheromone amount will be selected. The distance between each chosen position must be less than the transmission range of 250 m between the mobile nodes in NS2. To accomplish this condition a tabu list has been added to keep track of the visited forwarding positions by the mobile nodes along with the results of the obtained forwarding path as a solution. The tabu list was adapted from the Ant system algorithm to operate as memory capacity. When the mobile nodes reach the selected positions they pause for three seconds before they determine where to move next. The decision about where to move next during the simulation is taken by applying the probability rule, which is also adapted from the Ant System algorithm (Equation 4.5). In ASNC the values of the probability rule parameters for α, the impact of the trail, and β, the attractiveness, were set

to 0.5 while the distance value in the visibility $\eta_{ij} = \frac{1}{d_{ij}}$ (Equation 4.6) was calculated based on the distance between the communication nodes and the forward position. Changing the values of α and β in the experiments did not affect the PDR results.

- **Pheromone Trails**

The pheromone trails are not permanent; they evaporate over time from the more distant forward positions faster than those in forward positions closet to the source and the destination. A pheromone trail deposited on a forward position is modified when another mobile node moves toward this forward position, otherwise the pheromone evaporates over time. To update the pheromone trail τ the update equation (Equation 4.7) has been adapted from the Ant System algorithm where the evaporation coefficient ρ value is equal to 0.01 and the constant Q is equal to 1. Also, changing the values of Q and ρ did not affect the PDR result of the experiments. The update equation increases the pheromone amount on the visited forward positions and reduces and/or evaporates the pheromone amount on the unvisited forward positions. A new pheromone trail τ is deposited by the forwarding nodes each time the routing protocol detects a new forwarding path in MANET.

- **Problem Solution**

The mobile nodes visit several forwarding positions before they find the perfect positions to transmit packets between communication nodes. The proposed ASNC algorithm solves the problem of rapid disconnection in MANET due to the highly dynamic nature of the network. ASNC improves the network performance significantly by controlling the movement of the mobile nodes in a Swarm Intelligence mechanism based on awareness of the traffic condition.

7.2 Results and Analysis of ASNC

The experiments of the new proposed ASNC algorithm has been simulated in NS2 using two different ad-hoc routing protocols, AODV and DSDV. Each routing protocol has been tested with a density of 5, 10, 20, 30, and 40 mobile nodes then an average result for three runs has been taken for each. The experiment is conducted over a 500 second simulation run time and the performance of the MANET measured in terms of PDR, End-to-End Delay and FPTD.

7.2.1 Performance of ASNC over AODV

ASNC experiments result over AODV have been calculated to measure the performance of the new proposed technique to control the movements of mobile nodes in MANET. The results of the performance metrics PDR, End-to-End Delay and FPTD are illustrated in Sections 7.2.1.1, 7.2.1.2 and 7.2.1.3 respectively.

7.2.1.1 PDR for ASNC

Attracting the mobile nodes to move toward the best forward positions based on the deposited pheromone trails resulted in a significant increase in the delivery ratio in MANET, (Figure 7.1).

At a very low density with five mobile nodes no forwarding paths were detected by the routing protocol AODV during the entire simulation run time. As a result zero packets have been delivered in the experiments using five mobile node density. Increasing the mobile node density to 10 nodes shows a PDR of 26% compared with 4% in Random experiment. As in the SDS scenarios, the experiments with 20 mobile node density in ASNC shows the most interesting results and the best benefit of controlling the node movements based on the behavior of ants in nature.

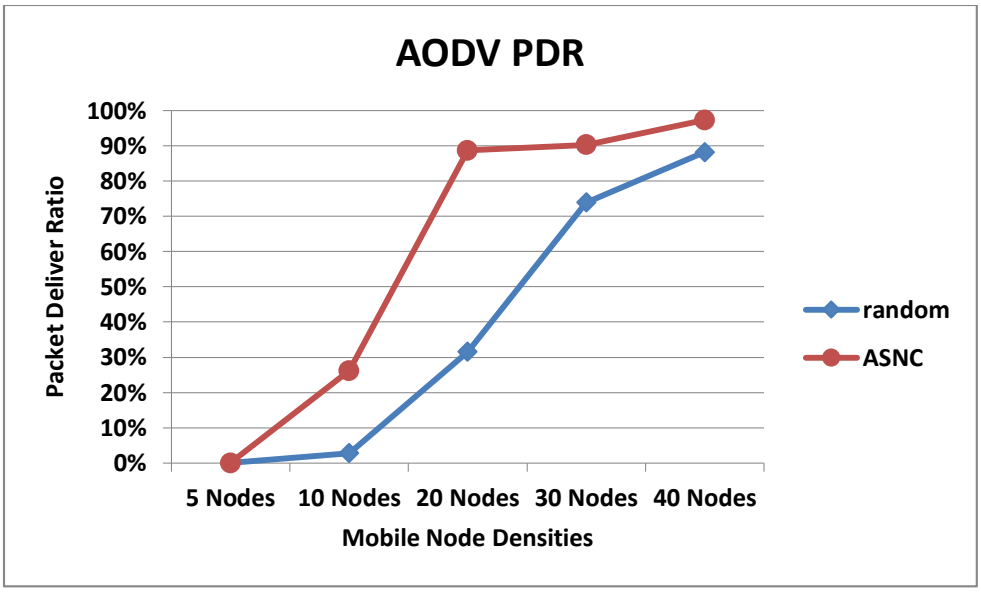

Figure 7.1: PDR for ASNC over AODV

In the 20 mobile nodes density 89% of the data packets have been successfully delivered to the destination resulting in an improvement of 57% over Random movement. The 30 and 40 mobile node densities, high densities, perform very well at any circumstances giving a delivery ratio of 90% and 97% in ASNC compared with 74% and 88% in the Random movement respectively. ASNC shows improvement for MANET even though the mobile nodes continued to move during the entire simulation run time.

7.2.1.2 End-to-End Delay for ASNC

In five mobile nodes density no packets have been delivered to the destination, which produces a zero packet delay during the whole 500 s of simulation run time, (Figure 7.2). The longest packet delay given by the 10 mobile node density drops from 786 ms in the Random scenario to 239 ms in ASNC.

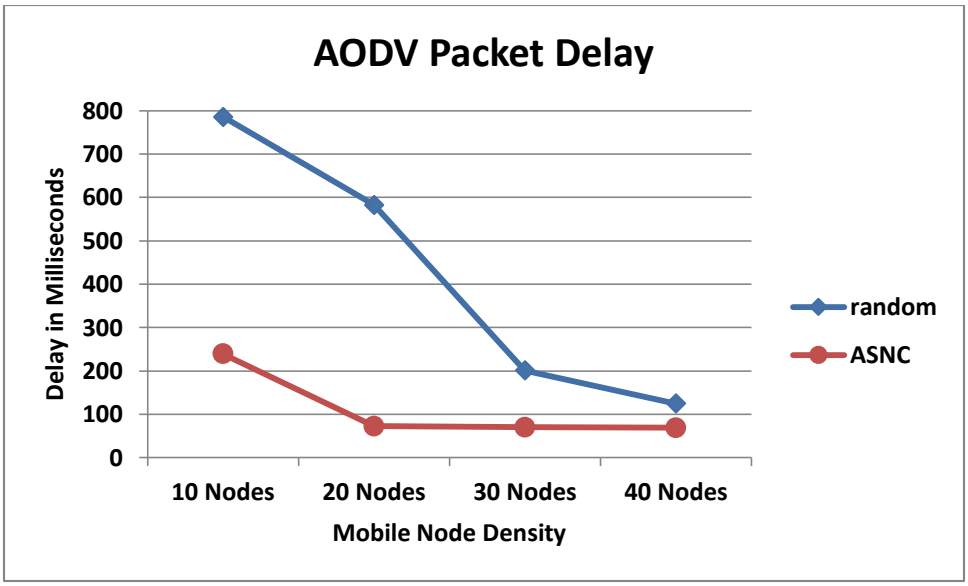

Figure 7.2: Packet Delay for ASNC over AODV

For 20 mobile node density the packet delay decreased to 73 ms corresponding with the increase in the PDR for 20 nodes compared with 583 ms in Random movement. This is followed by 70 ms and 69 ms, 201 ms and 125 ms in the Random scenario; for the high mobile node densities with 30 and 40 nodes respectively.

7.2.1.3 FPTD for ASNC

AODV routing protocol detected zero forwarding path during the whole 500 second of simulation time in the ASNC experiments of five mobile node density. For the experiments of 10 mobile nodes the average FPTD by AODV was 279 seconds, reduced to 87 seconds by increasing the number of the mobile nodes to 20, (Figure 7.3).

With high mobile node densities the AODV routing protocol is able to detect the first forwarding path in a short time, less than 10 seconds, which has improved the delivery ratio during the experiments.

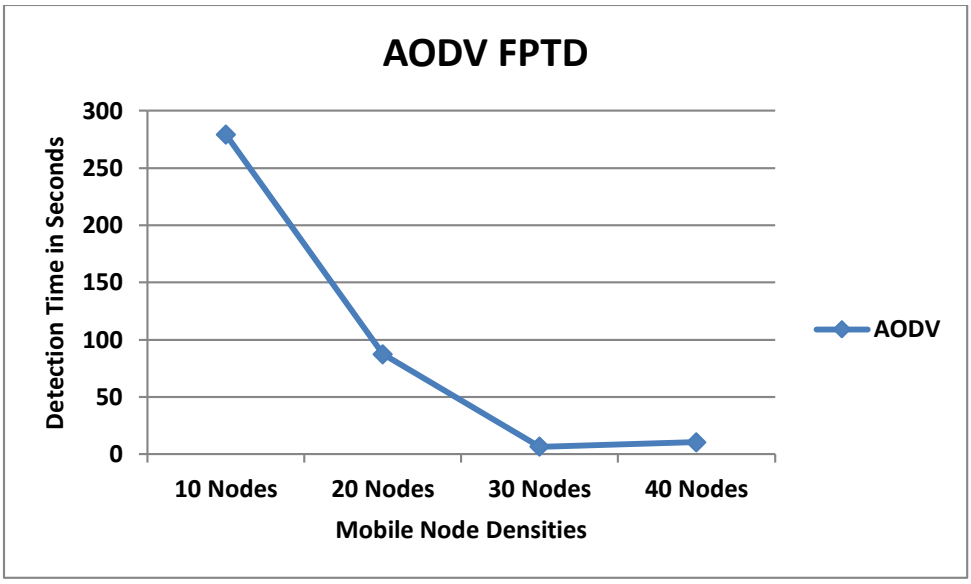

Figure 7.3: FPTD for ASNC over AODV

7.2.2 Performance of ASNC over DSDV

The performance of the new proposed ASNC algorithm over the proactive DSDV routing protocol is described in detail in Sections 7.2.2.1, 7.2.2.2 and 7.2.2.3. These sections contain the average results for PDR, End-to-End Delay and FPTD for five different mobile node densities. The results of these experiments show the improvement in MANET after controlling the movement of the nodes.

7.2.2.1 PDR for ASNC

The proposed ASNC has increased the delivery ratio by depositing a pheromone trail on the forwarding positions during the simulated experiments. These trails attract the mobile nodes to create forwarding paths from the source to the destination, resulting in carrying more packets and delivering them successfully during the simulation, (Figure 4).

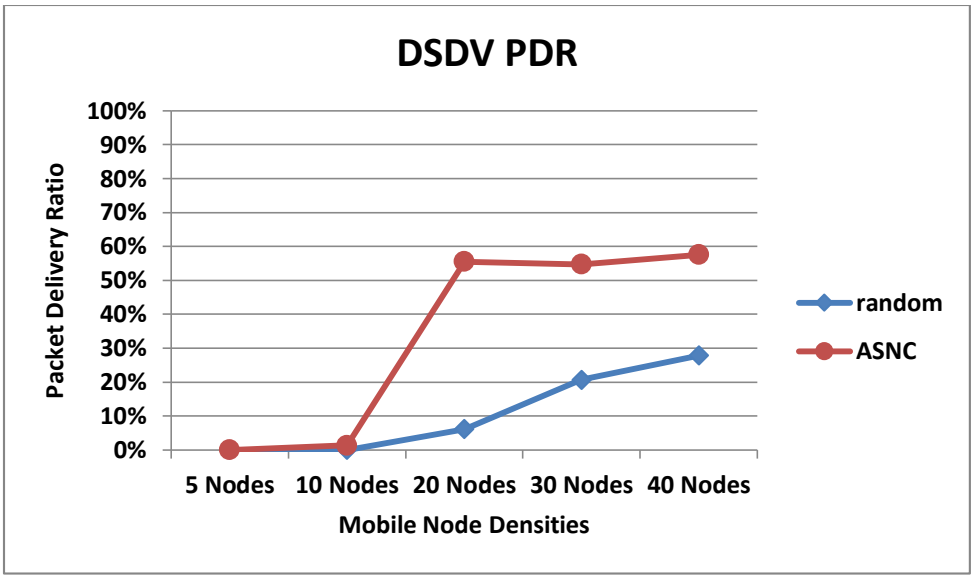

Figure 7.4: PDR for ASNC over DSDV

As expected no packet delivered in the very low mobile node density experiments. For 10 mobile nodes 1.3% of the total packets have been delivered, compared with 0.6% delivery ratio in the Random movement over DSDV routing protocol. These result changes by adding more mobile nodes to the experiments to become 20 mobile nodes rather than 10 nodes. The PDR reaches 55% during the experiments of 20 mobile node density almost 50% higher than Random movement. As in all the previous experiments using DSDV and AODV, 20 mobile node density shows the best performance of controlling the movements of mobile nodes in MANET. With 30 mobile node density, the PDR in Random movement of the nodes was 21% packets delivered, which improved to 55% in ASNC. The density of 40 mobile nodes have given a PDR of 58%, also higher than the Random scenario. In general, the PDR in ASNC over DSDV is still higher than the delivery ratio results in Random movement experiments. ASNC algorithm is preferred due to the fact that in ASNC the mobile nodes are moving during the entire simulation run time.

7.2.2.2 End-to-End Delay for ASNC

ASNC experiments over DSDV in MANET show zero packet delay for five mobile node density, as with all the other scenarios and experiments presented in this thesis. However, the packet delay for 10 mobile node density in ASNC has been improved to 208 ms compared with 760 ms in Random movement, (Figure 7.5).

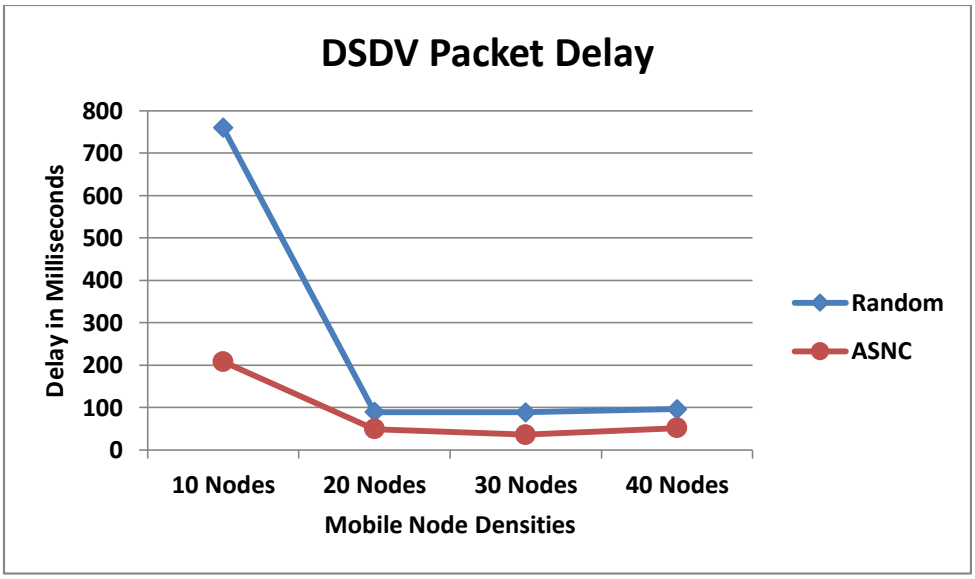

Figure 7.5: Packet Delay for ASNC over DSDV

Increasing the mobile node density to 20 nodes decreases the delay to 49 ms, which is half the result in Random movement. The packet delay in the high mobile node density of 30 and 40 nodes reach 39 ms and 50 ms respectively while in Random movement the results were above 80 ms.

7.2.2.3 FPTD for ASNC

Unlike AODV detecting a forwarding path in DSDV proactive routing protocol takes a considerable time. In ASNC experiments for 10 mobile node density it took DSDV 299 seconds to find a good route for transmitting data packets between the communication nodes, 20 seconds longer than the experiments over AODV (Figure 7.6).

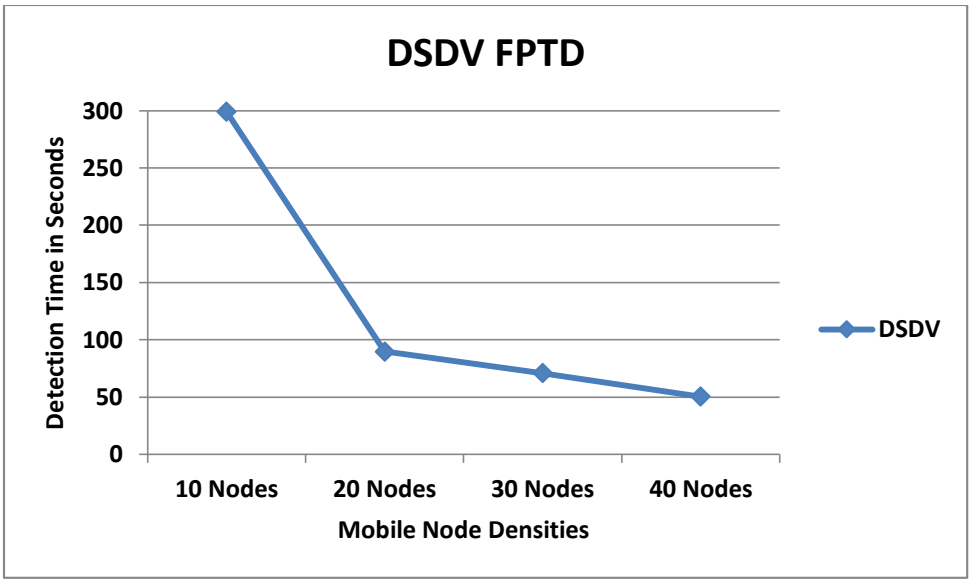

Figure 7.6: FPTD for ASNC over DSDV

The FPTD for 20 mobile node density utilizing DSDV was 90 seconds, followed by 71 seconds for 30 nodes and 50 seconds for 40 nodes. For ASNC over AODV experiments (Figure 7.3) the FPTD in 20 mobile nodes was 87 seconds, then reduced significantly for the high mobile node densities, showing a difference of at least 50 seconds compared to the DSDV results. DSDV delay in detecting forwarding paths affects the PDR and network performance. The experiments of five mobile nodes show zero forwarding paths during the entire 500 second simulation run time in both AODV and DSDV routing protocols.

7.3 ASNC Summary

The new proposed ASNC algorithm, Ant System Node Control, is an adaptive ACO algorithm that aims to control the movement of mobile nodes in MANET. The adapted algorithm is a complex self-organization paradigm implemented to control the mobile nodes to behave in the same way as a swarm of ants in nature. As in the SDS algorithm, the ASNC algorithm also observes the traffic condition in the network waiting for the right

moment to control the mobile nodes. The mobile nodes in ASNC represent the ant-agents; they leave a trail on their way, called pheromone, to attract other ant-agents to move toward the places they marked. These places are the forwarding positions in the network and moving toward these positions improves the network performance. The ASNC experiments are simulated for different mobile node densities over AODV and DSDV routing protocols. The ASNC PDR results increased significantly in all the mobile densities along with the number of mobile nodes. The low mobile node densities show a low delivery ratio compared with the PDR results for the high mobile node densities in both AODV and DSDV. Moreover, the 20 mobile node experiment demonstrates the best benefit of using ASNC in MANET to control the movement of the mobile nodes. In ASNC, the experiments running over AODV routing protocol perform better than the DSDV routing protocol experiments. Definitely, AODV is the choice for a high dynamic network such as MANET due to the quick discovery and maintenance of the routes. AODV was able to detect the first forwarding path in the experiment with high mobile node densities, in less than 10 seconds, while it took DSDV between 50 and 70 seconds. The performance of ASNC in MANET over AODV shows delivery ratio results of 10% to 50% more than in the experiments with Random movement. DSDV experiments also show performance ranging between 20-50 percent higher than the original movement of the mobile nodes without control in Random movement. ASNC improves the network performance and keeps the mobile nodes moving during the entire simulation.

Chapter 8

Comparison of Slow Down Speed versus Ant System Node Control Algorithms

8.1 SDS VS. ASNC

In this thesis, two new dynamic-movement node-control algorithms are proposed to control the mobile nodes in MANET according to awareness of the traffic condition. The proposed algorithms are the complex self-organization algorithm, Ant System Node Control (ASNC), adapted from the Swarm Intelligence Ant Colony Optimization algorithm, ACO, and the Slow Down Speed (SDS) algorithm, designed based on a simple self-organization mechanism. During the experiment studies of the new proposed algorithms on different mobile node densities, the 20 mobile node density experiments demonstrated the most benefit of the dynamic-movement node-control in MANET. The interesting case of 20 mobile node density was chosen for comparison purposes between the performance of ASNC and SDS over AODV and DSDV routing protocols. A movement file for 20 mobile nodes was generated using the Setdest tool to define the node positions, speeds and directions in NS2, with zero pause time and average speeds of 4 m/s for 500 s simulation run time in a 1000x1000 meter simulation area. To avoid any randomness in the comparison between ASNC and SDS over AODV and DSDV, the generated movement file of 20 mobile node density was used in all the experiments. The performance of ASNC and SDS in MANET have been compared over different performance metrics including the PDR, distance covered, active forwarding nodes, End-to-End Delay, and FPTD in addition to pheromone maps.

8.1.1 Performance of SDS VS. ASNC over AODV

Section 8.1.1.1 shows the delivery ratio results for ASNC and the five SDS scenarios including the Random movement scenario, followed by Section 8.1.1.2 that shows the distance covered by the mobile nodes in the ASNC and SDS experiments, with the active forwarding nodes that participated in forwarding packets presented in Section 8.1.1.3. The next Section (8.1.1.4) demonstrates pheromone maps for ASNC over the 500 second simulation run time. The packet delay results are presented in Section 8.1.1.5 and the FPTD in Section 8.1.1.6. These experiment results show the performance of SDS versus ASNC in MANET for 20 mobile node density over AODV routing protocol using the same movement file generated by Setdest tool.

8.1.1.1 PDR for SDS VS. ASNC

Controlling the mobile node movements in MANET using both the adapted ACO algorithm, ASNC, and the self-organization algorithm, SDS, over AODV routing protocol increases the packet delivery ratio, (Figure 8.1).

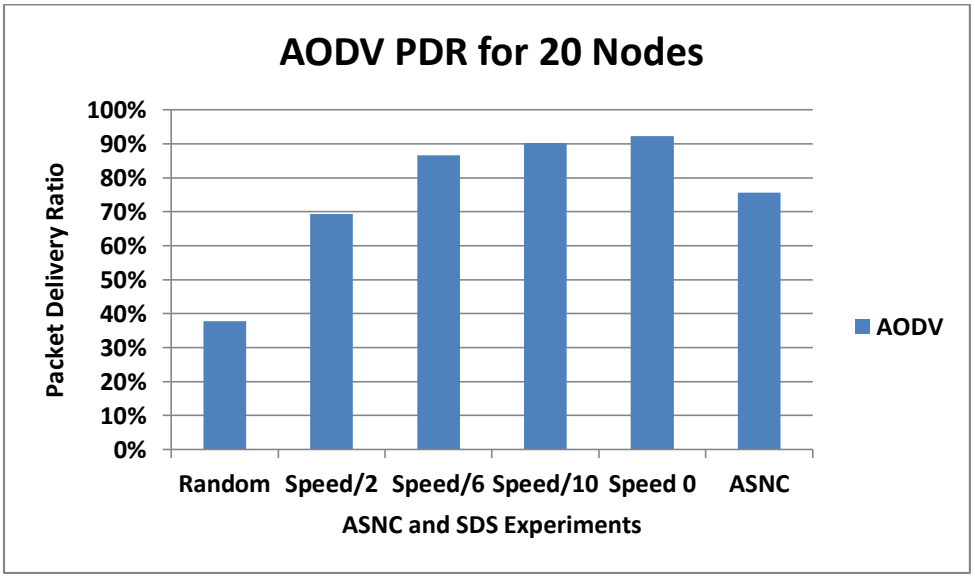

Figure 8.1: PDR Comparison over AODV

In the original node-movement experiment without control, the Random scenario, the PDR was 38% and increased with the slowing down of the forwarding nodes speed by half to reach 69%. The ASNC experiment shows double the result of the Random scenario, 76% delivery ratio, only 10% less than the PDR in the Speed/6 scenario. In slowing down the Speed/10, the average speed of the forwarding nodes is 0.4 m/s or less, while stopping the forwarding nodes in the Speed Zero scenario resulted in delivery ratios of 90% and 92% respectively. Even though slowing down the forwarding node speed in the network shows high PDR, the forwarding nodes are actually stopping or almost not moving, with speeds of less than 0.5 m/s.

Figure 8.2 shows the number of packets delivered at 30 second intervals, for the AODV routing protocol with a density of 20 nodes.

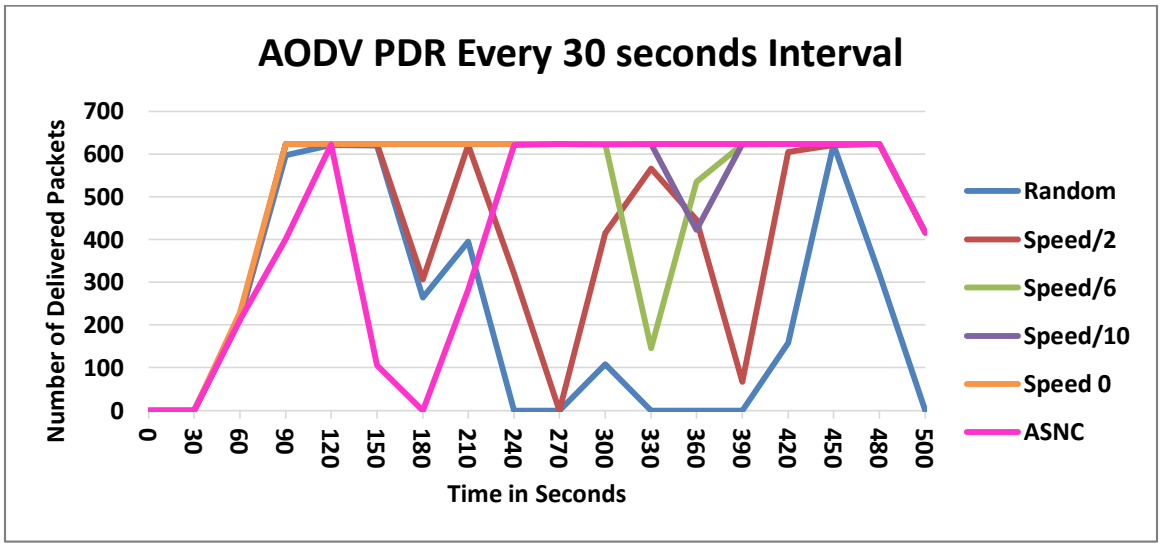

Figure 8.2: PDR every 30 seconds over AODV

There were no packets transmitted during the first 39 seconds, until the AODV protocol detected the first forwarding path. For the Random movement scenario, the maximum number of packets delivered was reached immediately (623) as there was a forwarding path

established between the source and the destination. However, due to the random movement of the mobile nodes, this path disconnected, so the number of packets delivered dropped significantly until it reached zero at 240 seconds. The number of packets delivered during the Random movement experiment remained relatively low by comparison with the other scenarios.

When the speeds are slower, the number of packets delivered are higher than the random movement, with 1/10 slowed speed having better results of packet delivery than 1/6, and both were better than 1/2. At Speed Zero, the forwarding nodes were not moving, which maintained the forwarding path unchanged resulting in delivering the maximum number of packets in 30 seconds (623) as soon as the forwarding path was detected at 39 seconds, and remained there throughout the simulation run time.

For the ASNC experiment, the packet started to reach the destination at 39 seconds when the first forwarding path was detected by the AODV routing protocol. The maximum number of packets (623) was delivered at the 120 second point, and then dropped at 180 seconds due to the ant-agent (mobile node) search movements for the best forwarding positions in the network. The maximum number of packets delivered was reached again at 240 seconds when the ant-agents took their forwarding places between the communication nodes, and remained moving from forwarding positions to other forwarding positions for the duration of the simulation run time.

8.1.1.2 Distance Covered in SDS VS. ASNC

The distance covered in the simulation using AODV routing protocol for ASNC and SDS (Figure 8.3) shows the average distance covered at intervals of 10 seconds by 18 mobile nodes in the 20 mobile node density that includes two stationary communication nodes.

The distance covered by the 18 mobile nodes was calculated for each experiment over a 500 s simulation run time. For the first 38 seconds all the experiments demonstrated random movement of the 18 mobile nodes. The action of controlling the movement of the nodes by ASNC and SDS algorithms was initiated at 39 seconds when the AODV routing protocol detected the first forwarding path.

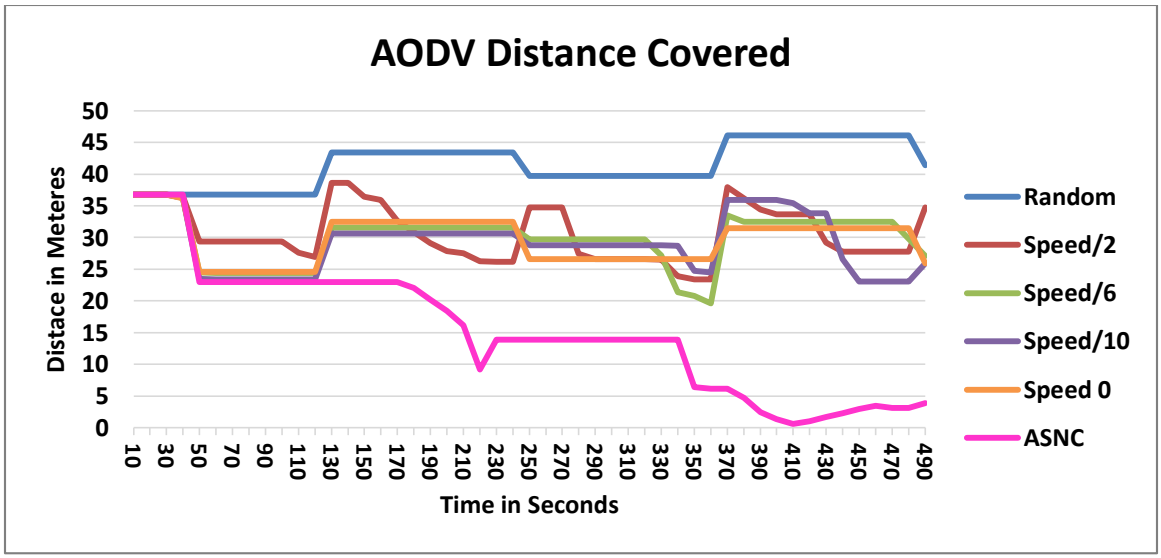

Figure 8.3: Distance Covered over AODV

For the Random scenario without any node-movement control, the 18 mobile nodes covered large distances with an average speed of 4m/s. In the first 120 seconds, the average distance covered by the 18 mobile nodes was 37 meters per 10 seconds. The following 120 seconds showed an average distance covered of 44 m, then 40 m for the next 120 seconds, and finally 46 m for the last 120 seconds.

The distance covered by the Speed/2, Speed/6 and Speed/10 scenarios during the first 38 seconds is equal to the distance covered by the Random scenario, 37 m, because at this point the 18 mobile nodes were still moving randomly without control. After detecting the first forwarding path at 39 seconds, the distance covered decreased along with the speed of

the forwarding nodes to range between 20 to 37 meters. Slowing down the speed of the forwarding nodes to zero shows a similar line with curves to the Random scenario with less distance covered. That is because in the Speed Zero scenario at 39 seconds at least four forwarding nodes between the communication nodes stop moving until the end of the simulation experiment, showing a result of a decrease in the distance covered almost by 15 m. The rest of the mobile nodes continue moving randomly with the same directions and speeds as the mobile nodes in the Random scenario.

For ASNC, in the first 10 seconds after the routing table detected the first forwarding path the distance covered by the 18 mobile nodes dropped from 37 to 23 m, then dropped to 9 m at the 220 second point, which proves that the mobile nodes are moving closer to the communication nodes. The distance covered then rose to 13 m, moving slightly further during the search for the best forwarding positions, where it stayed until the 340 second point. Again the distance covered by the mobile nodes dropped to 0.5 m at 410 seconds, and ended up at 4 m at the end of the simulation run time. These results show the most efficient movement control as the distance covered was more direct and restricted in a small area.

8.1.1.3 Forwarding Nodes in SDS VS. ASNC

Figure 8.4 illustrates the active forwarding nodes in the ASNC experiment over MANET using AODV routing protocol for 20 mobile node density, while Figure 8.5 illustrates the active forwarding nodes at Speed Zero, the SDS scenario with the highest PDR. Node zero is the source where node one is the destination in the simulated MANET. Both of the experiments used the same movement file generated by the Setdest tool and used in all experiments outlined in this chapter.

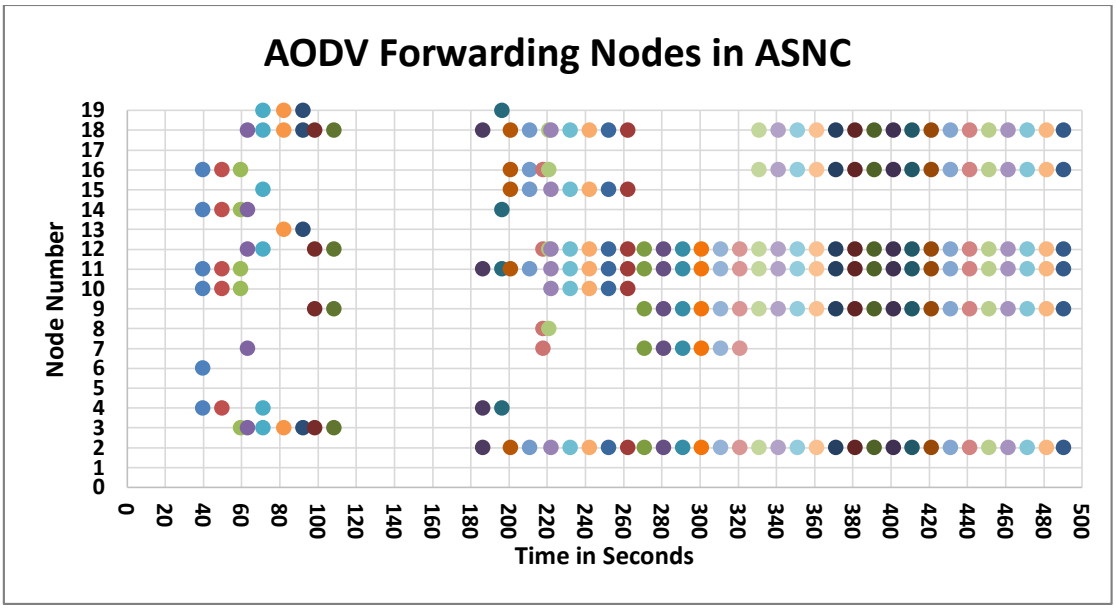

Figure 8.4: Active Forwarding Nodes in ASNC over AODV

The active forwarding nodes in ASNC (Figure 8.4) start showing up at 39 seconds with the detection of the first forwarding path. The first mobile nodes that participated in forwarding packets from the source to the destination were nodes 4, 6, 10, 11, 14, and 16. These forwarding nodes deposited a pheromone trail to attract the other mobile nodes. The remaining mobile nodes started moving towards the forwarding positions where the pheromone trails were deposited. The mobile nodes keep moving from one position to another, causing changes in the routes of the routing table and in the forwarding nodes that have been actively transmitting the data packets. Each forwarding path has an expired time and lasts for only 10 seconds. The path is maintained by the AODV routing protocol and replaced with a new path if it is broken; otherwise the forwarding path remains for a longer period. During the next 10 seconds, the active forwarding nodes changed to 4, 10, 11, 14, and 16, as node six moved out of the transmission range due to the node search-movements in the network. The mobile nodes continued to move between the forwarding positions

making new forwarding paths and ensuring connectivity between the communication nodes. At 110 seconds the network lost the connection due to the movement of the nodes; however, when the mobile nodes took a place on the forwarding position at 186 seconds the connection was restored. This period of time, 110-186 seconds, explains why the packet delivery ratio dropped in the ASNC experiment (Figure 8.5). The connection between the source and the destination remained for the rest of the simulation run time and at 330 seconds the forwarding path was optimized by the algorithm with active forwarding nodes 2, 9, 11, 12, 16, and 18, which remained the same until the end of the experiment at 500 seconds. In general, all the mobile nodes from node 2 to node 19, except nodes 5 and 17, participated in forwarding packets during the 500 seconds simulation run time.

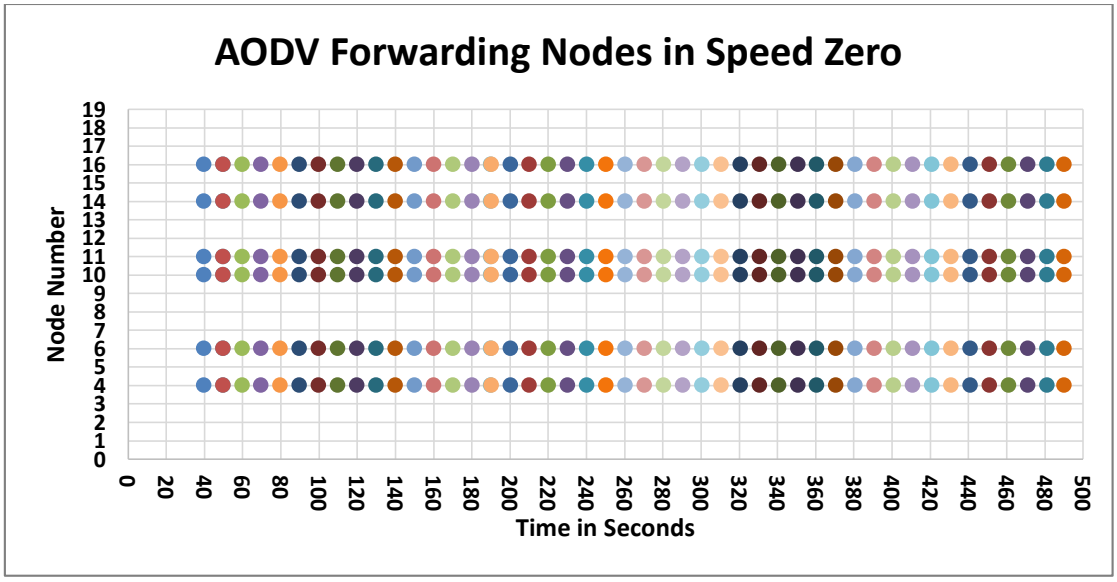

Figure 8.5: Active Forwarding Nodes in Speed Zero over AODV

Figure 8.5 shows the active forwarding nodes in Speed Zero scenario where the nodes that are participating in forwarding the data packets to the destination actually stopped moving. The first 38 seconds shows zero forwarding nodes until the AODV routing protocol detected the first forwarding path at 39 seconds. The nodes 4, 6, 10, 11, 14, and 16 are the

active forwarding nodes that create the first forwarding path, similar to the first active forwarding nodes in the ASNC experiment, both using the same movement file, (Figure 8.4). Because of the self-organization movement-control in Speed Zero scenario, the speed of the active forwarding nodes 4, 6, 10, 11, 14, and 16 reduced to zero, resulting in stopping the forwarding nodes from moving anywhere and remaining at the same positions until the end of the simulation run time. The other mobile nodes continue moving randomly in the network without affecting the forwarding path or changing the routing tables. The reactive nature of AODV routing protocol identifies a new route only when it is needed, otherwise the route remains unchanged, which is what happened in Speed Zero scenario. As shown in Figure 8.5 only six nodes participated in transmitting the data packets during the entire simulation run time, while in ASNC (Figure 8.4) the mobile nodes were moving all the time and most participated in forwarding data packets at various times.

8.1.1.4 ASNC Pheromone Map

In order to visualize the pheromone positions and amounts in ASNC experiment of 20 mobile node density an X Y bubble-scatter map is used (Figure 8.6). The map is divided into 100 meter squares representing the 1000x1000 meter simulation area; the 1100 and 1200 meter points on the X-axis are additional space to show the pheromone bubbles, during the 500 second simulation run time. For ASNC algorithm, at the beginning of the simulation run the source and destination communication nodes deposit a pheromone trail on their locations initially equal to one, (Figure 8.6). The source and destination are stationary, their deposited pheromones are constant, and remain until the end of the simulation run time with the same amount. Each bubble in the map represents the X and Y coordinate of the total pheromone amount deposited by different forwarding nodes in a

square of 100x100 meters. The radius of the bubble indicates the amount of the pheromone, the higher the pheromone amount the larger the bubble. At time zero, no forwarding path had been detected yet by AODV routing protocol and the only deposited pheromone in the area were the source and the destination pheromone trails.

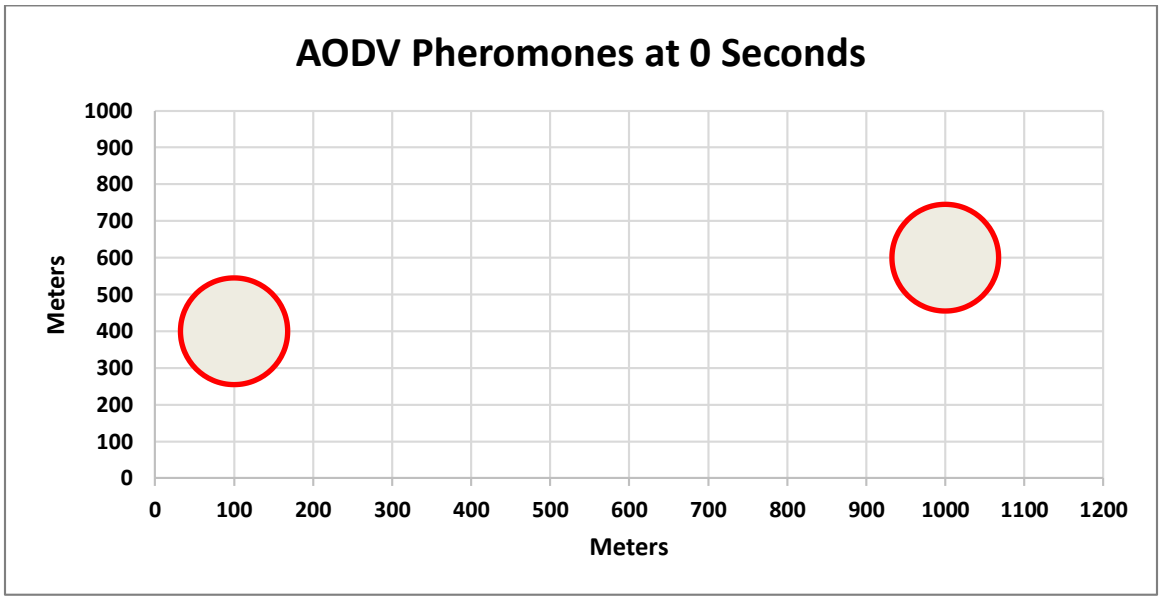

Figure 8.6: ASNC Pheromone Map over AODV at Time Zero

After 50 seconds of the simulation run time, AODV routing protocol was able to detect the first forwarding path at 39 seconds and the forwarding nodes deposited a pheromone trails on the forwarding positions (Figure 8.7). The deposited pheromone trails are located in an area between the source and the destination with a distance less than the transmission range of 250 meters between each pheromone position.

The deposited pheromone trail attracts other mobile nodes to move toward these forwarding positions, resulting in detection of other forwarding paths between the communication nodes and deposit of more pheromone trails on the newly discovered forwarding positions.

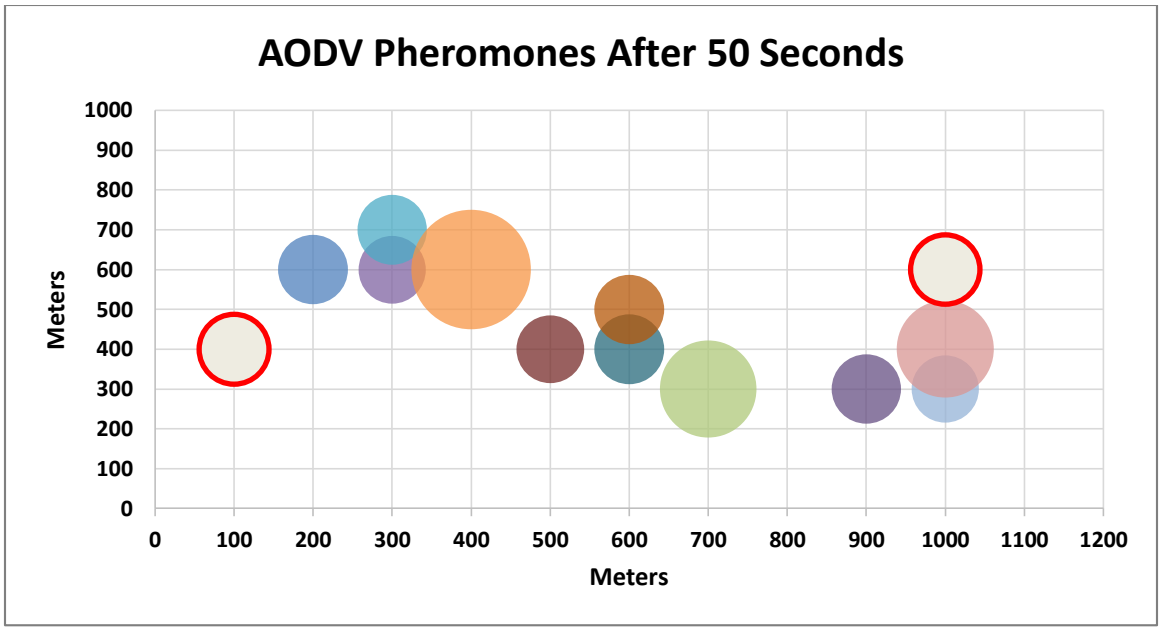

Figure 8.7: ASNC Pheromone Map over AODV at Time 50

The pheromone trail is initialized at one but is subject to update if another mobile node decides to move toward this forwarding position, otherwise the pheromone trail evaporates over time. The forwarding positions with the lowest pheromone amounts are located at (300,600) and (500,400) with a pheromone trail of 0.94, followed by the next lowest pheromone trail in the map, 0.95, located at (1000, 300). These pheromone trails evaporate over time and decrease from 1 to 0.94 and 0.95 in the total of the deposited pheromones within 100 meter squares. Five bubbles, located at (200, 600), (300, 700), (600, 400), (600, 500) and (900, 300) in the map, represent pheromone positions with amounts of 1. Two other pheromone amounts of 1.94 were located at (700, 300) and (1000, 400) coordinates. The largest amount of pheromone, 2.94, is found in the forwarding position located at (400, 600), in addition to the source and the destination bubbles with pheromone amounts of 1 which last until the end of the simulation. After the pheromone trails are deposited, the

mobile node tends to move in a rectanglular area between 200-1000 on the X-axis and 200-800 on the Y-axis.

After 100 seconds of the simulation run time, new bubbles showed up in the pheromone map for ASNC over AODV, (Figure 8.8). Moreover, some bubbles became larger and other bubbles became smaller due to the update and evaporation operations, except for the source and the destination.

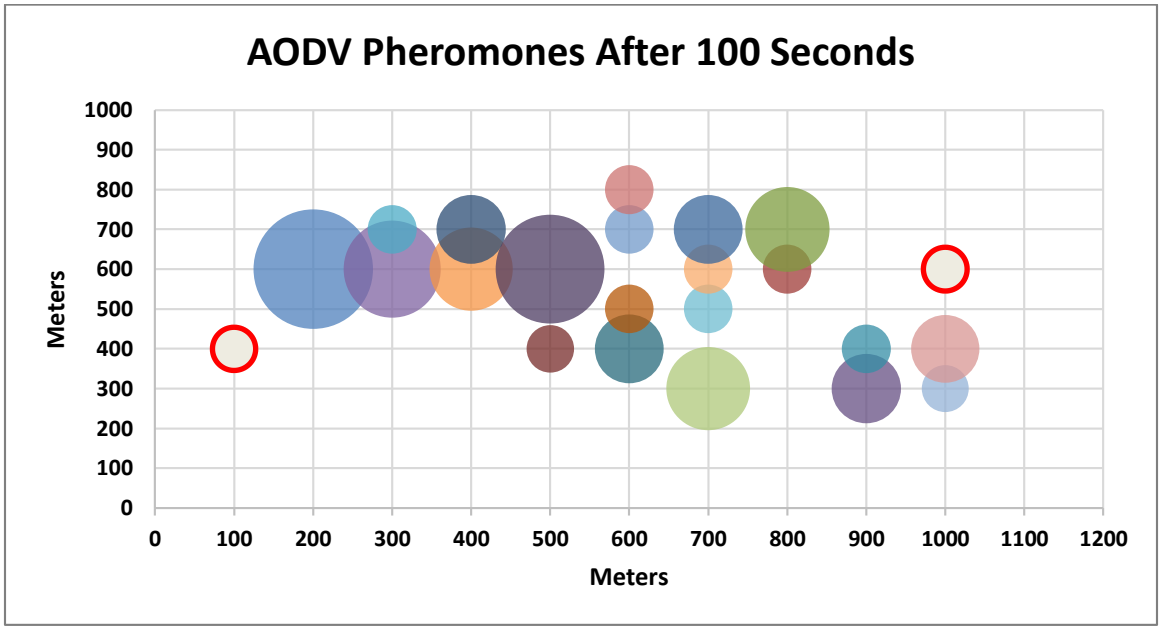

Figure 8.8: ASNC Pheromone Map over AODV at Time 100

Some of the existing pheromone trails in the first 50 seconds of the simulation run did not attract significant numbers of mobile nodes compared with other pheromone trails in the map. By the time that the simulation reached 100 seconds, these pheromone trails evaporated from their forwarding positions, while the other trails that attracted more mobile nodes increased their pheromone amounts. On the other hand, ten new pheromone trails deposited by the forwarding nodes showed on the map located at (400, 700), (500, 600),

(600, 700), (600, 800), (700, 500), (700, 600), (700,700), (800, 600), (800, 700), and (900, 400) representing new forwarding positions with different pheromone amounts.

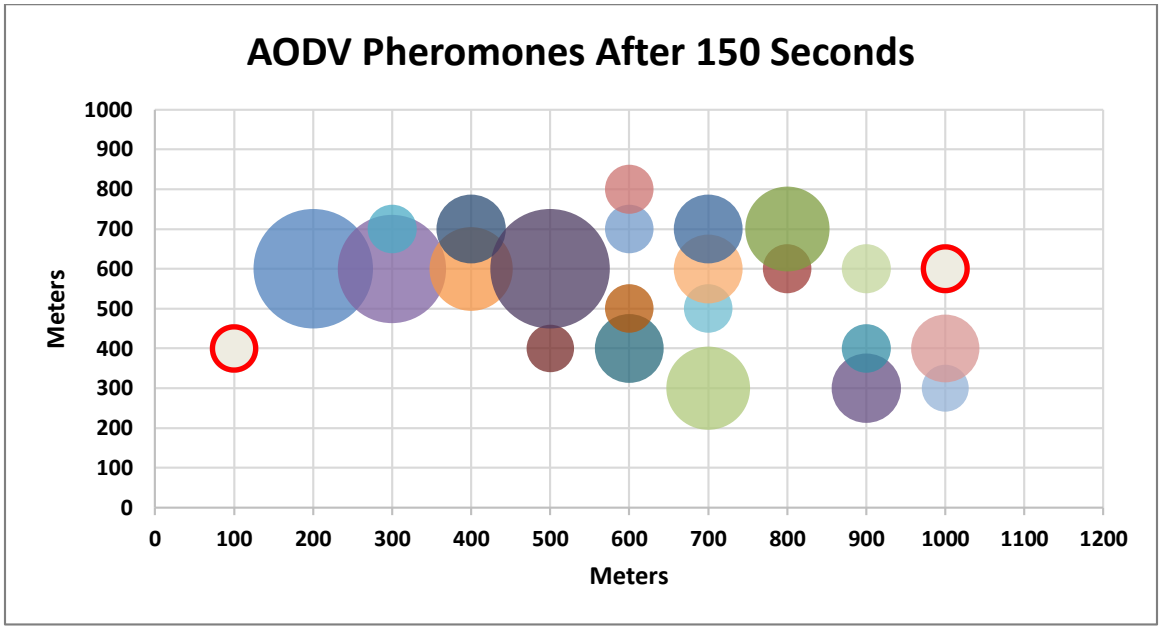

Figure 8.9: ASNC Pheromone Map over AODV at Time 150

The mobile nodes in MANET keep moving from one forwarding position to another searching the simulation area for the best forwarding positions to transmit the data packets between the communication nodes. The pheromone trail changes over the simulation run time, increasing at the positions with high attracting levels and decreasing at the unvisited forwarding positions, along with newly-deposited pheromone trails that showed up on the map of ASNC algorithm every time AODV detected a new forwarding path, (Figures 8.9, 8.10 and 8.11).

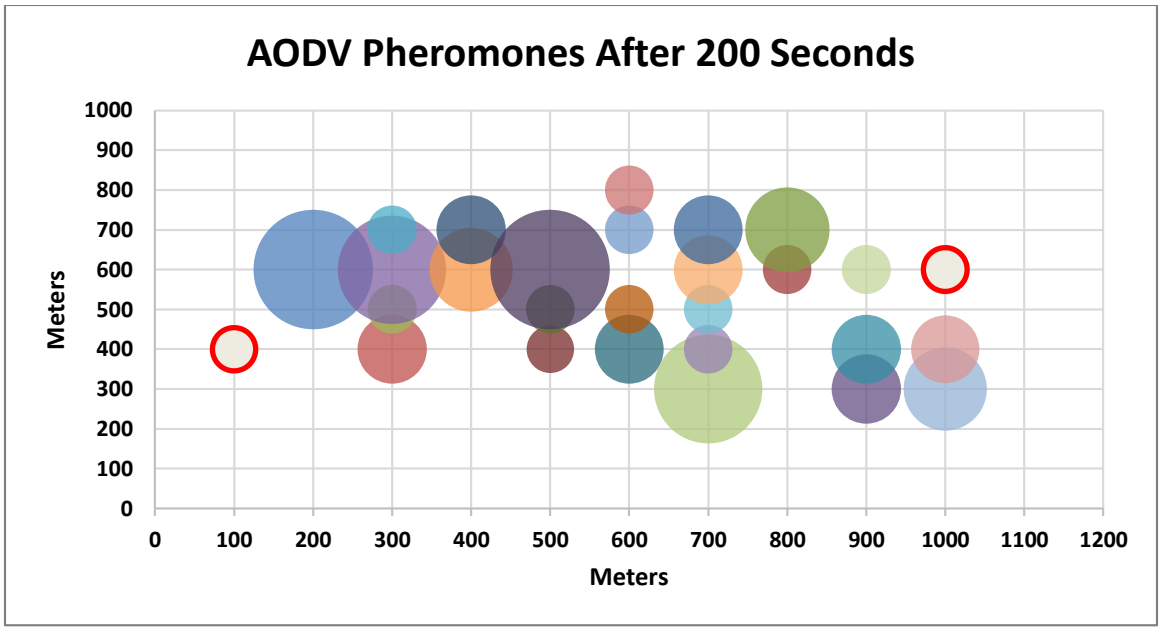

Figure 8.10: ASNC Pheromone Map over AODV at Time 200

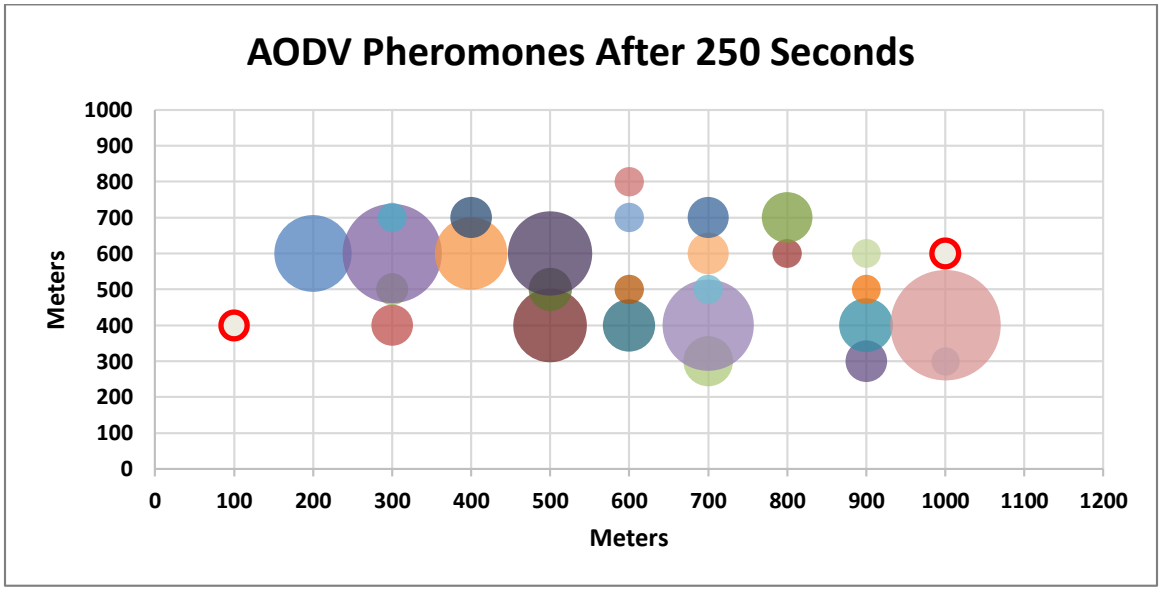

Figure 8.11: ASNC Pheromone Map over AODV at Time 250

For example, the pheromone trails located at (300, 400) and (300, 500) did not exist at 150 seconds (Figure 8.9) but were deposited later at a time between 151 and 200 seconds, (Figure 8.10). At 250 seconds, different pheromone trails started evaporating (Figure 8.11)

including the trails deposited at (300, 500) , (600, 500), (600, 700), (600, 800), and (1000, 300) on the map.

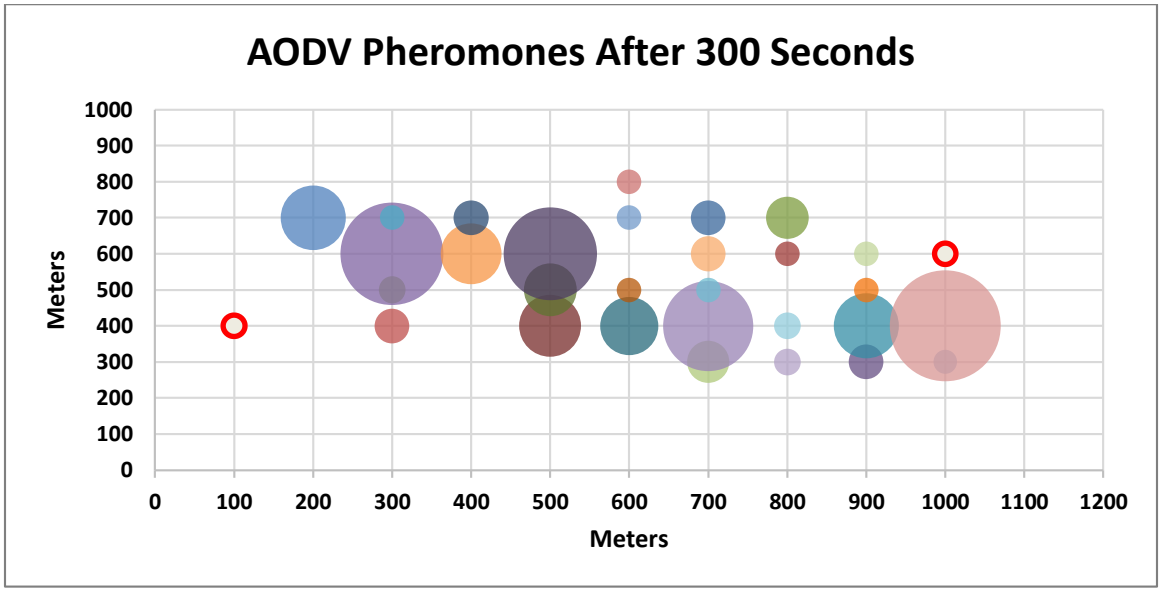

Figure 8.12: ASNC Pheromone Map over AODV at Time 300

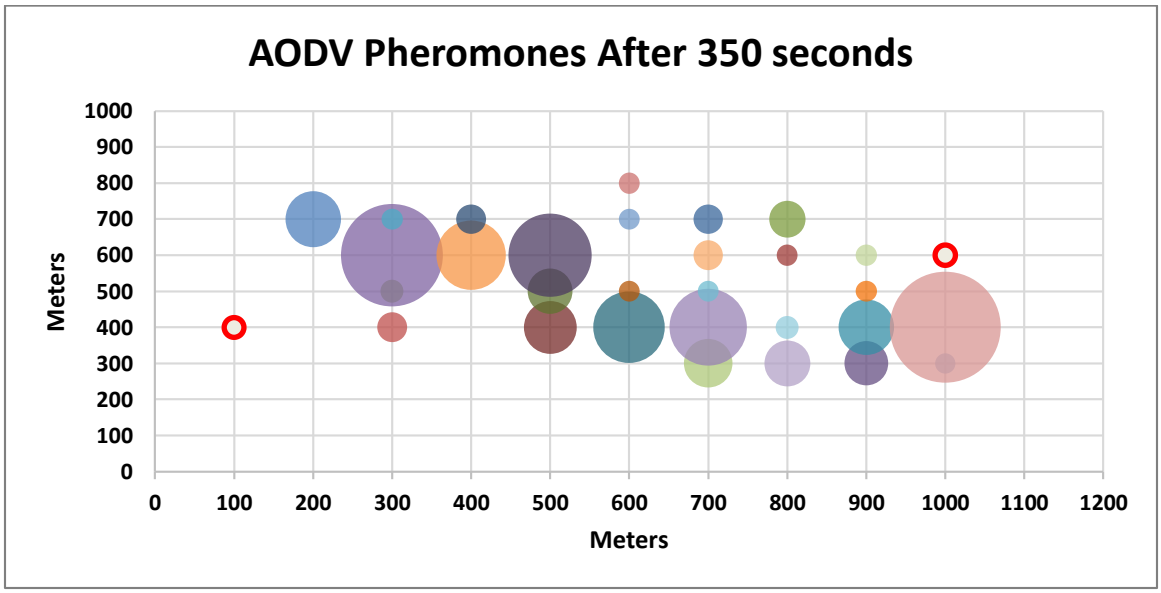

Figure 8.13: ASNC Pheromone Map over AODV at Time 350

At 300 seconds, the favorite forwarding positions become more obvious to the mobile nodes in MANET and the pheromone amount in some of these positions increased to reach

17 in total, (Figure 8.12). Moreover, the pheromone trails located at (200, 600), (300, 600), (400, 600), (500, 600), (600, 400), (700, 400), (900, 400), and (1000, 400) on the map have the biggest bubble size and the highest level of the pheromones.

The pheromone trails on the favorite forwarding positions attract the mobile nodes to update the pheromone amounts by adding more pheromone, (Figures 8.13 and 8.14). However, the unvisited forwarding positions, e.g. distant positions, continue to evaporate and only reached a level of 0.4 over the simulation run time.

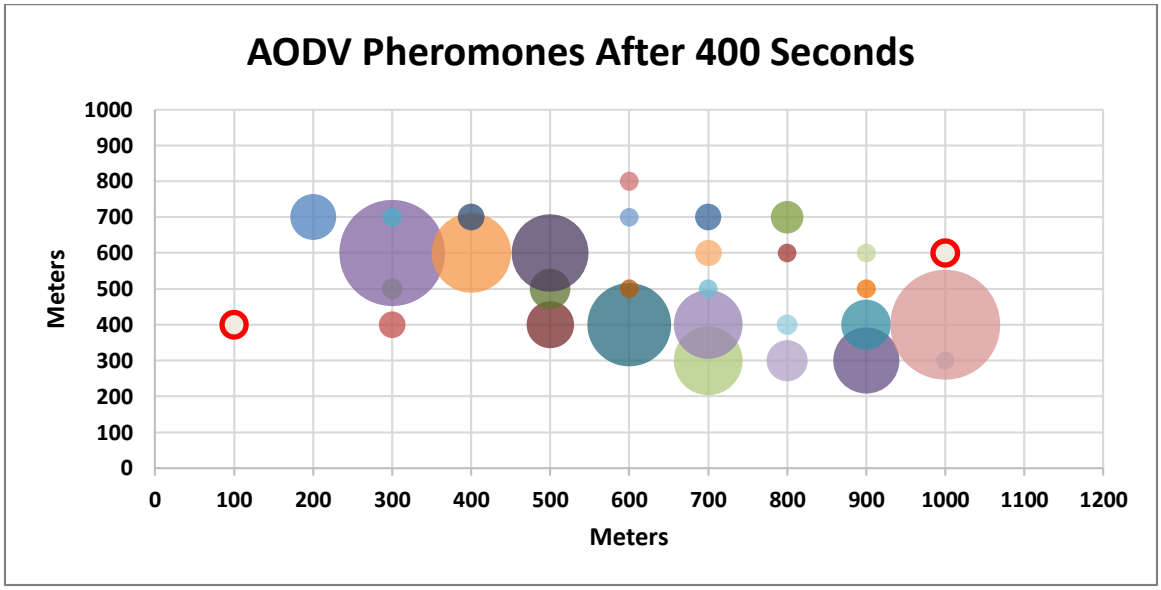

Figure 8.14: ASNC Pheromone Map over AODV at Time 400

In the final 100 seconds of the simulation run time, most of the pheromone trails evaporated, illustrated on the ASNC map as small bubbles depicting the remaining amount of pheromones (Figures 8.15 and 8.16). At this point, the mobile nodes know where to move and what place is their favorite based on the pheromone amounts that had been deposited on the forwarding positions.

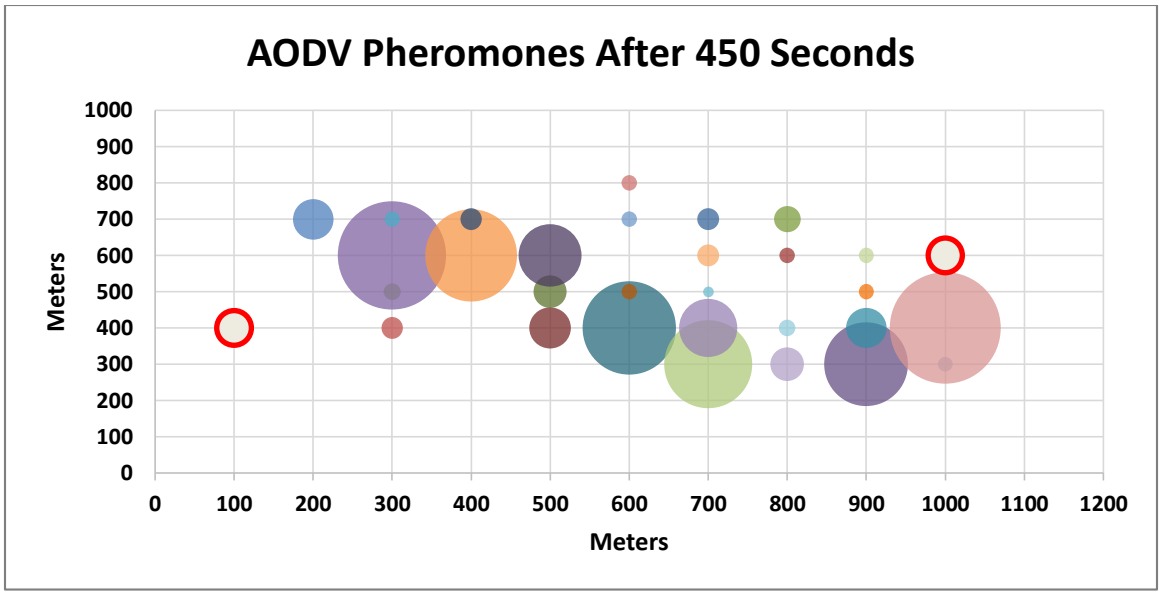

Figure 8.15: ASNC Pheromone Map over AODV at Time 450

As shown in Figures 8.15 and 8.16, during the simulation the mobile nodes prefer to be at six different forwarding positions in the simulation area, located at (300, 600), (400, 600), (600, 400), (700, 300), (900, 300), and finally (1000, 400). These positions have the largest bubbles on the map, showing the highest amount of pheromone located in a communication curve connecting the source and the destination. This curve starts to be obvious after 350 seconds of the simulation run time (Figure 8.13). As shown in Figures 8.14 and 8.15 during the next 100 seconds, the other pheromone trails evaporated quickly making the curve more clear.

The last 100 seconds definitely show that this is the forwarding path where the mobile nodes prefer to be a part as active forwarding nodes (Figures 8.15 and 8.16). The six participating forwarding nodes, 2, 9, 11, 12, 16 and 18 in ASNC at 330 seconds, shown in Figure 8.4, proves the observation about the six preferred forwarding positions in the pheromone map (Figure 8.16).

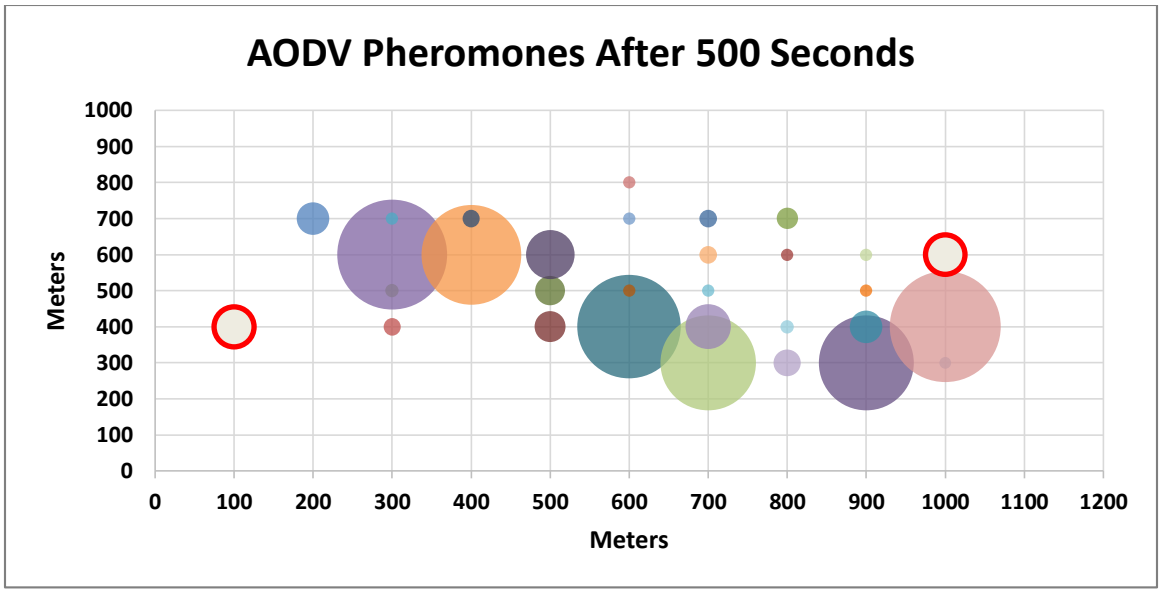

Figure 8.16: ASNC Pheromone Map over AODV at Time 500

8.1.1.5 End-to-End Delay for SDS VS. ASNC

Figure 8.17 shows the packet delay for 20 mobile node density in different experiments using AODV routing protocol. The experiments share the same movement file generated by the Setdest tool for comparison purposes.

The Random movement experiment shows the longest packet delay of 203 ms. By slowing down the speed to half in the next experiment, the packet delay reduced relatively to the reduction of the speed. For Speed/6, Speed/10 and Speed 0 the packet delay also reduced significantly with the reduction of the speed, and range between 75 and 68 ms. The packet delay for ASNC experiment shows a 90 ms delay less than half the delay for Random movement and in between the delays of Speed/2 and Speed/6.

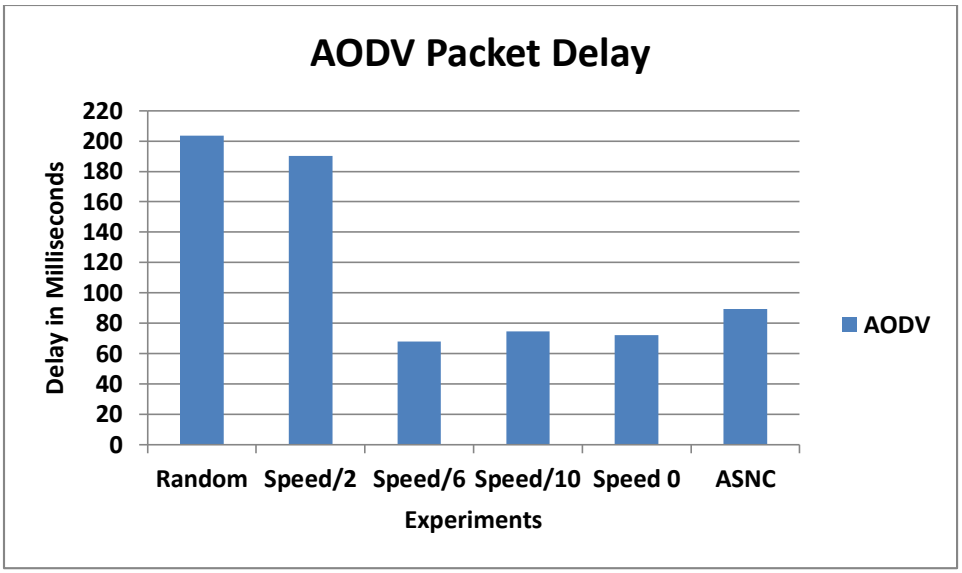

Figure 8.17: Packet Delay over AODV

8.1.1.6 FPTD for SDS VS. ASNC

The forwarding path detection-time in the comparison experiments between SDS and ASNC over AODV routing protocol in MANET is 39 seconds (Figure 8.18).

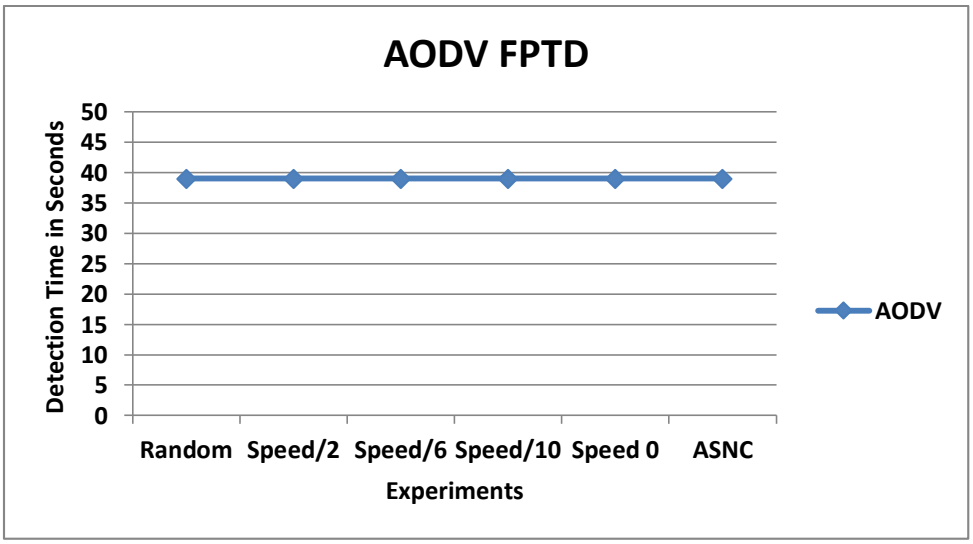

Figure 8.18: FPTD over AODV

This result is the same for all the 20 mobile node density experiments, Random, Speed/2, Speed/6, Speed/10, Speed Zero, and ASNC because all the experiments simulated in NS2 use the same movement file generated by Setdest tool.

8.1.2 Performance of SDS VS. ASNC over DSDV

The results relating to the packet delivery ratio are presented in Section 8.1.2.1 for SDS and ASNC experiments over DSDV routing protocol in MANET. The distance covered by the mobile nodes during the entire simulation run time is given in Section 8.1.2.2 for both SDS and ASNC experiments. Section 8.1.2.3 presents the active forwarding nodes over DSDV during the network simulation, while the pheromones map for ASNC experiment is illustrated in detail in Section 8.1.2.4, followed by description of the packet delay and the forwarding path detection time in Sections 8.1.2.5 and 8.1.2.6 respectively.

8.1.2.1 PDR for SDS VS. ASNC

In general, using SDS, the self-organization algorithm, and ASNC, the Swarm Intelligence algorithm, over DSDV routing protocol significantly increase the packet delivery ratio (Figure 8.19).

The experiment of Random movement for the mobile nodes in MANET shows that only 5% of the total packets were delivered over DSDV compared with 38% in AODV. This is followed by 16% PDR for slowing down the forwarding nodes by half, 53% less than AODV PDR, with 54% PDR for Speed/6 experiment (86% in AODV). Reducing the speed by 10% increases the delivery ratio to 61% and stopping the forwarding nodes movement shows 70% PDR. While in AODV the delivery ratio for Speed/10 and Speed Zero experiments were 90% and 92% respectively (Figure 8.1).

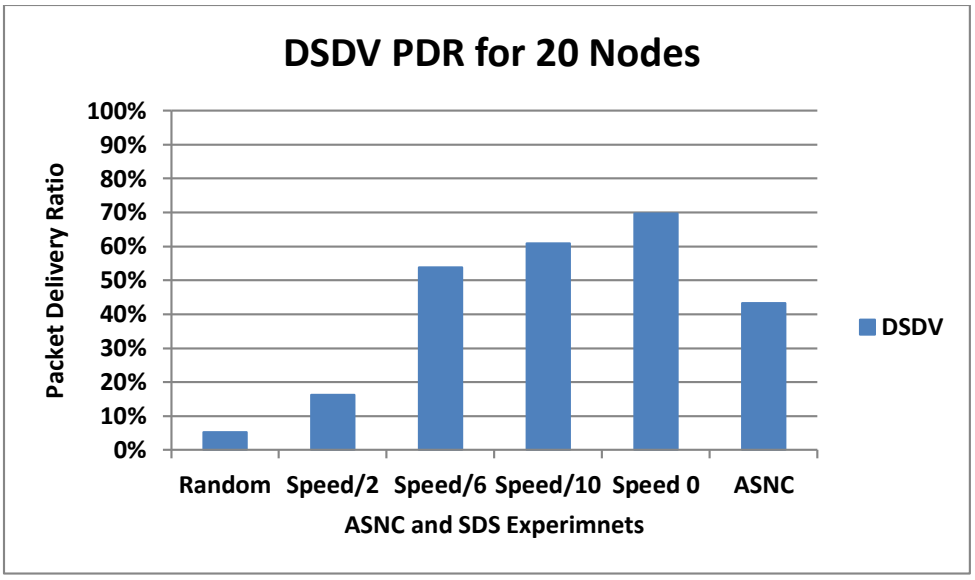

Figure 8.19: PDR Comparison over DSDV

In ASNC experiment over DSDV 43% of the packets were delivered to the destination, showing an improvement of 38% more than Random movement and 27% more than Speed/2 experiment. The PDR result for ASNC over AODV 76%, is higher than the DSDV result of, 43%, due to the DSDV delay in searching and advertising the routes even if not needed, which affects the network performance.

The experiments using DSDV routing protocol show that zero packets were transmitted during the first 113 seconds of the simulation run time (Figure 8.20) compared to 39 seconds in AODV. After DSDV routing protocol detected the first forwarding path at 113 seconds the packets started transmitting from the source through the forwarding nodes to reach the destination. After 150 seconds, 375 packets were delivered in 30 seconds in Random movement experiment, which dropped to zero at 180 seconds due to the dynamic movement of the mobile nodes. The number of packets delivered in Random movement remains at zero until 390 second points, raising up to deliver 171 packets at 420 seconds,

then dropping again to zero after 30 seconds, and remaining at zero till the end of the simulation.

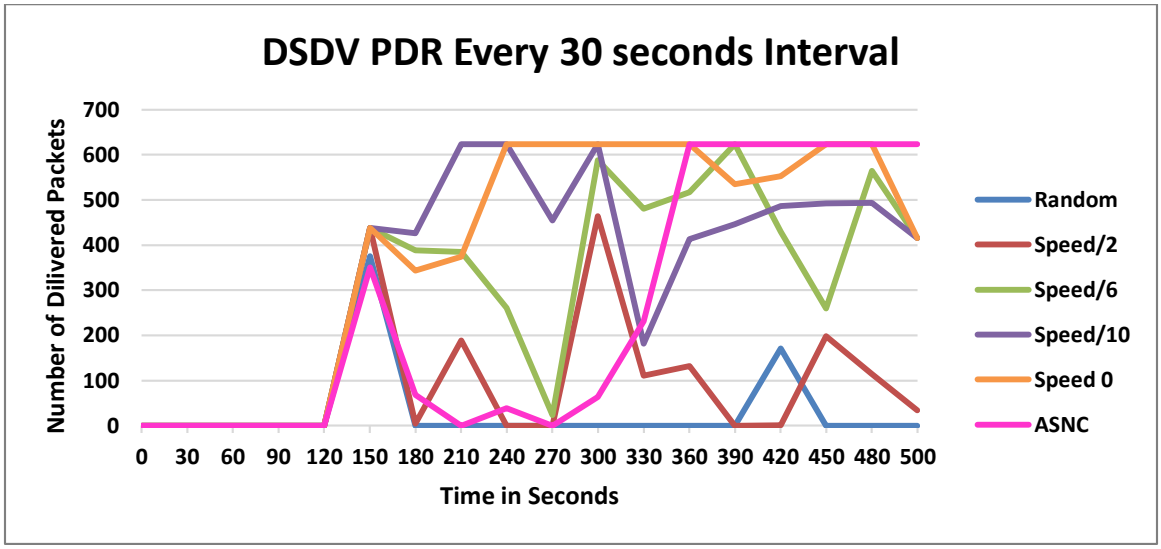

Figure 8.20: PDR every 30 seconds over DSDV

The experiment of slowing down the speed of the forwarding nodes by half delivered 437 packets at 150 seconds, 189 packets at 210 seconds, 463 at 300 seconds, 110 at 330 seconds, 131 at 360 seconds, 198 at 450 seconds, and zero packets at various points in between. Reducing the speeds of the forwarding nodes improved the packet delivery during the experiments, resulting in a higher number of packets delivered in Speed/6 and Speed/10 experiments. Even though the forwarding nodes stopped and not moving from the moment that DSDV detected the first forwarding path in Speed Zero experiment, Figure 8.20 shows the number of the delivered packets dropped at 180 seconds, and again, from the maximum of 623, at 390 seconds. The reason behind these drops in Speed Zero is the proactive nature of DSDV routing protocol to search for new routes even if there is a good connection route already between the communication nodes.

For ASNC experiment using DSDV, 350 packets delivered at 150 seconds dropped to zero packets at 210 seconds. The graph in Figure 8.20 then shows a rise in delivery of 68 packets at 240 seconds, dropping again to zero packets after 30 seconds. During the time between 180 to 270 seconds, the ant-agents were searching the simulation area in order to move toward the best forwarding positions. The preferred position was found at 360 seconds when the maximum number of packets delivered during 30 seconds, 623 packets, continued to be delivered until the end of the simulation run time.

8.1.2.2 Distance Covered in SDS VS. ASNC

Figure 8.21 shows the average of the distance covered by the mobile nodes in the 20 mobile node density experiments of ASNC and SDS using DSDV. The 20 mobile node density has two stationary nodes, source and destination, and 18 mobile nodes moving around the simulation area for the 500 seconds simulation run time. The distance covered for the 18 mobile nodes over DSDV is calculated at 10 second intervals in each experiment.

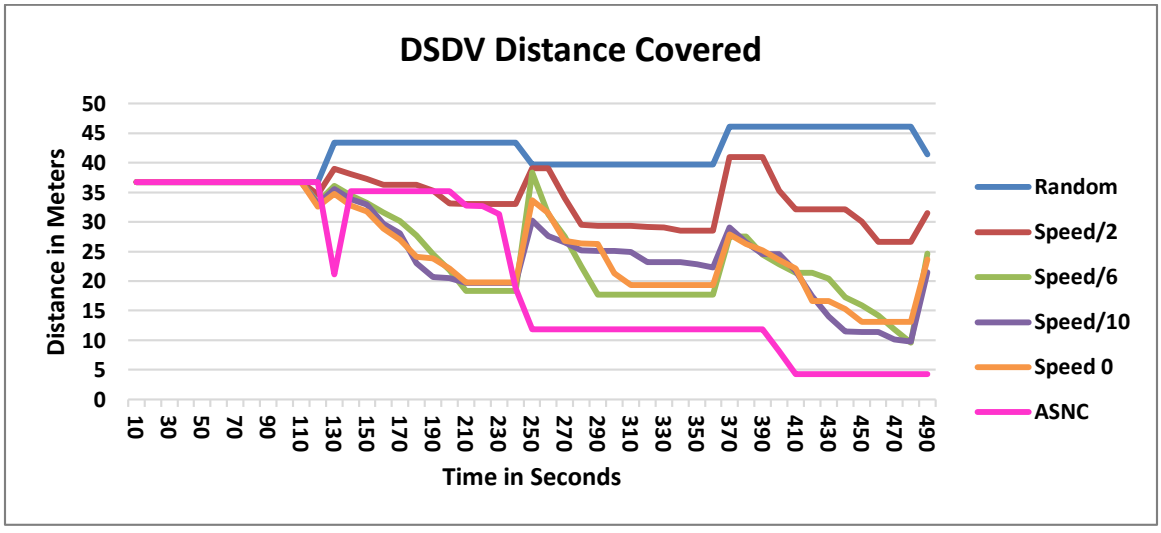

Figure 8.21: Distance Covered over DSDV

The DSDV routing protocol detected the first forwarding path at 113 seconds; before that the distance covered in all the experiments was 37 meters per 10 seconds. This result is the same as for AODV (Figure 8.2) because all the simulated experiments in Chapter 8 over AODV and DSDV routing protocols are using one movement file. For Random movement, the mobile nodes continue to cover the same distance as in AODV Random movement, 43 m from 130 second until 240 second, reducing to 40 m until 360 second and then 46 m to the end of the simulation run time. At 113 second the mobile node control mechanism starts in ASNC and SDS experiments after detecting the first forwarding path. In Speed/2, the distance covered reduced along with reducing the speed by half to range between 41 and 27 meters for 10 second intervals. Also in Speed/6, Speed/10 and Speed Zero experiments, the covered distance drops down relative to slowing the forwarding nodes speed while the other mobile nodes continue to move randomly. The distance covered for these experiments reaches around 20 meters at 210 seconds, rising up at 260 seconds, then dropping down to 20 meters again, to finally range between 25 and 10 in the last 100 seconds of the simulation, with an average of 22 meters.

In the experiment of ASNC, at 130 seconds after DSDV detected the forwarding path the covered distance drops to 21 meters, compared with 43 meters in Random movement. After that, the covered distance rises to 35 meters compared also with 43 meters in Random movement, until the 230 second points. During these seconds from 150 to 230 the mobile nodes, representing the ant-agents, were following the pheromone trails in the simulation area searching for good forwarding positions. At 250 seconds, the mobile nodes found their preferred forwarding positions and the distance covered drops significantly to 11 meters, compared to 40 meters in the Random movement, and at 410 seconds the distance covered

was 4 meters per 10 seconds, remaining the same until the end of the simulation run time. Following the pheromone trails in the simulation area control, the mobile node movements move closer and slower in the transmission range of the communication nodes, resulting in improving the network performance.

8.1.2.3 Forwarding Nodes in SDS VS. ASNC

The active forwarding nodes for the ASNC and Speed Zero experiments, the experiments yielding the highest PDR when using DSDV routing protocol, are illustrated in Figures 8.22 and 8.23 respectively. The experiments have been run in MANET with the same movement file for 20 mobile nodes density where node zero is the source and node one is the destination.

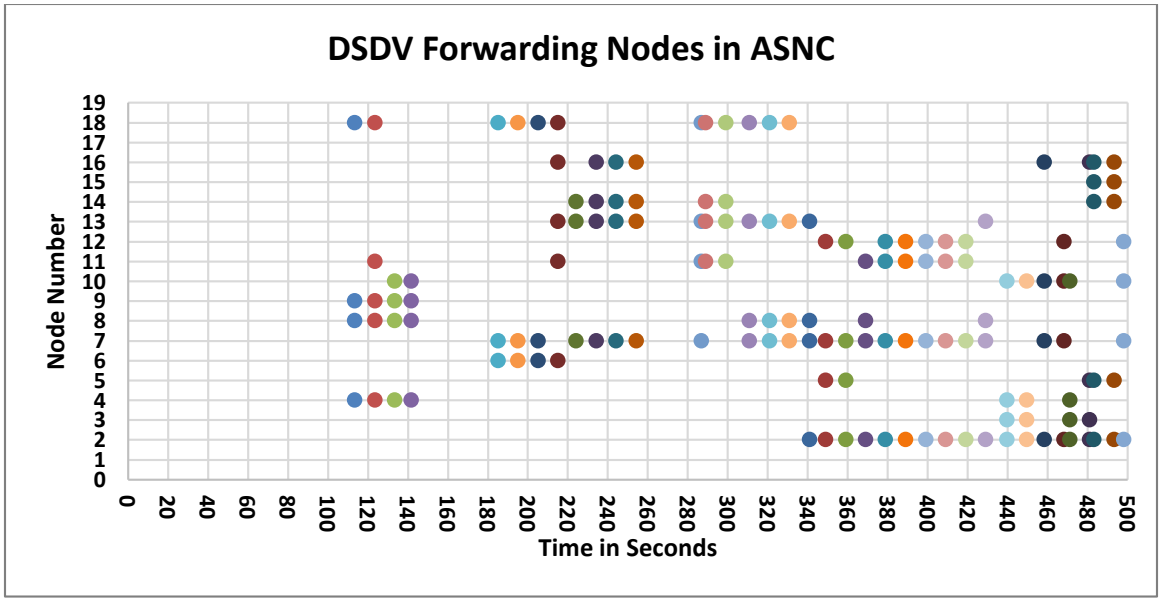

Figure 8.22: Active Forwarding Nodes in ASNC over DSDV

In Figure 8.22 the active forwarding nodes 4, 8, 9, and 18 start showing at 113 seconds of ASNC simulation run time, the detection time for the first forwarding path, to transmit the data packets from the source to the destination. In order to attract other mobile nodes in the

network, these four forwarding nodes leave pheromone trails on their positions. Other mobile nodes start moving toward the deposited pheromone, creating more forwarding paths between the communication nodes. When a route is disconnected due to mobile node movements, or when a better route shows up, DSDV routing protocol replaces the old route with the new one. Finding new routes and maintaining the broken ones in DSDV causes a long delay, which affects the performance of the network. The time from 142 seconds to 185 seconds show zero active forwarding nodes; no forwarding path was detected during this period of simulation time. The forwarding nodes show up again at 185 seconds creating eight forwarding paths until time 254. The two forwarding paths showed up at 185 and 195 seconds forwarded almost zero packets, that because the number of the active forwarding nodes in each path was only three nodes and they remained in their positions for a few seconds. The connection is lost for 30 seconds because of the delay in DSDV while the mobile nodes move from one forwarding position to another searching for the preferred places. The connection is established again at 286 seconds with the forwarding nodes 7, 11, 13, and 18. The connection remains for the rest of the simulation run time but with changes in the routing tables results of a different number of forwarding nodes participating. Most of the nodes forward packets through the network during 500 second of the simulation time, except for nodes 17 and 19.

In Speed Zero experiment (Figure 8.23) the active forwarding nodes 4, 8, 9, and 18 show up when DSDV detects the first forwarding path at 113 seconds the same as the first forwarding nodes in ASNC (Figure 8.22). Even though the forwarding nodes in Speed Zero stop moving at the moment of forward path detection, DSDV routing protocol still changes the routes during the rest of the simulation. Unlike AODV, DSDV tends to search for

alternative forwarding paths even if there is already a good connection between the source and the destination.

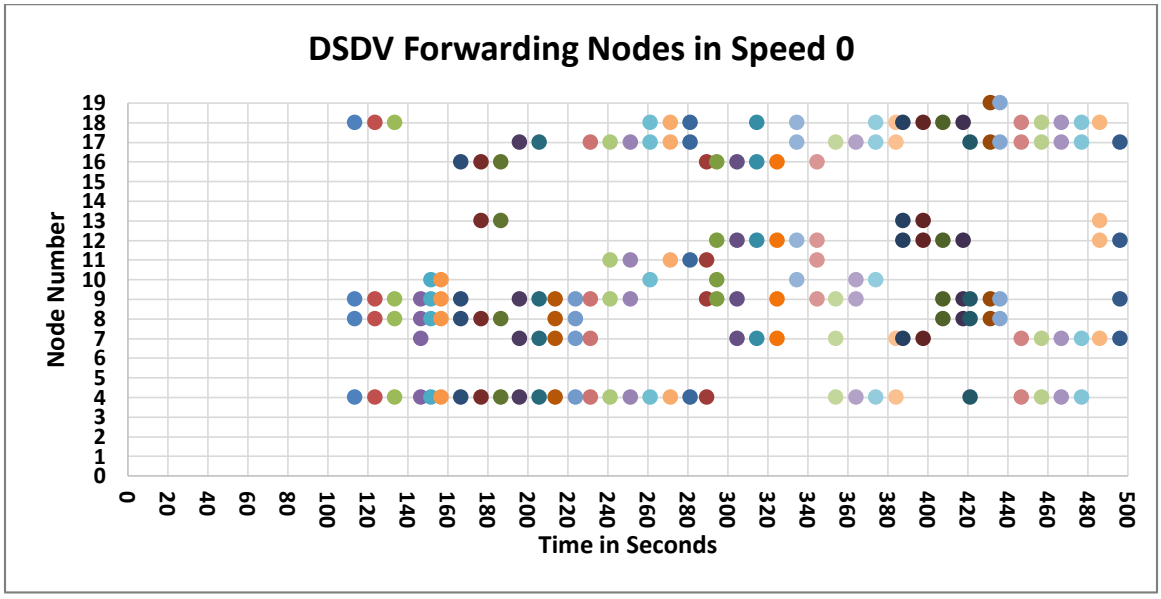

Figure 8.23: Active Forwarding Nodes in Speed Zero over DSDV

The continued changes in the forwarding path cause drops in the number of packets transmitted, which affects the network performance. However, the number of the participating forwarding nodes are higher than the number in Speed Zero experiment using AODV. In Speed Zero over DSDV, 12 nodes out of 18 participate in forwarding packets through the network during the 500 second simulation run time.

8.1.2.4 ASNC Pheromone Map

In ASNC experiment, the mobile nodes leave pheromone trails on their way, the same as a swarm of ants in nature. Different amounts of these pheromones are deposited in different positions over the simulation area. An X Y bubble-scatter map is used to visualize the deposited pheromone trails in the 20 mobile nodes density experiment of ASNC over DSDV. Each pheromone map is 1000x1000 meters and divided into 100 meter squares.

Each bubble represents the total amount of the deposited pheromone in a square of 100 meters and the bubble radius demonstrates the amount of the pheromone. The stationary communication nodes have constant pheromone amounts, equal to one, during the entire simulation run time.

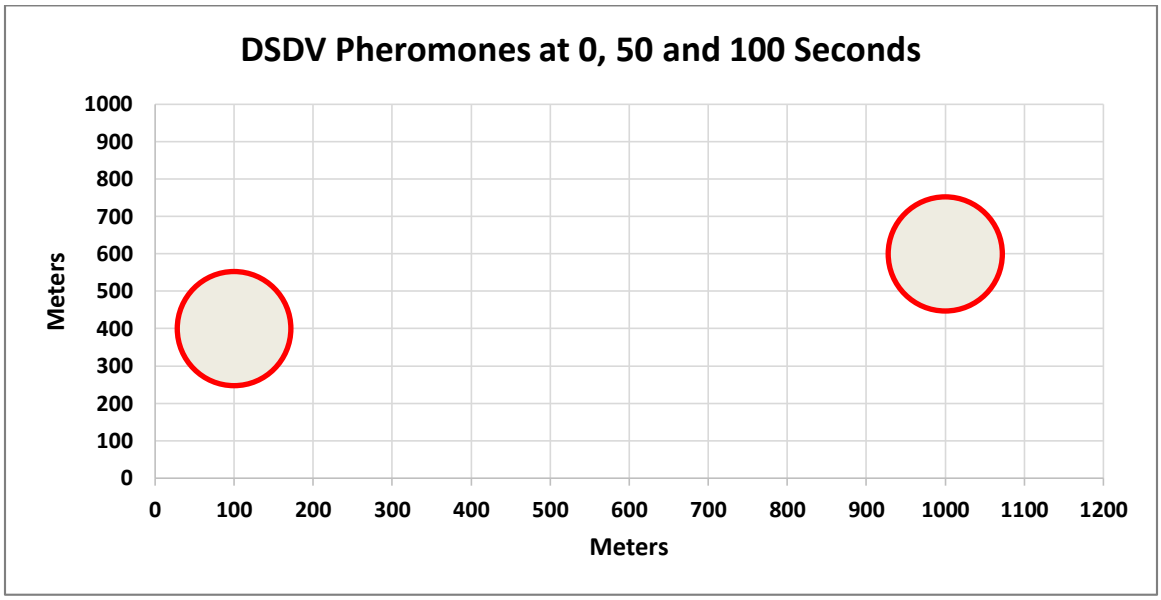

Figure 8.24: ASNC Pheromone Map over DSDV at Time Zero, 50 and 100

During the first 100 seconds, zero forwarding paths were found by DSDV routing protocol resulting in zero pheromone trails deposited, except for the pheromone of the communication nodes (Figures 8.24).

DSDV detects the first forwarding path at 113 seconds and the forwarding node leaves a pheromone trail on the found forwarding position, which attracts other mobile nodes to follow the steps of the forwarding nodes. By the time of 150 seconds, eight bubbles shows up in Figure 8.25, six of them at 600 Y coordinate and the other two at 500 Y coordinate. The lowest pheromone amounts are located at (200,600), (900,500), (900,600), and (1000,600), equal to one, followed by 1.96 at (300,600), 2.92 at (500,600), 3.92 at

(800,500), and 4.92 at (700,600) coordinates. The pheromone trails are not permanent, they evaporate over time if the mobile nodes decide not to move toward these positions and they increased when the mobile nodes choose these forwarding positions. The distance between the pheromone trails must be less than, 250 meters, the transmission range. For the rest of the simulation, the mobile node follows the pheromone from one forward position to another close to the communication nodes and far from the edges.

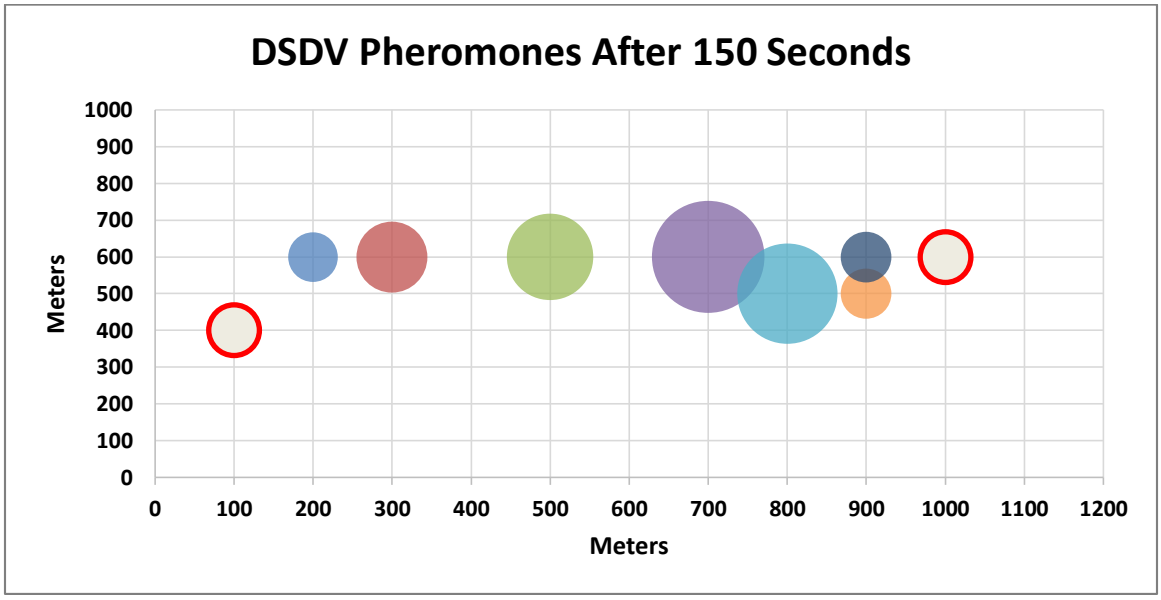

Figure 8.25: ASNC Pheromone Map over DSDV at Time 150

Figure 8.26 shows new bubbles at 200 seconds located at (300,400) and (500,500) coordinates with a size of 2. The amount of the pheromone trails existed before being updated at 200 seconds.

At 250 seconds, more forwarding positions are found by DSDV and more pheromone trails deposited by the forwarding nodes, (Figure 8.27). However, one of the existing pheromone positions located at (900,500) coordinate was not preferred by the mobile nodes, which caused its entire amount of pheromone to evaporate from the map. Also, the amount of the

other pheromone trails reduced or increased based on their positions and amounts of pheromone deposited by the mobile nodes.

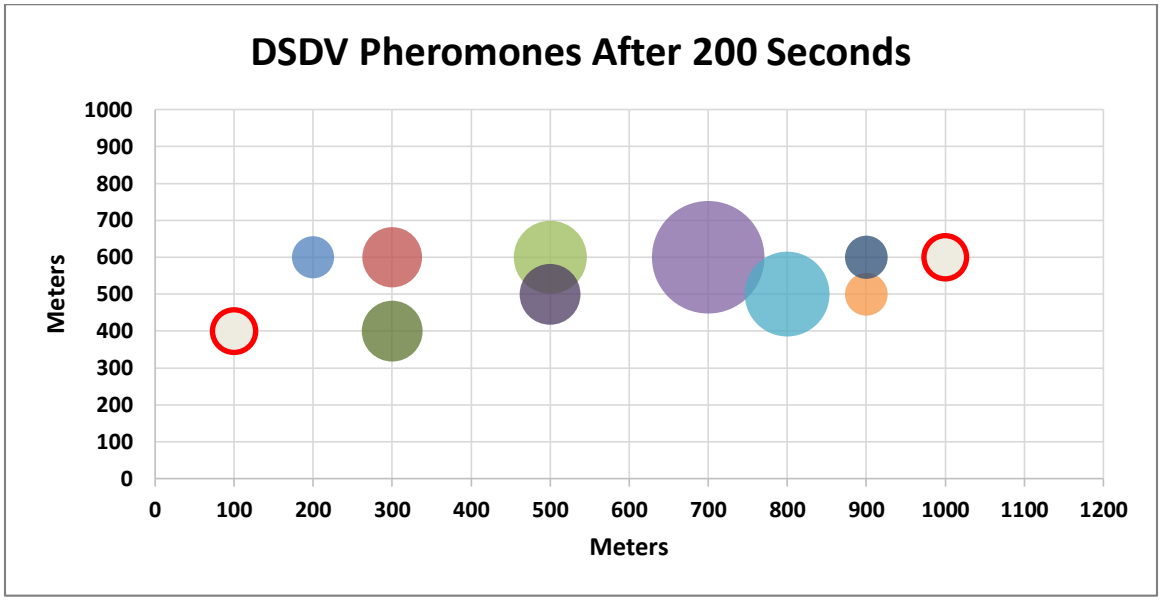

Figure 8.26: ASNC Pheromone Map over DSDV at Time 200

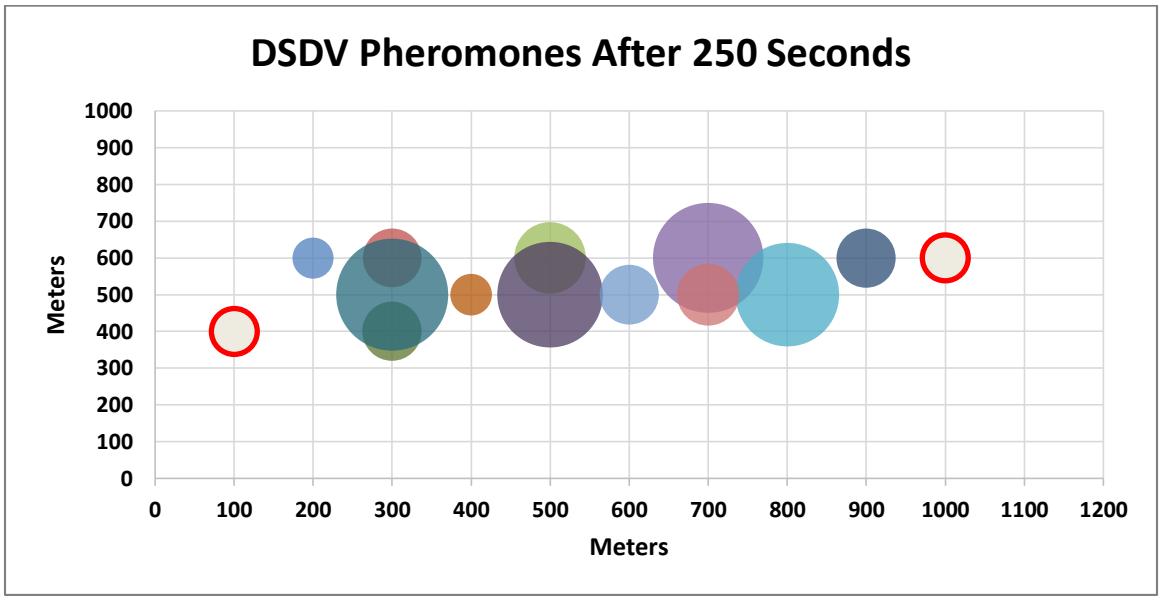

Figure 8.27: ASNC Pheromone Map over DSDV at Time 250

The preferred forwarding positions become obvious during the next 100 seconds of the experiment and their bubble sizes get larger (Figures 8.28 and 8.29). On the other hand,

the pheromone trails of the non-preferred positions continue to evaporate over time and the sizes of their bubbles shrink. The four bubbles located at (300,500), (500,500), (700,600), and (800, 500) coordinates have the highest amount of pheromones on the map.

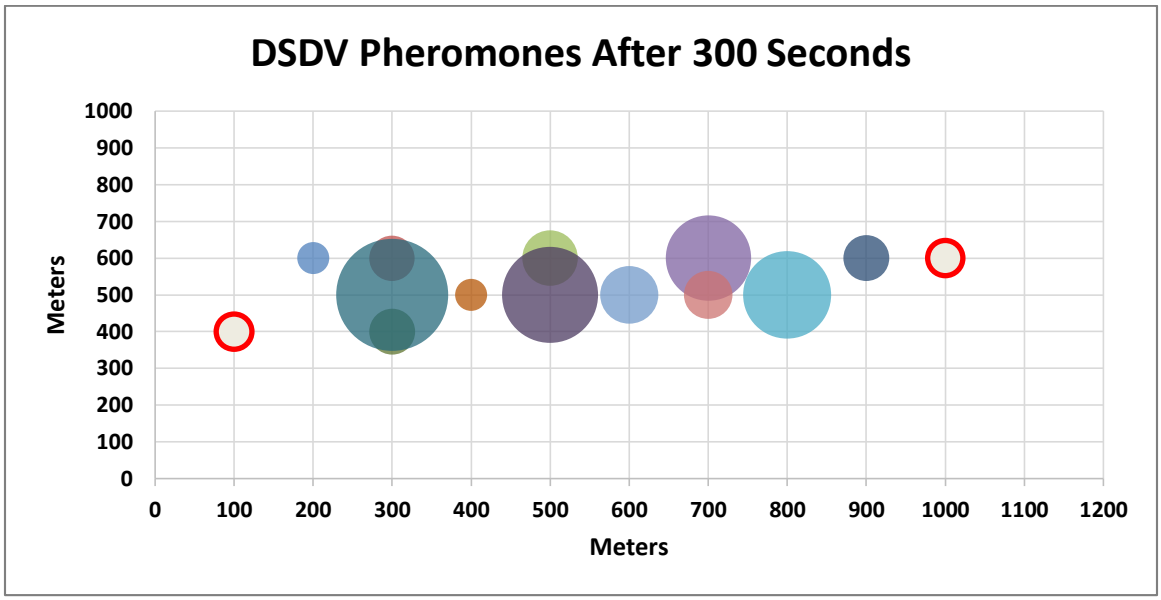

Figure 8.28: ASNC Pheromone Map over DSDV at Time 300

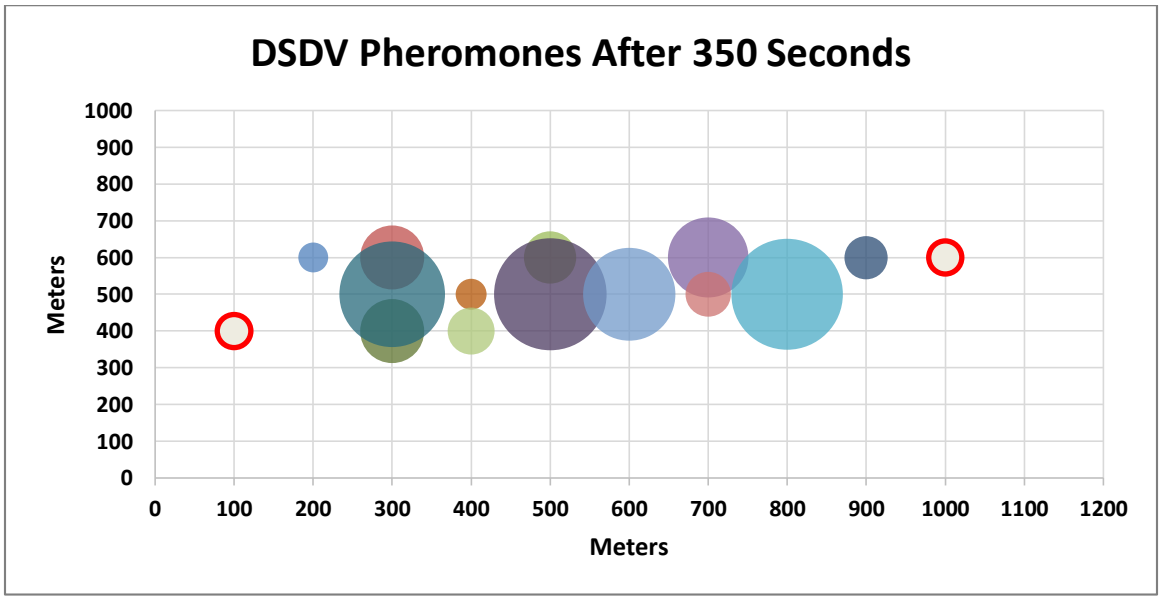

Figure 8.29: ASNC Pheromone Map over DSDV at Time 350

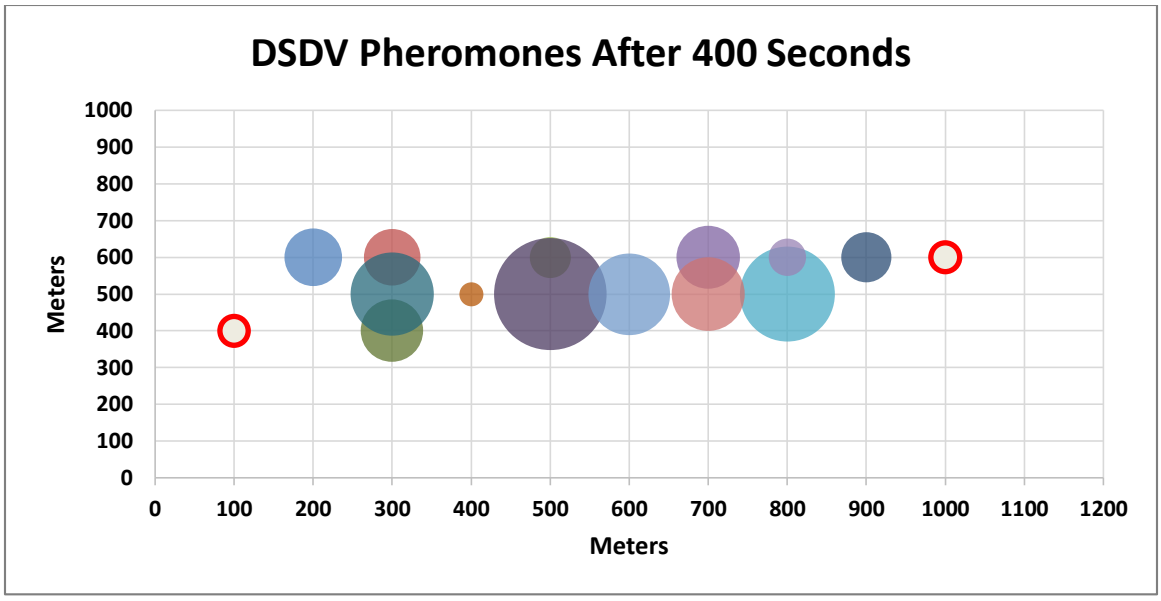

Figure 8.30: ASNC Pheromone Map over DSDV at Time 400

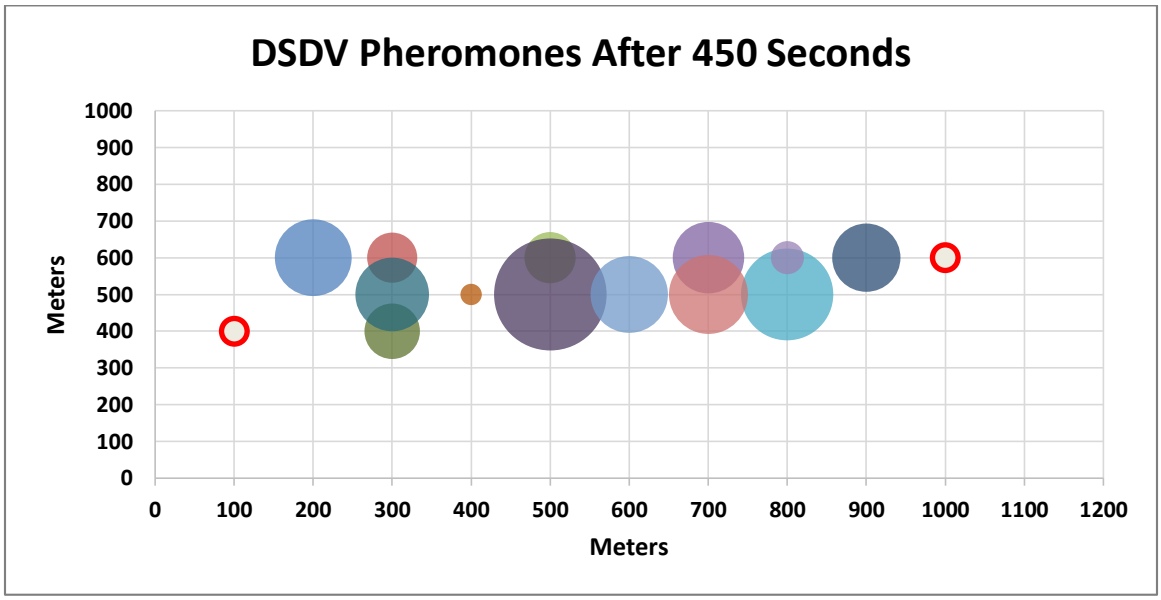

Figure 8.31: ASNC Pheromone Map over DSDV at Time 450

The pheromone amounts and positions keep changing over time at 400 and 450 seconds along with the changes in the movement of the mobile nodes and in the forwarding paths detected by DSDV routing protocols (Figures 8.30 and 8.31). During the simulation run

time, the area where the mobile nodes move become restricted to 400-600 coordinates on the Y-axis and 200-1000 coordinates on the X-axis.

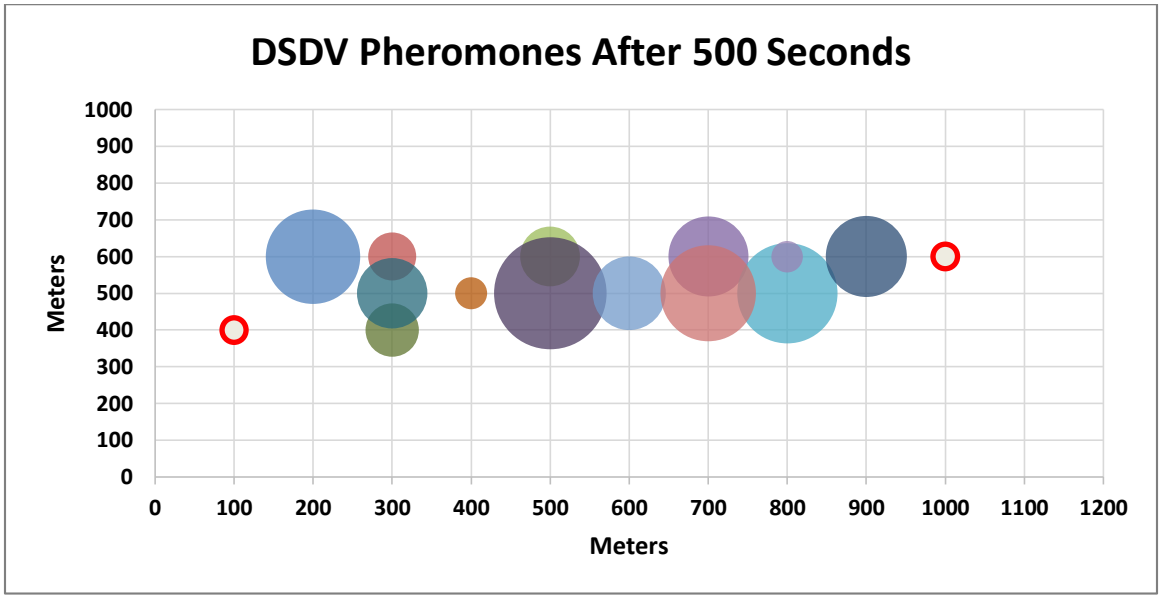

Figure 8.32: ASNC Pheromone Map over DSDV at Time 500

At the end of the simulation run time, Figure 8.32 shows the pheromone trails deposited in order, almost as a straight line, connecting the source and the destination. Due to the rapid changes in the DSDV routes, it is hard to predict which pheromone positions are preferred over others. From the active forwarding nodes in ASNC over DSDV (Figure 8.22), it is obvious that the forwarding path contains four hops. However, the four largest bubbles in the map at 500 seconds located at (200, 600), (500, 500), (700, 500), and (800, 500) coordinates are most likely to be the preferred positions.

8.1.2.5 End-to-End Delay for SDS VS. ASNC

The packet delay comparison between SDS and ASNC experiments over DSDV for 20 mobile node density is given in Figure 8.33. As expected, Random movement has the

longest delay with 132 ms, which reduces significantly after applying the dynamic-movement node-control algorithms.

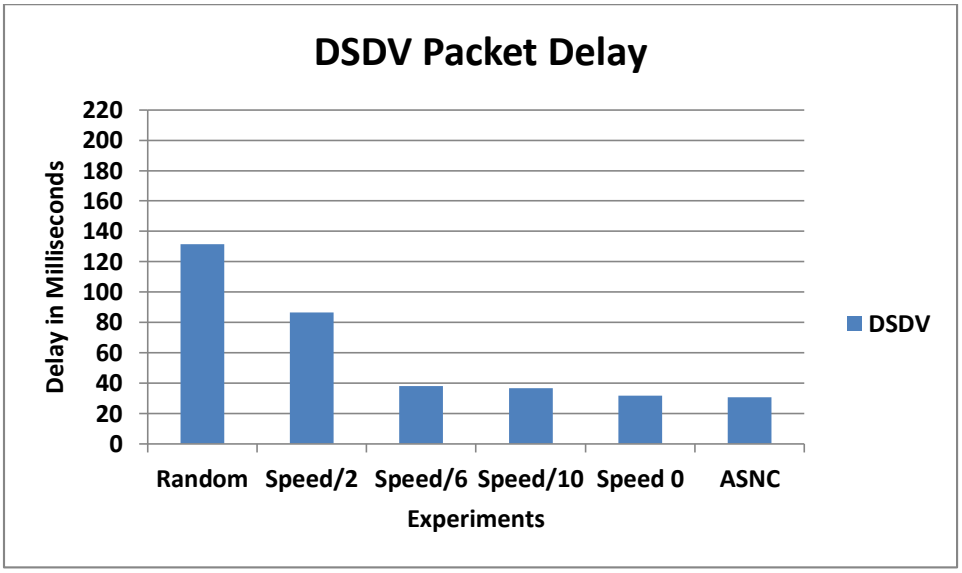

Figure 8.33: Packet Delay over DSDV

The next long delay, 87 ms, is for slowing down the speed of the forwarding nodes by half, followed by the delay for Speed/6 and Speed/10 experiments of 38 ms and 37 ms respectively. The shortest packet delays were presented by Speed Zero and ASNC experiments, where the delay was reduced to 32 and 31 ms respectively.

8.1.2.6 FPTD for SDS VS. ASNC

The detection time for the first forwarding path in SDS and ASNC experiments using DSDV routing protocol is 113 seconds (Figure 8.34).

The reason behind detection of the first forwarding path at the same second in all the experiments is that all the experiments use the same movement file thatwas used for all the comparison experiments in this chapter.

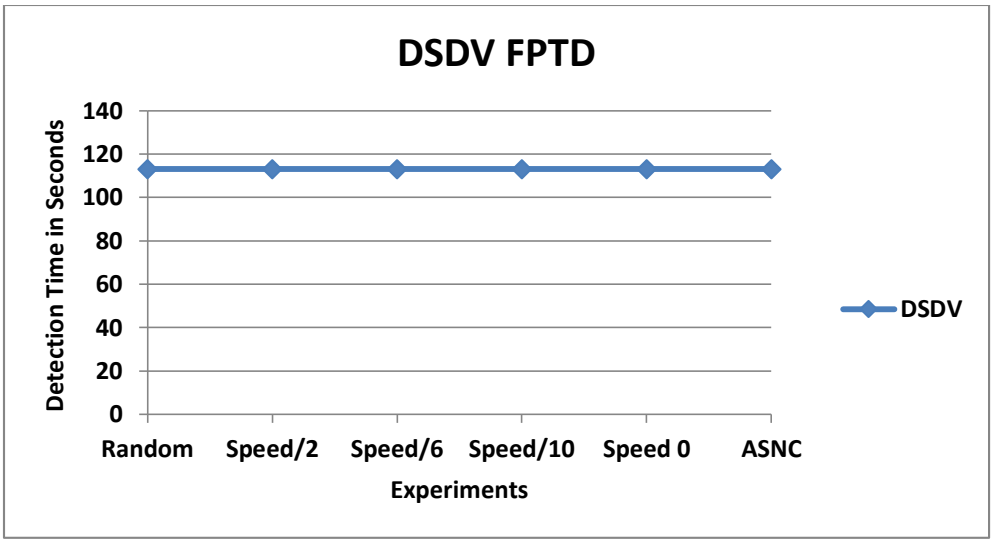

Figure 8.34: FPTD over DSDV

8.2 SDS VS. ASNC Summary

This chapter described a comparison between two self-organization algorithms over AODV and DSDV routing protocols. The first algorithm is a simple self-organization control that aims to slow down the speed of the forwarding nodes, while the other algorithm is a complex self-organization control adapted from the behavior of swarms in nature. The proposed SDS and ASNC algorithms compared in NS2 for 20 mobile node density utilized a movement file generated by the Setdest tool. Because all the experiments use the same movement file once using AODV and again using DSDV, the FPTD for all the AODV experiments was 39 seconds, and 113 seconds for the experiments using DSDV. In general, AODV routing protocol performs better than DSDV in all the experiments, confirming the observation about the efficiency of AODV in high dynamic networks.

Applying the movement control algorithms SDS and ASNC in MANET improves the packet delivery significantly. Slowing down the speed of the forwarding nodes by half shows an increase in the delivery ratio compared with Random movement results. The

packet delivery ratio then increases relatively by slowing down the speeds to reach the highest PDR in Speed Zero experiment. ASNC shows double the PDR of Random movement, almost as high as Speed/6 results. ASNC is preferred over Speed Zero even though Speed Zero experiment gives a high PDR.

For the total distance covered by the mobile nodes, ASNC demonstrates the best results of all the experiments using AODV and DSDV routing protocols, proving that following the deposited pheromones in MANETs controls the movement of the mobile nodes to reduce the distance and speed of the movements. The experiments of slowing down the speeds also reduces the total distance covered relative to reducing the forwarding nodes speed.

ASNC almost allows all the mobile nodes to participate in forwarding packets during the simulation run time in AODV and DSDV. However, in Speed Zero experiment using AODV routing protocol the forwarding nodes, six nodes on average, stop moving at the moment AODV detects the first forwarding path and remain stopped until the end of the simulation. In the experiment using DSDV routing protocol the results for Speed Zero change due to the proactive nature of DSDV. DSDV tends to replace the existing routes, which causes changes in the active forwarding nodes during the simulation run time.

Visualizing the pheromone positions and amounts in X Y bubble-scatter maps at 50 second intervals shows how the pheromone is distributed over the simulation area as well as the forwarding positions preferred by the ant-agents, in addition to the restricted area between the communication nodes where the agents tend to move after detecting the first forwarding path instead of moving far away to the edges.

Chapter 9

Conclusions

9.1 Results

This master's thesis presents two movement control algorithms based on awareness of MANET traffic condition that results in significant improvement in the network performance. The new proposed self-organization algorithms are SDS, Slow Down Speed, and ASNC, Ant System Node Control, algorithms. SDS follows a simple self-control decision to slow down the speed of the forwarding nodes to extend the forwarding path life time. On the other hand, ASNC is a more complex algorithm adapted from a Swarm Intelligence algorithm to control the mobile nodes in MANET. The mobile node in ASNC behaves as ants in nature; they search the surrounding area depositing pheromone trails on their favorite positions to attract other mobile nodes in the network. Both of the algorithms simulated in NS2 over different mobile node densities, 5, 10, 20, 30, and 40 nodes, have utilized AODV and DSDV routing protocols. AODV routing protocol shows better results than DSDV in SDS and ASNC experiments. DSDV advertising for new routes and maintaining the broken route cause long delays in detecting forwarding paths, which affects the network performance. AODV techniques for discovering and maintaining routes is definitely the best choice for high-dynamic movement networks such as MANET. In SDS and ASNC experiments, AODV was able to detect the first forwarding path 50 seconds earlier than DSDV on average.

The results of SDS algorithm show improvement in MANET performance in the four implemented scenarios: Speed/2, Speed/6, Speed/10, and Speed Zero compared with

Random movement. Slowing the speeds of the forwarding nodes increased the packet delivery ratio especially in 10 and 20 mobile node density experiments. However, the high mobile node densities of 30 and 40 mobile nodes perform very well even in Random movement, while the very low mobile density of five shows zero packets delivered during all SDS scenarios including Random movement. The delivery ratio increases relative to slowing down the speed showing a very high PDR for Speed Zero scenario, followed by Speed/10, Speed/6, and finally Speed/2 with the lowest PDR of them all.

ASNC experiments also improve the performance of MANET significantly by providing connection between the communication nodes most of the time to increase the number of the delivered packets. The pheromone trails in ASNC attract the mobile nodes toward the forwarding positions to improve the delivery ratio. The PDR in ASNC increases by 20% to 60% along with the increase in the mobile node densities compared with Random movement.

The experiments of 20 mobile node density demonstrate the best benefit of SDS and ASNC in MANET over AODV and DSDV routing protocols. Based on this, the density of 20 mobile nodes is used in comparison between the performances of the two proposed algorithms. The comparison experiments utilized one movement file generated by Setdest tool to avoid any randomness in the results. The results of the comparison show low PDR for Random movement compared with all the experiments of SDS and ASNC, with the highest PDR result found the Speed Zero experiments, stopping the forwarding nodes. The results of the ASNC experiments shows PDR higher than Random movement and Speed/2 almost as good as the results of Speed/6 results. Although Speed Zero results in a high

delivery ratio in MANET, ASNC algorithm is preferred because the mobile nodes are actually moving the entire time.

9.2 Future Work

The unpredictable movement of the mobile nodes causes many issues that need to be solved in MANET including the rapid disconnection between the communication nodes. This thesis is focused on controlling the movement of the mobile nodes based on awareness of the traffic condition in the network. Due to the time limitations, the new proposed algorithms, SDS and ASNC, have been simulated over AODV and DSDV routing protocols with stationary source and destination for 5, 10, 20, 30, and 40 mobile node densities.

In future, other scenarios may be considered in experiments, such as changing the source and destination positions during the simulation rather than being stationary for the entire time. In addition, a scenario with multiple communication nodes, two senders and one receiver or one sender communicating with different receivers may provide interesting results. Other routing protocols may also be utilized, e.g. DSR, in order to compare the performance over different routing protocols.

For SDS algorithm, it is possible to add new speed scenarios for example Speed/4, Speed/5 and/or Speed/8. It would also be interesting to reduce the speeds of the forwarding nodes in one scenario by different amounts over time.

The ASNC algorithm adapts the Ant System from the Ant Colony Optimization algorithms in order to control the movement of mobile nodes. Other Swarm Intelligence algorithms presented in this thesis could also be adapted to control the movement of the mobile nodes instead of the Ant System algorithm, for example Rank-Based Ant System, Max-Min Ant

System and Best-Worst Ant System. Algorithms from Bee Colony Optimization rather than Ant Colony Optimization could also be explored, using for example Artificial Bee Colony.

Bibliography

[1] S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic (Eds.), Mobile Ad Hoc Networking, New York: Wiley-IEEE Press. 2001.

[2] F. Bei and A. Helmy, A survey of mobility models in wireless Ad hoc Networks, University of California, USA, 2004.

[3] J. DeDourek and P. Pochec, M2ANET: a Mobile Medium Ad Hoc Network, Wireless Sensor Networks: Theory and Practice, WSN 2011, Paris, France, Feb. 2011, pp. 1-4.

[4] Aarti and S. S. Tyagi, Study of MANET: Characteristics, Challenges, Application and Security Attacks, IJARCSSE, Volume 3, Issue 5, May 2013.

[5] Mesut Gunes, Udo Sorges and Imed Bouazizi, ARA – The Ant-Colony Based Routing Algorithm for MANETs, IWAHN, Vancouver, British Columbia, Canada, August 2002.

[6] F. Correia and T. Vazao, Simple Ant Routing Algorithm, ICOIN, IEEE, January 2008, pp. 1-8.

[7] Sundaram Rajagopalan and Chien-Chung Shen, ANSI: A swarm intelligence-based unicast routing protocol for hybrid ad hoc networks, Elsevier, 2006.

[8] Gianni Di Caro, Frederick Ducatelle and Luca Maria Gambardella, AntHocNet: an Ant-Based Hybrid Routing Algorithm for Mobile Ad Hoc Networks, Proc. 8th International Conf. on Parallel Problem Solving from Nature, 2004, pp. 461-470.

[9] G. Di Caro and M. Dorigo. AntNet: A Mobile Agents Approach to Adaptive Routing in Communication Networks. Unpublished presentation, Ninth Dutch Conference on Artificial Intelligence (NAIC '97), November 1997.

[10] Jianping Wang, Eseosa Osagie, Parimala Thulasiraman, and Ruppa K. Thulasiram, HOPNET: A hybrid ant colony optimization routing algorithm for mobile ad hoc network, Elsevier, Vol. 7, No. 4, June 2009, pp. 690-705.

[11] Nada Alsalmi, J. DeDourek and P. Pochec, "Self-organizing Mobile Medium Ad hoc Network", The Fourth International Conference on Mobile Services, Resources, and Users MOBILITY 2014, Paris, France, July 20 - 24, 2014.

[12] Abdullah Alshehri, J. DeDourek and P. Pochec, "The Advantage of Moving Nodes in Formations in MANETs and M2ANETs", The Ninth International Conference on Wireless and Mobile Communications ICWMC 2013, Nice, France, July 21-26, 2013, pp. 228-232.

[13] Mohammed Alzaylaee, J. DeDourek and P. Pochec, "Linear Node Movement Patterns in MANETS", The Ninth International Conference on Wireless and Mobile Communications ICWMC 2013, Nice, France, July 21-26, 2013, pp. 162-166.

[14] P. Sivakumar, R. Srinivasan and K. Saranya, An Efficient Neighbor Discovery Through Hello Messaging Scheme in MANET Routing Protocol, IJETAE, Vol. 4, No. 1, January 2014.

[15] G. S. Mamatha and S. C. Sharma, Analyzing the Manet Variations, Challenges, Capacity and Protocol Issues,IJCSES, Vol. 1, No. 1, August 2010.

[16] Ram Ramanathan and Jason Redi, A Brief Overview of Ad Hoc Networks: Challenges and Directions, IEEE Communication Magazine, May 2002.

[17] Pravin Ghosekar, Girish Katkar and Pradip Ghorpade, Mobile Ad Hoc Networking: Imperatives and Challenges, IJCA, Specila Issue on Mobile Ad-hoc Networks, 2010.

[18] Imrich Chlamtac, Marco Conti and Jennifer J.-N. Liu, Mobile Ad hoc Networking: Imperatives and Challenges, Elsevier B. V., 2013, pp. 13-64.

[19] Priyanka Goyal, Vinti Parmar and Rahul Rishi, MANET: Vulnerabilities, Challenges, Attacks, Application, IJCEM, Vol.11, January 2011.

[20] Wei Liu, Yanchao Zhang, Kejie Lu, and Yuguang Fang, Energy Conservation Through Resource-Aware Movement in Heterogeneous Mobile Ad hoc Networks, J Comb Optim, 2006, pp. 7-20.

[21] R. R. Roy, Handbook of Mobile Ad Hoc Networks for Mobility Models, e-ISBN 978-1-4419-6050-4, USA, 2011, pp.23-31.

[22] Tracy Camp, Jeff Boleng and Vanessa Davies, A Survey of Mobility Models for Ad Hoc Network Research, WCMC, special issue on Mobile Ad Hoc Networking research, Vol. 2, No. 5, 2002, pp. 483-502.

[23] D. Johnson and D. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing, Chapter 5, Kluwer Academic Publisher, 1996, pp. 153-181.

[24] Josh Broch, David Maltz, David Johnson, Yih-Chun Hu, and Jorjeta Jetcheva, A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols, MobiCom, ACM, Dallas, TX, October 1998, pp. 85-97.

[25] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, Advances in Network Simulation, IEEE Computer, vol. 33, No. 5, May 2000, pp. 59-67.

[26] C. Bettstetter and C. Wagner. The Spatial Node Distribution of the Random Waypoint Mobility Model, WMAN, Ulm, Germany, GI Lecture Notes in Informatics, no. P-11, Mar 25-26, 2002, pp. 41-58.

[27] E. M. Royer, P. M. Melliar-Smith, and L. E. Moser. An Analysis of the Optimum Node Density for Ad hoc Mobile Networks, IEEE, ICC, Helsinki, Finland, June 2001.

[28] Changling Liu and Jorg Kaiser, A Survey of Mobile Ad Hoc network Routing Protocols, University of Ulm, Tech.Report, October 2005.

[29] akshai Aggarwal, Savita Gandhi and Nirbhay Chaubey, Performance Analysis of AODV, DSDV and DSR in MANETs, IJDPS, Vol. 2, No. 6, November 2011.

[30] C.E. Perkins and P.Bhagwat, Highly Dynamic Destination Sequenced Distance-Vector Routing(DSDV) for Mobile Computers, IGCOMM, London, UK, August 1994.

[31] C. E. Perkins and P. Bhagwat, DSDV Routing over a Multihop Wireless Network of Mobile Computers, Mobile Computing, Chapter 6, Kluwer Academic Publisher, 1996, pp.183-206.

[32] H. Narra, Y. Cheng, E. Centinkaya, J. Rohrer, and J. Sterbenz, Destination-Sequenced Distance Vector (DSDV) Routing Protocols Implementation in ns-3, WNS, Barcelona, Spain, March 2011.

[33] B.N. Jagdale, Pragati Patil, P. Lahane, and D. Javale, Analysis and Comparison of Distance Vector, DSDVand AODV Protocol of MANET, IJDPS, Vol. 3, No. 2, March 2012.

[34] Jaya Bhatt and Naveen Hemrajani, Effective Routing Protocol (DSDV) for Mobile Ad Hoc Network, IJSCE, ISSN: 2231-2307, Volume-3, Issue-5, November 2013.

[35] Mehran Abolhassan, Tadeusz Wysocki and Eryk Dutkiewicz, A Review of Routing Protocols for Mobile Ad hoc Networks, Elsevier B. V., 2003.

[36] C. Perkins, E. Royer, S. Das, Ad hoc on-Demand Distance Vector (AODV) Routing, The Internet Society, July 2003.

[37] A. Boukerche, Performance Evaluation of Routing Protocols for Ad Hoc Networks, Mobile Networks and Applications, Kluwer Academic Publishers, 2004.

[38] W. Ross Ashby, Principles of the self-organizing system" in "Principles of Self-Organization", Transactions of the University of Illinois Symposium, H. Von Foerster and G. W. Zopf, Jr. (eds.), Pergamon Press: London, UK, 1962, pp. 255-278.

[39] E. Bonabeau, M. Dorigo, and G. Théraulaz. Swarm Intelligence: From Natural to Artificial Systems, Oxford, UK: Oxford University Press, 1999.

[40] Camazine, Deneubourg, Franks, and et al., "Self-Organization in Biological Systems", Princeton University, 2001.

[41] Zhongshan Zhang, Keping Long, Jianping Wang, and Falko Dressler, On Swarm Intelligence Inspired Self-Organized Networking: Its Bionic Mechanisms, Designing Principles and Optimization Approaches, Communication Surveys & Tutorials, IEEE, Volume 16, Issue 1, July 2013, pp. 513-537.

[42] G. Beni and J.Wang, Swarm intelligence in cellular robotics systems, in Proceeding of NATO Advanced Workshop on Robots and Biological System, 1989.

[43] Aleksandar Jevtic and Diego Andina, Swarm Intelligence and Its Applications in Swarm Robotics, 6th WSEAS Int. Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, Tenerife, Spain, December 14-16, 2007.

[44] A. K. Mishra, M. N. Das and T. C. Panda, Swarm Intelligence Optimization: Editorial Survey, International journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 1, January 2013.

[45] Saida ZIANE and Abdelhamid MELLOUK, A Swarm Intelligent Multi-path Routing for Multimedia Traffic over Mobile Ad hoc Networks, Q2SWinet'05, Montreal, Quebec, Canada, October 2005.

[46] Crina Grosan, Ajith Abraham and Monica Chis, Swarm Intelligence in Data Mining, Studies in Computational Intelligence (SCI) 34, 2006, pp. 1–20.

[47] Dario Floreano and Claudio Mattiussi, Bio-inspired Artificial Intelligence: Theories, Methods, and Technologies, The MIT Press, e-ISBN: 9780262272858, August 2008.

[48] James Kennedy, Russell C. Eberhart and Yuhui Shi, Swarm Intelligence, Elsevier, ISBN 1-55860-595-9, 2001.

[49] Dennis Plougman Buus, Constructing Human-Like Architecture with Swarm Intelligence, Aalborg University, Project Library, Aalborg, Denmark, August 2006.

[50] James Kennedy and Russell Eberhart, Particle Swarm Optimization, IEEE, 1995.

[51] Qinghai Bai, Analysis of Particle Swarm Optimization Algorithm, CCIS, Vol. 3, No. 1, February 2010.

[52] J. van Ast, R. Babuˇska, and B. De Schutter, Particle Swarms in Optimization and Control, Proceedings of the 17th IFAC World Congress, Seoul, Korea, July 2008, pp. 5131– 5136,.

[53] Marco Dorigo, Vittorio Maniezzo and Alberto Colorni, The Ant System: Optimization by a colony of cooperating agents, IEEE, Transactions on Systems, Man, and Cybernetics-Part B, Vol. 26, No. 1, 1996, pp.1-13.

[54] Marco Dorigo and Thomas Stutzle, Ant Colony Optimization, MIT Press, , ISBN 0-262-04219-3, June 2004.

[55] Satchidananda Dehuri, Susmita Ghosh and Sung-Bae Cho, Integration of Swarm Intelligence and Artificial Neural Network, World Scientific, ISBN: 978-981-4467-31-5, June 2011.

[56] Louis P. Walters, Application of Swarm Intelligence, Chapter 7 & 10, ISBN 978-1-61728-813-5, Nova Science Publishers, 2011.

[57] S. Prasad, Y.P.Singh and C.S.Rai, Swarm Based Intelligent Routing for MANETs, International Journal of Recent Trends in Engineering, Vol 1, No. 1, May 2009.

[58] Rajeshwar Singh, D. K. Singh and Lalan Kumar, Swarm Intelligence Based Approach for Routing in Mobile Ad Hoc Networks, International Journal of Science and Technology Education Research Vol. 1(7), December 2010, pp. 147 – 153,.

[59] Binitha S and S Siva Sathya, A Survey of Bio inspired Optimization Algorithms, International Journal of Soft Computing and Engineering (IJSCE), ISSN: 2231-2307, Volume-2, Issue-2, May 2012.

[60] S.D. Shtovba, Ant Algorithms: Theory and Applications, Programming and Computer Software, Vol. 31, No. 4, 2005, pp. 167-178.

[61] Andries P. Engelbrecht, Computational Intelligence An Introduction, John Wiley & Sons, Ltd, ISBN 978-0-470-03561-0, Chapter 17, 2007.

[62] G.C. Onwubolu and B.V. Babu, New Optimization Techniques in Engineering, Studies in Fuzziness and Soft Computing, Springer, Vol. 141, Chapter 5, ISBN 978-3-540-39930-8, 2004.

[63] Ping Guo and Zhujin Liu, An Ant System based on Moderate Search for TSP, ComSIS, Vol. 9, No. 4, special Issue, December 2012.

[64] Bernd Bullnheimer Richard F. Hartl and Christine Strauß, A New Rank Based Version of the Ant System - A Computational Study, Central European Journal for Operations Research and Economics, 1997.

[65] Thomas Stützle and Holger Hoos, Improvements of the ANT-System - Introducing the Max-Min Ant System, Technical Report AIDA-96-12, TH Darmstadt, revised version, 1996.

[66] Thomas Stützle and Holger Hoos, Improvements on the Ant System: Introducing the MAX-MIN Ant System, ICANNGA, 1997.

[67] Thomas Stützle and Holger Hoos, The MAX-MIN Ant System and Local Search for the Traveling Salesman Problem, IEEE, 1997.

[68] R. Laptik and D. Navakauskas, MAX-MIN Ant System in Image Preprocessing, Electronics & Electrical Engineering, Issue 89, January 2009.

[69] Aristidis Vlachos, Optimizing Large- Scale Combinatorial Problems using Max-Min Ant System Algorithm, E-ISSN: 2224-266X, Issue 9, Volume 11, September 2012.

[70] Oscar Cordo´n, In˜aki Erna´ndez de Viana, and Francisco Herrera, Analysis of the Best-Worst Ant System and Its Variants on the QAP, ANTS, LNCS 2463, Springer, 2002, pp. 228-234.

[71] Sjoerd van Egmond, Dynamic Ant Colony Optimization for the Traveling Salesman Problem, LIACS, Chapter 3, Internal Report, Leiden University, July 2012.

[72] Karl Von Frisch, "Decoding The Language of The Bee", University of Munich, Federal Republic of Germany, Nobel Lecture, The Nobel Foundation, December 1973.

[73] Fred C. Dyer, The Biology Of The Dance Language, Annual Reviews, Vol. 47, January 2002.

[74] Dušan Teodorović and Panta Lučić, Computing with Bees: Attacking Complex Transportation Engineering Problems, International Journal on Artificial Intelligence Tools, World Scientific, Vol. 12, No. 3, September 2003.

[75] Li-Pei Wong, Malcolm Yoke Hean Low and Chin Soon Chong, Bee Colony Optimization with Local Search for Traveling Salesman Problem, IEEE, July 2008, pp. 1019-1025.

[76] Dušan Teodorović, Bee Colony Optimization (BCO), Springer, e-ISBN 978-3-642-04225-6, 2009, pp.39-60.

[77] Dusan Teodorovic´and Mauro Dell' Orco, Mitigating Traffic Congestion: Solving the Ride-

Matching Problem by Bee Colony Optimization, Taylor & Francis, Vol. 31, No. 2, April 2008, pp. 135-152.

[78] Dervis Karaboga, Bahriye Akay and Celal Ozturk, Artificial Bee Colony (ABC) Optimization Algorithm for Training Feed-Forward Neural Networks, Modeling Decisions for Artificial Intelligence, Springer, Vol. 4617, 2007, pp. 318-329.

[79] Dervis Karaboga and Bahriye Basturk, Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems, Modeling Decisions for Artificial Intelligence, Springer, Vol. 4529, 2007, pp. 789-798.

[80] Dervis Karaboga and Bahriye Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Jurnal of Global Optimization, Springer, Vol. 39, No. 3, November 2007, pp. 459-471.

[81] Mustafa Servet Kiran and Mesut Gunduz, A Novel Artificial Bee Colony-Based Algorithm For Solving the Numerical Optimization Problems, ICIC International, Vol. 8, No. 9, September 2012, pp. 6107-6121.

[82] Dervis Karaboga, Selcuk Okdem and Celal Ozturk, Cluster based wireless sensor network routing using artificial bee colony algorithm, Wireless Networks, Springer, Vol. 18, No. 7, October 2012, pp. 847-860.

[83] Dervis Karaboga, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Vol. 42, No. 1, June 2014, pp. 21-57.

[84] Teerawat Issariyakul and Ekram Hossain, Introducation to Network Simulator 2, Springer, e-ISBN 978-1-4614-1406-3, 2012.

[85] Kevin Fall, The ns Manual, The VINT Project, A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, November 2011.

[86] Saba Siraj, Ajay Kumar Gupta and Rinku Badgujar, Network Simulation Tools Survey, IJARCCE, Vol. 1, No. 4, June 2012.

[87] J. Ousterhout, Tcl: An Embeddable Command Language, Proc. USENIX Winter Conference, January 1990, pp. 133-146.

[88] A. Aho, P. Weinberger and B. Kernighan, AWK- A Pattern Scanning and Processing Language, Bell Labs, 1978.

Curriculum Vitae

Hanin Ali Almutairi
Hanin.Almutairi@unb.ca

EDUCATION

University of New Brunswick **2015**
Fredericton, New Brunswick, Canada

- Diploma in University Teaching

University of Taibah **2008**
Medina, Saudi Arabia

- Bachelor in Computer Science

EXPERIENCE AND PUBLICATIONS

- H. Almutairi, J. DeDourekand P. Pochec, "Dynamic Node Movement Control in a Mobile Medium Ad hoc Network", The Seventh International Conference on Emerging Networks and Systems Intelligence, EMERGING 2015, July 19 -24, 2015 -Nice, France
- H. Almutairi, "Dynamic Node Movement Control in a Mobile Medium Ad hoc Network Poster", Research Exposition, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, 10th April, 2015.
- Hanin Almutairi, Hawra Alseef and Nada Alsalmi, "WIRELESS NETWORK MOVEMENT SIMULATION in NS2", Research Exposition, Faculty of Computer Science, University of New Brunswick, Fredericton, Canada, 27th March, 2013.
- Leader of the Saudi Woman Students Association in Fredericton, NB, Canada, from 2012 to 2015.
- From POSL to d-POSL: Making the Positional -Slotted Language Defeasible- semantic web techniques, 2011, http://d-posl.wikispaces.com/.
- Windows Server, NetSchool, and Network Maintenance Trainer, 70 hours Diploma for teachers in the Supervision and Guidance Center for Training and Development, Ministry of Higher Education, Medina, Saudi Arabia, 2010.
- Computer Trainer, New Horizon Institution for Training and Development, Gulf Company, Medina, Saudi Arabia, 2009-2010.
- Bachelor project, Intrusion Detection System, University of Taibah, Medina, Saudi Arabia, 2008.