BIOLOGICALLY INSPIRED ROUTE ACQUISITION PROCESS FOR AODV PROTOCOL IN MANET.

By

Tinsae Tadesse

A Thesis Document Submitted to the Department of Computer Science and Engineering,

School of Electrical Engineering and Computing.

Office of Graduate Studies

Adama Science and Technology University

October 2020

Adama, Ethiopia

BIOLOGICALLY INSPIRED ROUTE ACQUISITION PROCESS FOR AODV PROTOCOL IN MANET.

By

Tinsae Tadesse

Advisor: Ketema Adere (PhD).

A Thesis Document Submitted to the Department of Computer Science and Engineering,

School of Electrical Engineering and Computing.

Office of Graduate Studies

Adama Science and Technology University.

October 2020

Adama, Ethiopia

APPROVAL PAGE

We, the undersigned, members of the Board of Examiners of the final open defense by ***Tinsae Tadesse*** have read and evaluated her thesis entitled "***Ant Colony Optimization Based Route Acquisition Process for AODV Protocol in MANET.***" examined the candidate. This is, therefore, to certify that the thesis has been accepted in partial fulfillment of the requirement of the degree of Master of Science in Computer Science and Engineering.

Name	**Signature**	**Date**
Tinsae Tadesse		
Name of the Student		
Ketema Adere (Ph.D.)		
Advisor		
External Examiner		
Internal Examiner		
Chair Person		
Head of Department		
School Dean		
Post Graduate Dean		

DECLARATION

I hereby declare that, this M.Sc. thesis is my original work and has not been presented as a partial degree requirement for a degree in any other university and that all sources of materials used for the thesis have been fully acknowledged.

Name: Tinsae Tadesse

Signature: _____

Date: _____

This thesis has been submitted for examination with my approval as thesis advisor

Name: Ketema Adere (PhD)

Signature: _____

Date: _____

ACKNOWLEDGMENT

I would like to thank my thesis advisor Ketema Adere (Ph.D.) of the Network Science and Security Research Group Leader in the Department of Computing at Adama Science and Technology University. The door to his office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my work but steered me in the right direction whenever he thought I needed it. He also did provide me his unreserved time even after his work hours.

I would like to also thank individuals who participated and answered my questions in the Network Simulator 3 users' group page. The implementation of this research would not be a success if they did not provide the help I required.

Tinsae Tadesse

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ACO	Ant Colony Optimization
AODV	Ad hoc On-Demand Distance Vector
BAN	Body Area Network
DSR	Dynamic Source Routing
DTN	Delay Tolerant Networks
FV	Fairness Value
GA	Genetic Algorithm
IF	Irresponsible Forwarding
IP	Internet Protocol
MANET	Mobile Ad hoc Networks
MIB	Management Information Base
PAN	Personal Area Network
PSO	Particle Swarm Optimization
QORA	Quality of Service-Aware Routing Architecture
QoS	Quality of Service
RERR	Route Error
RREP	Route Reply
RREQ	Route Request
SDS	Stochastic Diffusion Search
VANET	Vehicular Ad hoc Networks

ABSTRACT

Ad hoc On-Demand Distance Vector (AODV) is one of the widely used protocols within the category of Mobile Ad hoc Networks (MANETs). Unlike the Dynamic Source Routing (DSR) protocol, AODV provides a mechanism to know newness of a route. Furthermore; AODV has the ability to know and remove outdated routes. However; its limitation to consider and use node level parameters and its failure to utilize alternative routes were motivations for this research.

Our objective was to study network parameters that can be used in the best route selection process, propose a generalized form of node quality metric, and use the ant colony optimization algorithm to provide an alternative route. But, due to our limited timeframe we were forced to focus on addressing AODV's limitation to utilize node level measurements when choosing the best route. For the sake of simplicity, we have measured a single node-level property, namely the remaining energy, during our implementation.

The implementation was simulated using the Network Simulator 3.30 software. The performance of our proposed method was compared with AODV using metrics such as average end-to-end delay, average throughput, and average packet delivery ratio. Simulation results indicated that our method creates a higher average end-to-end delay between communicating pairs. However; it is the best mechanism to measure path quality using a collection of node level measurements. In more stressful situations, our method has a 1% advantage over AODV in-terms of average packet delivery ratio.

The key finding from this research was that the tradition method of measuring path quality was not the best. Instead of using the summation of individual node measurements, it was found to be efficient when the product of individual values were used. The product method of measuring path quality was proved, mathematically and using simulation, to be better than the summation method.

Keywords: Ant Colony Optimization, Ad hoc On-Demand Distance Vector, End-to-end path quality, Multi-path routing, Mobile Ad hoc Network.

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the Study

Mobile Ad-hoc Network (MANET) is coined from two words: Mobile and Ad-hoc. Ad-hoc networks are those networks that do not require pre-established infrastructure; whereas mobility refers to a state in which networked devices exist. Formally, MANET is a system of mobile wireless nodes that can create temporary network topology which allows them to communicate without using any preexisting infrastructure. [1] Depending on their area of coverage; ad-hoc networks can be classified as body area, personal area, local area, and wide area. Body area ad hoc network (BAN) is composed of wearable devices that are distributed over one's body, and the connectivity between them is called a body area network. Personal area ad hoc networks (PAN) are formed by connecting mobile devices carried by individuals. PAN is all about the interconnection of devices around an individual, while BAN is the interconnection of devices on an individual. Local area ad hoc networks provide a mechanism to automate homes and offices. [2] Based on the number of nodes within a region; MANETs can also be categorized into small, moderate, large, and very large scale groups of networks. Small scale network contains$\text{nodes} < 30$, moderate scale networks contain$30 \leq \text{nodes} < 100$, large scale networks contain$100 \leq \text{nodes} < 1000$, and a very large scale network has $\text{nodes} \geq 1000$. [3]

MANETs are characterized by dynamic multi-hop topology, limited bandwidth, and energy, low security, high data loss rate, high delay, and jitter. The dynamic topology change is the result of free and arbitrary movement design in MANET. Variations in distance between nodes, area of deployment, and the available power are also determinant factors for the dynamic topology change in MANET. Wireless communications are affected by multiple access mechanisms, signal fading, and interference between signals. These properties restrict MANETs from using the available bandwidth to the maximum level. Communicating nodes in MANET are battery-operated, which makes the issue of energy critically important. Furthermore, MANETs using wireless networks are highly vulnerable to security threats. [4, 3] Despite being infrastructure-independent, MANET nodes can perform routing functions. As a result, nodes in MANET can be seen as combinations of routers and hosts. [4]

This research is a typical example of basic research which adds value to the existing scientific knowledge. However; MANETs that implement AODV are applied in various areas such as military, interplanetary communication, vehicular communication, search and rescue, disaster recovery, etc. [5] In the military, operations take place in places where there is no or untrusted infrastructure. The requirements are relaying information from the battlefield to the headquarters, coordination between groups of soldiers, military vehicles, and different operational groups. Minefields can be covered with mines that are equipped with radio technology. Mines periodically send a signal to ensure the existence of all nodes in the network. The absence of a response indicates a territory breach that would be reported to a remote station. [6]

Previously the Internet was limited to objects that are on earth, however; recent research developments are thriving towards interplanetary internet connectivity. If we consider the communication between satellites, shuttles, and space stations; distance, delay, signal attenuation, and interference leads to an impossible end-to-end communication. The only feasible solution seems to have multi-hop communication between devices that are in range. Such networks are collectively called Delay-Tolerant Networks (DTN). DTN allows networked devices to start communication when nodes with links are available. [5] By nature, such type of network formation exhibits the property of demand-based MANET routing.

Vehicular communication is being highly demanded by intelligent transportation systems. Vehicles equipped with wireless capabilities establish communication with others or the road infrastructure around them. Due to the current high interest in driving safety, accident and congestion warning systems; such networks are separately classified as Vehicular Ad-hoc Networks (VANET). Sensory data gathered from vehicles can be sent to a center or exchanged with other vehicles in the area so that the event could be known. Since such implementations require multi-hop routing, VANETs can be categorized as one application area of MANET. [5]

As stated by Y. Feng [7], AODV's main problem is its flooding mechanism used for route discovery and route maintenance. Such behavior requires a huge number of control-packet to be exchanged, which in turn creates a higher routing load. Reducing routing load and making a communication protocol resistant to route failure, by providing alternative routes and eliminating flooding property, will improve AODV's performance for use in any

domain. Hence; this research contributes to the theoretical foundation of the AODV protocol, which could be applied to any of those application areas.

1.2. Statement of the Problem

Starting from its early days, AODV has got the focus of many researchers. Many researchers have been comparing the performance of AODV with its equivalent routing protocols. In their performance comparison of AODV and DSR, the researchers found AODV having a high routing load. This higher routing load in AODV was around 74% more than that of DSR. If we compare only the route request (RREQ) packets, AODV generated around 6 times more route request packets than DSR. This clearly shows RREQ packets generated in AODV needs to be minimized so that its performance can be improved. [8] To reduce the amount of RREQ packet generated; alternative route searching algorithms, that can produce optimized results, should be studied.

In this research two issues of the routing and route acquisition process had been studied.

- Using a network performance metric-centered route acquisition process instead of a simple broadcast.
- Balancing the load on a route using an alternative path.

The high routing load was mainly the result of the RREQ transmission mechanism used in AODV. Similar to that of DSR, AODV uses flooding (broadcasting) as a route discovery mechanism. This flooding behavior has been manipulated to perform security attacks in MANET. Furthermore, the flooding mechanism is not fit for the low bandwidth and energy requirements of MANETs. Even after a route has been established, it may fail due to various reasons such as mobility. In this case, the reinitiated RREQ transmission will create additional congestion on intermediate nodes. Hence; it is important to be able to guide the route acquisition process in a way that can improve the overall network performance.

C. E. Perkins et al. [8] stated the other problem observed in AODV as its limitation to maintain alternate paths to a destination. AODV maintains a single route per destination, which becomes problematic if that route fails. If a route to destination fails, AODV reinitiates the route discovery process. However, this creates additional overhead to the existing network traffic of intermediate nodes. On the other hand; its counterpart, DSR, benefits from having many alternate routes to a destination by minimizing the flooded route discovery packets. After all of the flooded RREQ packets over the entire network, the return

is a single route to a destination; which is unfair and not feasible. Hence; nodes should be able to maintain multiple paths.

1.3. Objectives of the Study

1.3.1. General Objective

The ultimate goal of this research was to modify the route acquisition mechanism of the AODV routing protocol using a biologically inspired mechanism.

1.3.2. Specific Objectives

Specifically, the following points were targeted to achieve the general goal:

- Model route path quality from node level measurements,
- Propose multipath route discovery mechanism using the Ant Colony Optimization algorithm,
- Simulate and evaluate our proposed approach.

1.4. Research Question

To solve the problems stated in section 1.2, we have raised three research questions. These research questions were used to guide the research so that the problems can indirectly be solved. The research questions raised are:

- How can a MANET protocol be modeled to include various network parameters?
- How can biologically inspired algorithms be used to emulate the routing process in AODV?
- How can we evaluate the performance of the new approach?

1.5. The Methodology of the Study

The employed approach is used to answer the research questions stated in section 1.4. In this research, a literature review and experimental simulations were used as the main methods. A literature review was conducted to understand how mobile ad hoc networks could be modeled using graph theory, and various mobility models were also studied to assess their suitability for MANETs. Algorithms that are based on theories from biology and various fields in science are studied. From these algorithms, researches that are based on ACO and Stochastic Diffusion Search (SDS) were further reviewed. The Ant Colony

Optimization algorithm was used in our proposed algorithm. The reason behind this choice is discussed in section 2.5 of chapter 2.

Mathematical foundations and the way data is stored/exchanged are also be described in the proposed algorithm. Since the network was modeled using a graph, link weights between two nodes need to be calculated. Link weight is a local measurement of the link quality between two nodes, and it is calculated using link properties such as signal strength, congestion, and remaining energy level. However; in our model, we used weights to represent node level measurements. On the other hand; the Pheromone level was used to measure route quality between source and destination. Since a route is a combination of multiple links, the Pheromone value is also dependent on individual link weights between the source and destination. Data, about a route or link quality parameters, in networking should be exchanged between communicating nodes. This information should also be stored for future use. The way of exchanging and storing data in our proposed algorithm is also described in chapter 4.

E. Weingartner et al. [9] compared the Network Simulator (NS) 2, 3, Objective Modular Network Testbed in C++ (OMNeT++), SimPy, and Java in Simulation Time (JiST) network simulation tools. Based on their comparison result of these tools on memory usage, NS-3 beats all its counterparts. However, comparisons based on computational time show JiST and NS-3 being the best tools for network simulation. NS-2 on the other hand; is used by most researches and provides most of today's networking protocols for free. But on the contrary, NS-3 has a limited set of routing protocols. Despite NS-2's well-established sets of protocols available, we used NS-3 in the research as a simulation tool. The following table shows how NS-3 outperforms other simulators in-terms of memory usage and computational time.

Table 1.1 Performance comparison of network simulators. [9]

Performance Metric	Comparison Result
Memory Usage	NS-3 < OMNeT++ < SimPy < NS-2 < JiST
Computational Time	JiST < NS-3 < OMNet++ < NS-2 < SimPy

As stated in [4], throughput, delay, and route acquisition time can be used to measure the performance of a MANET routing protocol. When measuring performance, the context of the network should also be considered. Network context refers to parameters such as network size, topological rate of change, and mobility model. To measure the efficiency of our algorithm, the average throughput, end-to-end delay, and packet delivery ratio were measured. The following figure illustrates the process used in this study.

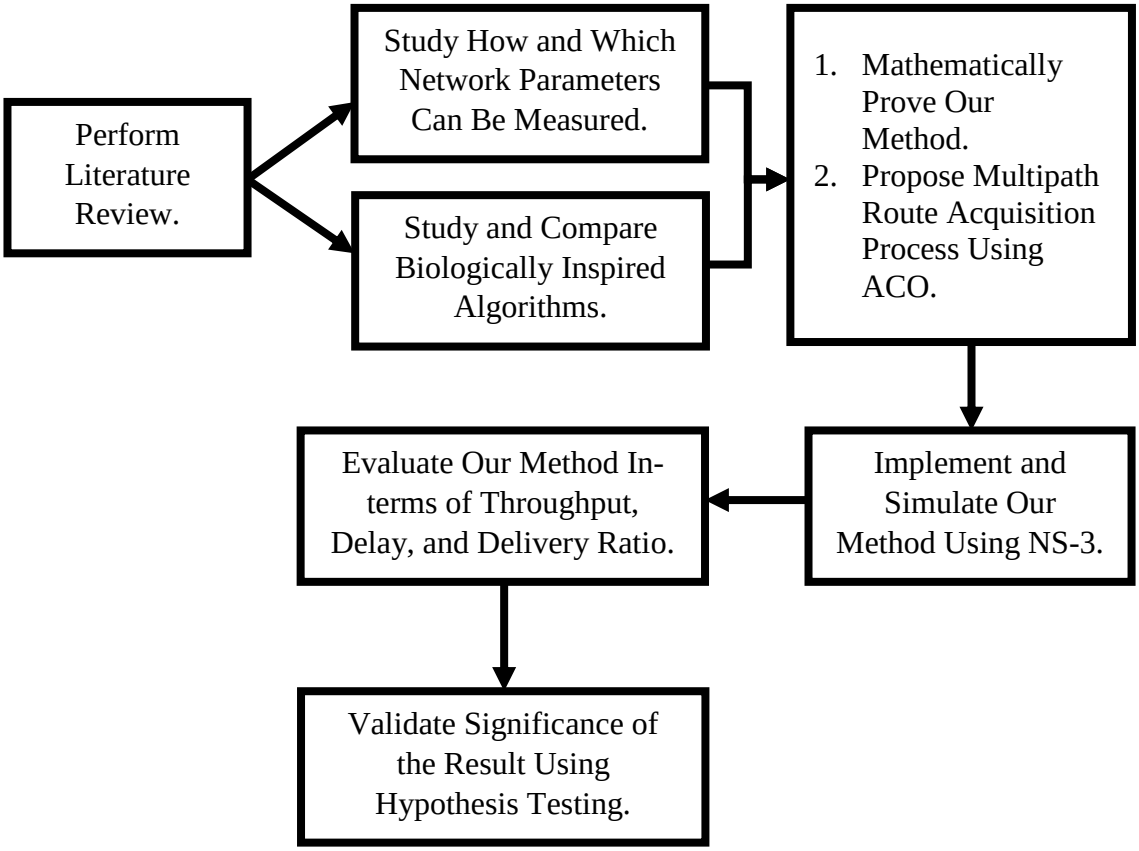


Figure 1.1 Method employed in the research.

Results from our simulations were tested for significance using t-test. The average end-to-end delay and packet delivery ratio were the metric tested.

1.6. Scope and Limitations

Several biologically inspired algorithms imitate various natural processes. Studies that use these algorithms have been conducted by many researchers too. However; as the research is constrained with time, it was focused only on a single biological algorithm. This is the Ant Colony Optimization (ACO) algorithm. The criteria for selecting this algorithm is stated in section 2.5. ACO is ideal to use in this research because of its distributed nature, the

inclusion of search and optimization techniques together, and it does not require preexisting global knowledge. However; future works might include features from other biological algorithms that are not considered in this research. Even though routing, in general, involves route acquisition, best route selection, and route repairing; this research focuses on route acquisition and best route selection processes only. Furthermore; the implementation only attempts to solve the first research problem. Hence; the simulation results do not implement a multi-path routing scheme.

1.7. Organization of the Thesis

This document is divided into six chapters. The next chapter briefly introduces concepts from other researches that served as a base for this research. Sections in chapter 2 give details on characteristics and application areas of MANETs, some of the available routing protocols in MANET, and popular algorithms that are based on various concepts from science. Chapter 3 discusses the approaches used by other researchers and the limitations of their work. Sections in chapter 3 are divided into two parts: the first deals with solutions that are based on ants' behavior, and the other part presents how researchers used various techniques to improve AODV. Chapter 4 presents our alternative for the stated problems in this chapter. Sections in this chapter systematically take us from measuring a node's quality to the choice of the best path using two approaches. The fifth chapter outlines the performance of our proposed approach. Finally, chapter 6 summarizes the steps involved, results, future works that follow this research.

CHAPTER TWO

2. LITERATURE REVIEW

2.1. Routing in MANET

Routing protocols in mobile ad hoc networks are expected to have properties such as demand-based or proactive modes of operations. The selection of the appropriate operational method is based on the available bandwidth and energy resources. [4] Proactive routing protocols can be also called table-driven routing protocols. Proactive routing protocols preserve route entries, to every other node, in the routing table. Such networks can minimize the time spent on route discovery, however; maintaining the routing table creates additional overhead. Furthermore, such routing mechanisms might never use most of the routing information stored. [10] As their name implies, Proactive (table-driven) routing protocols extensively use one or more tables to store routing information and adapt to the changing network topology.

One expectation of MANET was to let routing algorithms be adaptable to the traffic pattern. [4] As a result, the network becomes more intelligent and operates based on demand. Demand-based routing protocols initiate the routing process only when the source needs to transmit data. [11] The process of identifying a route from source to destination is called route discovery. Demand-based routing protocols are good for effective energy and bandwidth utilization. However; applying such a technique comes at the cost of delay that comes from the route discovery process.

Even though MANET routing protocols are classified as demand-based and proactive, some [12] also include geographic routing into the categories of classifications. Geographic routing protocols use localization information before establishing paths. Intermediate node selection is based on the position of the destination node. Such routing mechanisms are capable of reducing the amount of routing load in the network as they are destination-oriented. However, global location knowledge is required from nodes that support such protocols.

Several routing protocols can be categorized under each of the three groups, the following figure illustrates those common and selected ones.

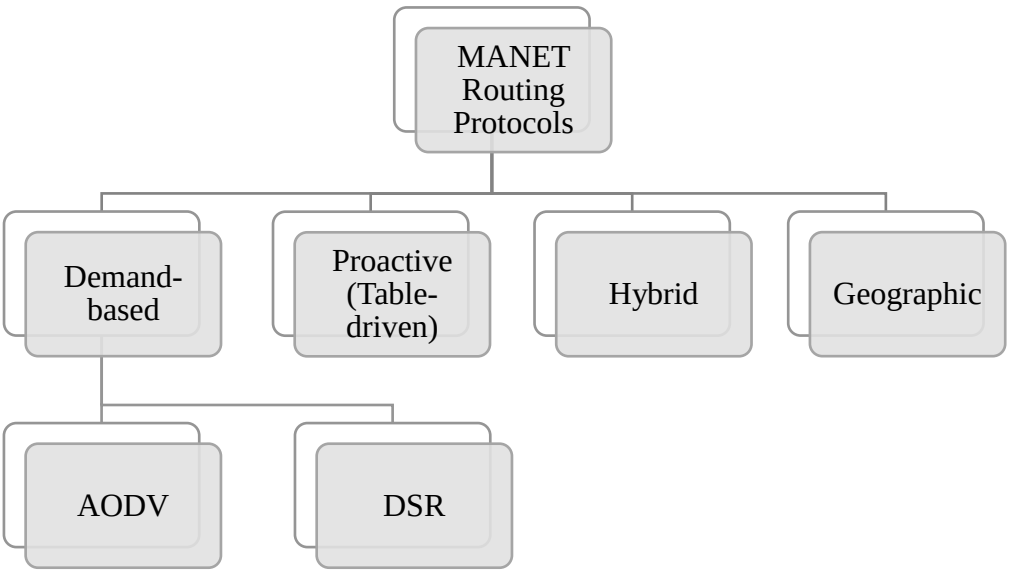


Figure 2.1 Classification of routing protocols in MANET. [10, 12]

2.2. Demand-Based Routing Protocols in MANET

There are some routing protocols, under the two categories, used in MANET. However, our focus is concentrated on demand-based routing protocols. Specifically, the Ad-hoc On-demand Distance Vector (AODV) routing protocol is the one that we studied. It is also worthy to know how AODV differs from its counterparts. As a result; not only AODV but other demand-based routing algorithms are also discussed in this section.

2.2.1. Dynamic Source Routing (DSR)

DSR is also a demand-based routing protocol that relies on the concept of source routing and route caching. DSR follows two steps of operation; those are route discovery and route maintenance. The moment a node has a packet to be transmitted to the destination, it checks its route cache. If a route could not be found, then it initiates route discovery by broadcasting the RREQ packet. Every intermediate node receiving the RREQ packet adds its address into the packet and then forwards it. A route reply is generated if the destination is reached, or an intermediate node with a route cache to a destination is reached. A destination node, generating the RREP packet, copies the sequence of intermediate nodes from RREQ into the RREP packet. As a result, the RREP packet contains the complete route between source and destination.

Upon link break, RERR and acknowledgment packets are used to maintain the route. RERR packets are generated when a node runs into a transmission problem. As a result, the node with a failed link gets removed from all route cache entries of the next-hop node. On the other hand, acknowledgment messages are sensed by neighboring nodes to verify whether the link to a node is still working or not. This way a node in a route can check if links are still operational. [13]

2.2.2. Ad hoc On-demand Distance Vector (AODV)

AODV is an on-demand or reactive routing protocol, which means the route discovery process is activated by a source node when it needs to send data to a destination node for which it has no routing information in its routing table. By the time a source node needs to send data, it initiates the route discovery process by sending route request (RREQ) packet. Nodes receiving the RREQ packet rebroadcast it until it reaches the destination node or a node with the route to the destination. During the propagation of the RREQ packet, each node records the neighbor node into the routing table from which it received the RREQ packet. This mechanism lets intermediate nodes establish a reverse path to the source.

Once the RREQ reaches the destination or a node with the route to the destination, it responds using a unicast route reply (RREP) message to the neighbor node from which the RREQ was received. As the RREP packet traverses back on the reverse path towards the source node; each intermediate node records its neighbor, from which it received the RREP, into the routing table. This route is called the forward route. Once a complete route is set up between source and destination, data exchange can proceed. However; if a link between intermediate nodes fails, the neighbor node immediately generates route error (RERR) packet and propagates it towards the source node. A node periodically uses the HELLO message to be certain that neighboring nodes are in its transmission range. [14]

According to C. Perkins et al. [15], the RREQ message is composed of 32 bits of route request id (RREQ ID), destination IP address, destination sequence number, source IP address, and source sequence number. It also reserves 8 bits for packet type and hops count; and 5 bits for several flags. Similarly; RREP packets of AODV contain 32 bits of destination IP address, destination sequence number, source IP address, and route lifetime. It also reserves 8 bits for packet type and hop-count; and 2 bits for two flags.

The following figures show the structure of the RREQ and RREP packets.

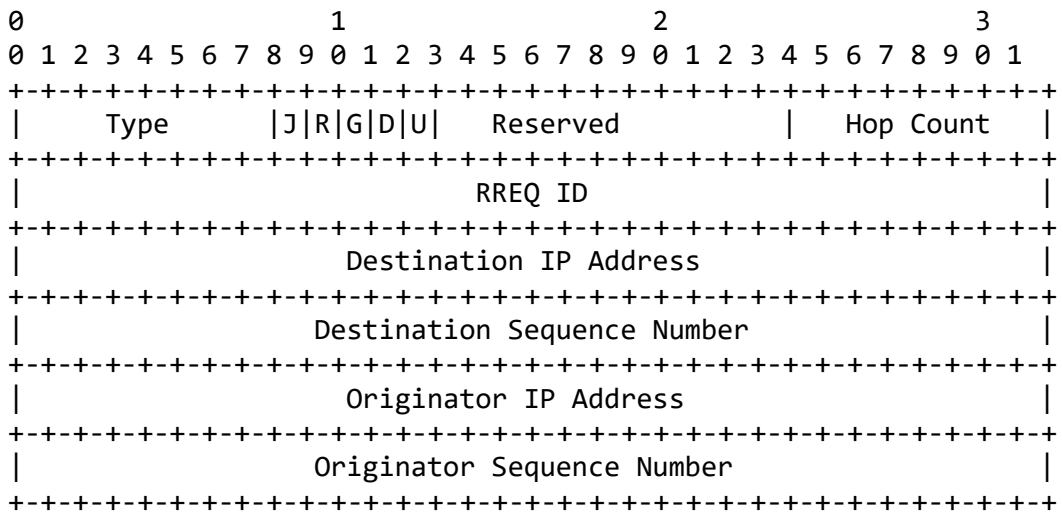


Figure 2.2 RREQ message format. [16]

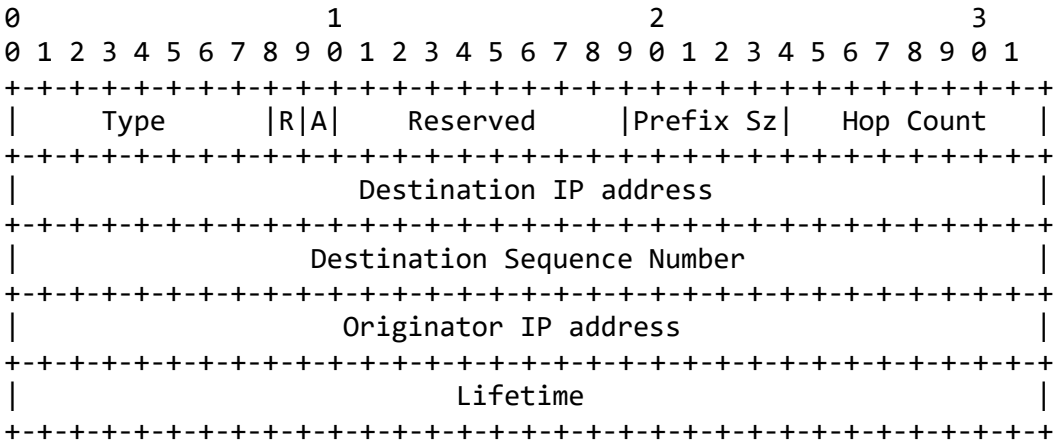


Figure 2.3 RREP message format. [17]

2.3. Modeling Wireless Networks

In their work [18], Nieberg et al. modeled a wireless communications network using a graph. Let us represent the entire network of mobile nodes using an undirected, weighted graph. Graph $G = (V, A)$ represents the network topology where V is a set of nodes that are equipped with radio capability. The coverage area of a node $v \in V$ is represented byA_v. Let location of a node $u \in V$ is represented by the function $f(u)$, then vertex $v \in V$ can receive a message if $f(u) \in A_v$ is satisfied. In this case nodes $(u, v) \in V$ will have direct communication with each other. However; if$(A_n \cap A_v) \neq \emptyset$ holds and another node is in the intersection region, then the hidden terminal problem may happen. In doing so, we can model the wireless network using a graph structure. However, communication graphs have

link weights. A weight, according to [18], is an indicator of a node's capability to perform its activities. Link weights could be calculated based on properties such as signal strength (distance), available energy, memory, and processing capacity, or the number of neighbors. The following figure shows an undirected and unweighted graph generated from the wireless network.

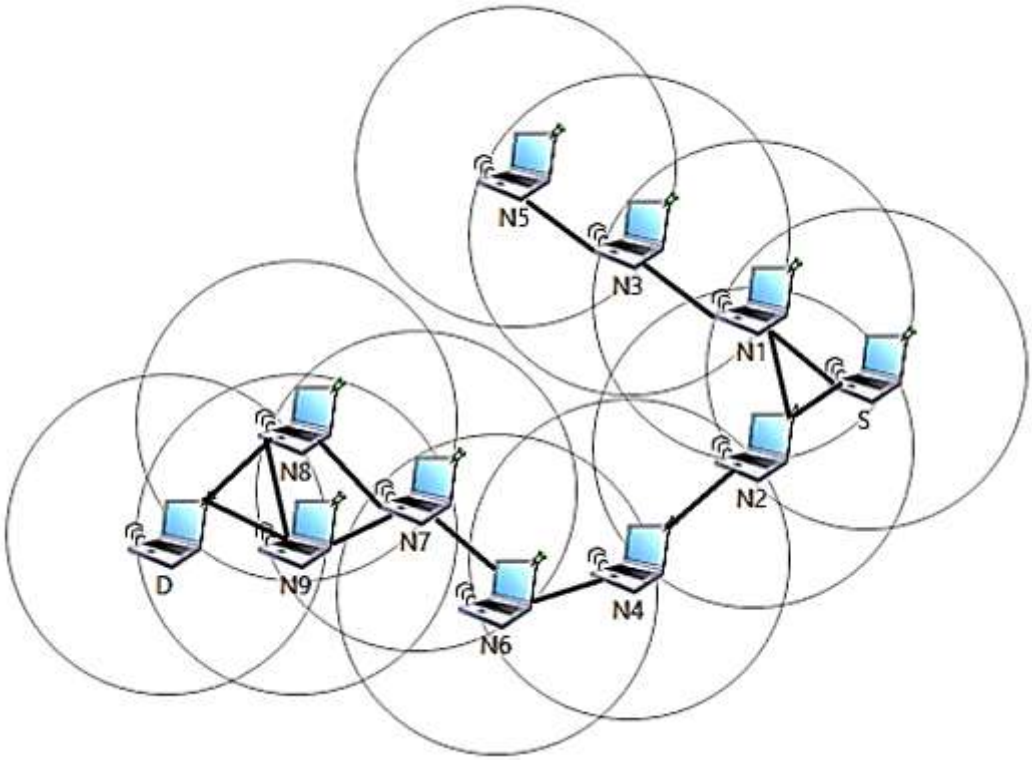


Figure 2.4 Wireless network graph.

A graph modeled from communicating wireless devices may not be static; rather due to mobility, the network topology could dynamically change from time to time. Such dynamic behavior is modeled using mobility models. Mobility models try to imitate the movement of a real mobile node; and they are generally categorized as node and group mobility models. Node mobility models try to mimic movement patterns of individual nodes; while, group mobility models try to mimic movement patterns of a group of nodes. The use of mobility models in an ad-hoc network is helpful to simulate problems caused by the movement of nodes. It is also useful to examine the appropriateness of a protocol for a particular scenario. There are several node mobility models for MANETs; some are Random Walk, Probabilistic Random Walk, Random WayPoint, and Gauss-Markov. On the other hand; Nomadic Community and Reference Point Group are some of the group mobility models available for MANETs. [19]

Random Walk [19] can be described as a movement where a node chooses its next location by randomly selecting a direction and speed. Movement in a random walk is free for selection of direction $\theta(t)$ and speed $v(t)$ values$[0,2\pi]$, $[v_{min}, v_{max}]$ respectively. A node in random walk motion chooses its next direction and speed after moving for a constant time t or distanced. Such a mobility model was developed to simulator the unpredictable movements of objects.

Probabilistic Random Walk [20], is sometimes called Markovian Random Walk because it is based on Markov's chain model. At a point of time, a node has three states: state 0 representing the current position, state 1 representing the previous position, and state 2 representing the next position. In this model, a movement to the next/previous step is determined by probability. The movement from a state to another state is based on a selection from a probability matrix. With the assumption of nonstop mobility, the probability of following the same direction is higher than changing direction for a node. The probability of moving to the next state without passing through the current state is zero. And the probability of retracing is rare. The following figure and matrix represent the above description of probability distributions:

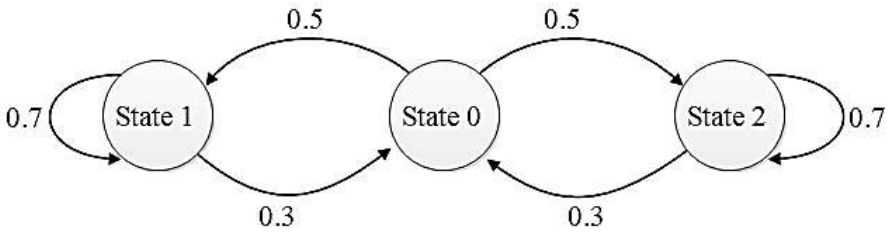


Figure 2.5 Markov's chain of probabilistic Random Walk. [19]

$$P = \begin{bmatrix} P(0,0) & P(0,1) & P(0,2) \\ P(1,0) & P(1,1) & P(1,2) \\ P(2,0) & P(2,1) & P(2,2) \end{bmatrix}, P = \begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.3 & 0.7 & 0 \\ 0.3 & 0 & 0.7 \end{bmatrix}$$

Where probability matrix $P(i,j)$ represents the probability of moving from state i to$j$. The probability matrix interpreted as $P(0,0)$, is the probability of staying in the same position, $P(0,1)$ and $P(0,2)$ indicate the probability of moving from the current state to the previous/next state respectively. $P(1,0)$ And $P(2,0)$ indicate the probability of retracing to the previous state from past/future states respectively. $P(1,1)$ And $P(2,2)$ indicate the probability of maintaining the same state from the previous/future states respectively. $P(1,2)$ And $P(2,1)$ indicate the probability of moving from previous to next state/next to

the previous state without passing through the current state. This implementation of a random walk produces a more realistic behavior than a simple random walk of nodes.

The Random Way Point mobility model can be described as a model that allows a node to move from its current position to a new location by randomly choosing its destination coordinates, its speed, and the amount of time that it will pause at the destination. For node j, P_i^j represents its mobility pattern using a random waypoint model for a period i. Variables i and j indicate a discrete unit of time for the movement of a node and a particular node, respectively. Random Way Point mobility model provides a pause time between the changes in direction and speed. If we suppose that a node selects its new speed V_i to move from P_{i-1} to P_i and a pausing time $T_{p,i}$ randomly. As a result, the entire motion can be represented as:

$$\{(P_i, V_i, T_{p,i})\}_{i\in N} = \{(P_1, V_1, T_{p,1}), (P_2, V_2, T_{p,2}), (P_3, V_3, T_{p,3}) \dots\} \quad 2.1$$

The value of pausing time and speed is chosen randomly using functions$f_{T_p}(\tau_p)$ and $f_{V_p}(v)$ respectively, where$(0 < \tau_p < \infty)$ and$(v_{min} \le v \le v_{max})$. The random waypoint mobility model becomes similar to the random walk model if the pausing time is set to zero. We can also define the distance traveled by node j as:

$$\{L_i^j\}_{i\in N} = \{L_1^j, L_2^j, L_3^j \dots\} \text{ Where } L_i^j = |P_i^j - P_{i-1}^j| \quad 2.2$$

This is because the distance traveled by node j at a discrete-time i is dependent on the starting movement period $i-1$, which is also the endpoint of the previous movement. As a result, the average movement length of node j and the average length traveled by all nodes at time interval i becomes:

$$\lim_{m\to\infty} \frac{1}{m}\sum_{i=1}^{m}(l_i^j), \text{ and} \lim_{n\to\infty} \frac{1}{n}\sum_{j=1}^{n}(l_i^j) \quad 2.3$$

In the Gauss-Markov mobility model, speed and direction at the n^{th} time interval is dependent on the value at the $(n-1)^{th}$ interval and a random variableα. On each time interval n, movement is calculated based on the previous location(x_{n-1}, y_{n-1}), speedv_{n-1}, and directionθ_{n-1}. As a result, the current location is calculated by:

$$x_n = x_{n-1} + |v_{n-1}| \cos\theta_{n-1} \quad 2.4$$

$$y_n = y_{n-1} + |v_{n-1}| \sin\theta_{n-1} \quad 2.5$$

Where the speed and direction at a specific time interval n is:

$$v_n = x_{n-1} + |v_{n-1}| \cos \theta_{n-1} \quad 2.6$$

2.4. Biologically Inspired Optimization Algorithms

The following series of works done by researchers focus on developing a new algorithm from scratch. However, existing protocols such as AODV had their qualities and yet could be incorporated with biologically featured algorithms.

2.4.1. Stochastic Diffusion Search

Stochastic Diffusion Search [21] was introduced as a distributed (parallel), and probabilistic model (pattern) searching technique applied on a search space (problem). Sometimes there may not be a pattern to start with, in such cases, the goal becomes finding the pattern itself. SDS has three main stages: initialization, testing, and diffusion. [22]

During the initialization phases, each agent is inactive and randomly selects a hypothesis about the solution. If there is a piece of prior knowledge about the pattern to be searched, then this can be exploited and used to bias the search for the best solution. The hypothesis is a solution Tobe for the problem raised.

Testing involves the evaluation of the hypothesis. If the test result is a success, then agents become active and diffusion follows. However; if the test result does not succeed, agents remain inactive. If an agent's hypothesis is denoted by *h*, then partial function evaluation of this hypothesis becomes $pFE = f(h)$. The return value of this function is used to set the agent's activation state.

At the diffusion stage, active agents recruit other inactive agents and communicate their hypothesis. Each inactive agent starts by randomly choosing another agent. If the selection resulted in an inactive agent, then the hypothesis gets copied from the active to an inactive agent. (Diffusion of information) Agents generate a new random hypothesis otherwise. This lets more agents converge at good hypothesis estimations.

The following algorithm describes the steps involved in the stochastic diffusion searching algorithm.

Algorithm 1 Stochastic Diffusion Search

Initialization phase: Agents randomly generates hypothesis.

while (halting condition not met)

Test phase: Agents evaluate their hypothesis.

Diffusion phase: Agents exchange hypothesis evaluation results with peers.

end while

2.4.2. Ant Colony Optimization

In the research [23], an ant colony was used as a source of inspiration to develop a route optimization algorithm. Ants use a collaborated effort of the colony to find the shortest path between their nest and food source. During their movement, ants lay a chemical called pheromone which will be used as an indicator of a path towards their food. As a result, other ants prefer to follow a path with higher pheromone concentration. Due to this, the movement of ants will be controlled by pheromone concentration. This technique eventually attracts more ants towards the area of more pheromone levels, resulting in convergence to the shortest path. (As shorter paths can quickly lead to their food, are marked by more ants and have more pheromone level) The level of pheromone concentration along a path evaporates with time. If pheromone level concentration is represented by τ_{ij}^{k} and η_{ij}^{k} indicate heuristic information of the route, then the probability of ant k choosing a route to destination j, which is reachable through node i, can be calculated as:

$$P_{ij}^{k} = \frac{\left(\tau_{ij}^{k}\right)^{\alpha}\left(\eta_{ij}^{k}\right)^{\beta}}{\sum_{l\in N_i^k}\left(\tau_{lj}^{k}\right)^{\alpha}\left(\eta_{lj}^{k}\right)^{\beta}} \qquad 2.7$$

N_i^k Indicate a set of neighbors (alternate routes) to destination j through node i. Heuristic information collected by the ant is used in the selection of a route. Thus higher heuristic value increases the probability of a route being selected. However, a priori knowledge gained about the route such as the distance s_{ij} has often an inverse relationship with the probability of selecting that route $\eta_{ij} \propto \frac{1}{s_{ij}}$. This implies that shorter routes will be highly

likely to be selected due to their shorter traveling time. If ρ is the percentage of evaporation, then the pheromone level after evaporation can be calculated as:

$$\tau_{ij} = (1 - \rho) \times \tau_{ij} \quad 2.8$$

If k ants travel across that route after the evaporation, then the new pheromone level becomes:

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad 2.9$$

Where $\Delta\tau_{ij}^{k}$ represents newly released pheromone on the path from i to j by ant k. Such a mechanism requires ants to keep a record of each intermediate node they pass through, and this will increase the packet size if the network is very large.

2.4.3. Particle Swarm Optimization

The simulation-based research conducted by James Kennedy et.al. [24], was based on the concept of social information sharing between members of a community. Their simulation was mainly based on group behavior within a flock of birds. The main reason for communication among group members is to exploit the advantage gained from the discoveries and experiences of individuals. The first concept raised was the issue of Nearest Neighbor Velocity Matching. When a flock of birds flies, any two of them "cannot occupy the same position in space without colliding". As a result, they try to match their movement and create synchronized motion. In a simulation, this can be achieved by assigning the velocity of any actor x to actor y. (For randomly placed actors x and y, x is the nearest neighbor of y)

The second concept introduced was Cornfield Vector. As described by them, flock members are capable of advancing knowledge of the flock. If one member of the flock spots a food (cornfield), then it is only a matter of time before all the other members are notified. As a result, each actor will evaluate its current position to its personal and the (global) group's best experience of the swarm. Let vectors P_i^k, P_g^k consists of the personal best and global best evaluations at each particular interval k, respectively. Evaluation of current position for personal best evaluation is represented by $P_i^k - X_i^k$, where X_i^k is the current position of actor i and is extracted from its components$[x_i^k, y_i^k]$. Similarly, $P_g^k - X_i^k$ represents an evaluation of the current position from the global best solution point of view. To make the system stochastic, let say each actor tends to move randomly. Then actors' movement will be

influenced by random amountsr_1, r_2 toward a personal and global best solution respectively. Since actors are attracted towards both personal and global best solutions, the level of attraction to these solutions can be represented by c_1 andc_2. Furthermore, as described in [25] there are two properties of any metaheuristic algorithms: exploration and exploitation. Exploration refers to a thorough search in the search space to generate more new solutions. Exploitation, on the other hand, refers to the use of local information to generate better solutions. These two properties should be balanced to achieve a good result. The movement of an actor at time $k + 1$ depends on its previous motion v_i^k and its inertiaθ^k. Therefore, the movement equation will become:

$$v_i^{k+1} = \theta v_i^k + c_1 r_1 \left(P_i^k - X_i^k\right) + c_2 r_2 \left(P_g^k - X_i^k\right) \quad 2.10$$

As stated in [26], the equation without the last two parts drives actors at the current speed and in the same direction. In this case, the actor will not find the target solution unless the solution itself is in the way. On the contrary, the equation without the first part will prevent actors from searching for new solutions and are only limited to local searches. So, the discovery of new solutions may take generations. The position of actors is updated using the function:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad 2.11$$

2.4.4. Genetic Algorithm

The genetic algorithm, as described by [27], is a search and optimization technique based on natural selection and natural genetics theories. He aimed to improve the robustness of artificial systems so that costly operations can be removed. When the issue of robustness and performance is raised, there is no better system to call than nature. For solving optimization problems, a genetic algorithm represents a population of solutions as a set of chromosomes. Each chromosome represents a solution to the problem; hence, it passes through crossover, and mutation stages to achieve a better-fitted solution.

The first step in the genetic algorithm is to represent possible solutions in the form of chromosomes. Such representation eases applying the appropriate genetic operation on potential solutions. A naïve genetic algorithm represents each solution using binary strings. Such representation is called direct encoding. However, many real-world problems are too complex to be represented using binary strings. Hence in the indirect encoding method, only essential parts of a solution are represented in a chromosome.

The second step of the genetic algorithm is to evaluate the fitness of each solution (chromosome). This stage is also called a reproduction stage. During this stage, each solution is evaluated against a fitness function f, and those with higher fitness value will have a higher probability of surviving until they produce the next generation. This, according to Charles Darwin in his evolution theory, is called the process of natural selection or "survival of the fittest". Once the fitness of each solution is evaluated, their probability of being selected can be determined by the equation:

$$P_i = \frac{f_i}{\sum_{i=1}^{n} f_i} \qquad 2.12$$

The final step followed by the genetic algorithm is applying genetic operations (such as crossover and mutation) to produce better solutions (offspring). Crossover operation produces a new solution (offspring) by joining solution (chromosome) features starting at a particular location. A mutation, on the other hand, produces random changes in various sections of a chromosome. Mutation operation is useful to create diversity among the chromosomes.

2.5. Summary

As stated in [28], a search can be defined as an act of finding something. However, the objective of the search conducted can be two: finding a solution (a question of existence) and/or finding a good solution (a question of optimality). As a result, these two questions need to be answered for any routing algorithm in MANET. The following table shows a summary of algorithms studied in this research which are based on biological theories.

Table 2.1 Summary of biological algorithms.

Algorithm	Concept	Reference
Stochastic Diffusion Search (SDS)	Distributed Searching and Knowledge Diffusion	[21], [22]
Ant Colony Optimization (ACO)	Distributed Searching and Learning from Others' Experiences	[23]
Particle Swarm Optimization (PSO)	Search towards Personal and Global Best Experiences	[24], [25], [26]
Genetic Algorithm (GA)	Have a Solution, Generate New solutions from the Best ones.	[27]

To choose the best-fit algorithm from those discussed above, one has to compare each algorithm with AODV. The comparison can be based on the following features, and these features exist either in MANET or AODV:

- The algorithm should start by answering the question of existence. This is because routing is based on the identification and optimization of routes between source and destination.
- The algorithm should not assume a pre-existing global view of the topology. If a global view were available, it would be used to find the shortest/fastest/maximum-flow… path. Several algorithms leverage such knowledge, such as Shortest Path First [29], and find a solution using a greedy approach. Unfortunately, in MANET there is no such knowledge. This is because the topology is temporary and dynamic.

CHAPTER THREE

3. RELATED WORKS

3.1. Multi-Path Routing Using Ant Colony Optimization

P. Vijayalakshmi et al. [30] represented the underlying network using a directed graph and their objective was to find and use alternate routes during link failure. To achieve this all available routes should be listed first. At this stage, nodes having residual energy greater than the threshold value ($E_r(v_i) \geq E_{th}$) are selected from all of the available routes. Then routes through these nodes are evaluated to find the minimum energy cost path $\sum_{i=0}^{i=k-1} min\big(E_r(v_i) - c(v_i, v_{i+1})\big)$ where $c(v_i, v_{i+1})$ is the cost of transmission between the nodes. As a result, one optimal route will be selected to transfer data traffic. This involves the use of forwarding and backward ants. Finally, the next best path will be used as an alternate route in case of link failure. However, the complexity of this algorithm is high as packets are transmitted between the source and destination multiple times: 1) to find relative paths using the Max-Min-Path approach; 2) to find the optimal path among relative paths using forward and backward ants. [31] Furthermore; in case of link disconnection, a new source to destination path (from the relative paths) search is initiated to establish an alternative route using the Max-Min-Path approach. This creates overhead on the entire network.

C. Shubhajeet et al. [32] envisioned a multipath reactive routing protocol. Their work considered network properties that contribute to QoS such as congestion, route length (number of hops), and link-level metrics. The congestion level of a route is measured by considering nodes' buffer space and packet drop rate in the calculation. Nodes' buffer space is measured depending on the frequency of incoming traffic, i.e. high packet arrival rate results in a high buffer space sampling rate. Packet drop rate can be measured using the number of packets arrived at each node and the number of dropped packets. [33] The link connectivity level is measured using the Received Signal Strength which in turn is used to predict link break. Received signal strength is calculated using $G_r G_t S_t / \left(\frac{4\pi x}{\lambda}\right)^2$ and its ratio with a threshold value gives a Received Signal Strength Metric$0 \leq RSSM_i \leq 1$. These quality indicators are included in the forward and backward ants to measure the overall route quality from source to destination. In this scheme Pheromone value is a result of the number of hops, link metric, and congestion metric. One limitation of their work is the mechanics

used in the Pheromone update; where an update is dependent on the global mobility model parameter, the average maximum speed of nodes. Furthermore; their proposed work considers two modes of operations: sparse and dense modes. However; to choose the right mode, it expects explicit specification of the underlying network mode i.e. either sparse or dense network. As a result, it is not ideal for dynamic network topology as it is dependent on the user.

Al-ani et al. [34] proposed a QoS-aware routing architecture while using an ant colony optimization approach. This architecture has two components: QORA Entity and QORA SNMP Entity. The QORA Entity runs on each node in the network and it is responsible for finding available paths from source to destination. Furthermore; it is responsible for managing ants, maintaining neighborhood and route information, decision making, generating commands, and calculation of QoS parameters. Each of these tasks is controlled by five specialized components: Ant Management, Neighbor Table, Routing Table, Decision Engine, and QoS Manager. QORA SNMP Entity on the other hand; is responsible for collecting and retrieval of management-related information. It contains two components called SNMP Agent and Management Information Base. SNMP Agents collect detailed QoS related information about outgoing links of each node in the network. Management Information Base (MIB) is only a tree-based structure to store the collected QoS related link information. This protocol considers the following QoS parameters:

1) Bandwidth- since transmission speed of a path is limited by the minimum transmission speed link in that path.

$$B_{s,j,d} = \min\{\text{ifSpeed}(n), \forall_n \in P_{s,j,d}\} \geq B_{min} \quad 3.1$$

2) Delay- is the sum of propagation, transmission, and queueing delays.

$$D_{total}(n) = D_{prop} + D_{trans} + D_{queue} \quad 3.2$$

$$D_{s,j,d}(n) = \sum_{n \in P_{s,j,d}} D_{total}(n) \quad 3.3$$

3) Packet loss- is the ratio between the number of lost packets by the total number of packets received and sent. The number of received packets is calculated by summing all the incoming unicast, multicast, and broadcast packets. Similarly, the number of sent packets is calculated by summing all the outgoing unicast, multicast, and broadcast packets. However; lost packets are those either discarded or dropped due

to error or unknown protocol. If the packet loss ratio is represented by $L(n)$, success rate becomes $1 - \mathrm{L(n)}$. Hence; the success rate along a route becomes

$$\mathrm{S_{s,j,d}} = \prod_{\mathrm{n \in P_{s,j,d}}} \left(1 - \mathrm{L(n)}\right) \qquad 3.4$$

The major problem observed in this algorithm was its property of keeping all the traversed nodes in the stack. (Similar to that of DSR)

3.2. Summary

In general; a protocol is said to be a multipath routing algorithm if, it can construct multiple paths between a source and destination. These paths can be constructed by considering node level parameters (such as energy, packet loss rate, channel load, and buffer space), and network-level parameters (such as end-to-end delay, route length, and amount of load on a path). But, the mechanism used to measure these parameters between the source and destination can lead to accuracy issues. Furthermore; the above methods only target a certain network parameter, rather than providing a generalized method to represent path quality. The following table summarizes the observed gap on these literatures.

Table 3.1 Summary of reviewed literatures.

Measured Parameters	Strength	Gap	Reference
Energy	Provides alternative route.	Considers one parameter, uses the summation method.	[30]
Congestion, and Signal Strength.	Provides alternative route, uses the product method.	Measures the parameters separately, and adds additional fields for all of them.	[32]
Bandwidth, Delay, and Loss Rate.	Provides alternative route, uses both the product and summation methods.	Measures the parameters separately, and adds additional fields for all of them.	[34]

Our proposed method provides a generalized method of measuring path quality by considering multiple parameters. Furthermore; our proposed method uses the product method to measure quality of a path between a source and destination.

CHAPTER FOUR

4. THE PROPOSED ALGORITHM

This section presents the foundations and detailed flow of the proposed route acquisition mechanism. As stated earlier, our first research question was about measuring and incorporate different node/network parameters in the messages exchanged between nodes. Node and network-level parameters measured by nodes must be added to a packet header and exchanged between nodes. This allows nodes to be able to know how their neighbors are performing. Furthermore; nodes at both ends of a route will be able to know the quality of a path they are choosing to communicate through. Our second research question was about making AODV maintain multiple routes per destination using a biologically inspired approach. The following subsections of this chapter will provide an answer to those questions.

4.1. Node Level Parameters

The first step is to define node parameters to be measured. The remaining energy and packet drop rate are some of the parameters that can be measured. Let node i measure its local parameters of residual energy E_i, and the packet drop rate D_i.

$$E_i = \frac{E_i^{Residual}}{E_i^{Start}}, D_i = \frac{D_i^{Dropped}}{D_i^{Received}} \quad 4.1$$

Where $0 < E_i \,\&\, D_i < 1$. Node weight W_i, which shows the quality of a node for packet transmission, is calculated from E_i and D_i as

$$W_i = \alpha_1 E_i - \beta_1 D_i \quad 4.2$$

More generally it is summarized as:

$$W_i = \sum_{j=1}^{n} \alpha_j \, p_j - \sum_{k=1}^{m} \beta_k \, p_k \quad 4.3$$

For n network parameters that positively affect the weight, p_j represent the values of those parameters. Similarly, for m network parameters that negatively affect the weight, p_k

represent the values for those parameters. Furthermore; $\sum_{j=1}^{n} \alpha_j = \sum_{k=1}^{m} \beta_k = 1$ which are coefficients of each parameter.

Since nodes in MANET are capable of adapting to the change in aspects of the network and resources, there are two modes of operation. (Proactive and reactive) A node is in a proactive mode as long as its weight is greater than or equal to a threshold level as $W_i \geq W_{th}$. During proactive mode, the node can perform operations that demand high resource usage. If its weight is less than a threshold level as $W_i < W_{th}$, it switches to reactive mode. Operations in both situations are described in the next sections.

4.2. From Node to a Path

Once we have the node weight, it is possible to compute the total path quality. Path quality can be calculated either by computing product or sum/mean of all weights along a path. Consider the following network graph.

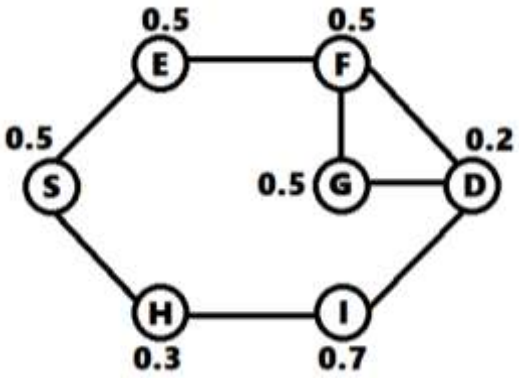


Figure 4.1 An example network graph.

Consider two paths $P_1 = \{S, E, F, D\}$ and $P_2 = \{S, H, I, D\}$. Both have similar length but different weights. Which path do you think is best? we would say P_1, because it is relatively stable. When using the mean approach, both paths are equal as $\frac{1}{n}\sum_{i=1}^{n} W_i^1 = \frac{1}{n}\sum_{i=1}^{n} W_i^2 = 0.5$. However; $\prod_{i=1}^{n} W_i^1 = 0.125$ and $\prod_{i=1}^{n} W_i^2 = 0.105$. As expected, the result from the product-method is better at depicting the difference in the two paths. What if the two paths have a different number of hops as $P_1 = \{S, E, F, G, D\}$ and $P_2 = \{S, E, F, D\}$. In this case, both paths are equal as $\frac{1}{n}\sum_{i=1}^{n} W_i^1 = \frac{1}{n}\sum_{i=1}^{n} W_i^2 = 0.5$. However; $\prod_{i=1}^{n} W_i^1 = 0.0625$ and

$\prod_{i=1}^{n} W_i^2 = 0.125$. Again the product method gives shorter paths a better rank. Hence; the product method is better than the sum/mean method in modeling path quality.

However; for large networks consisting longer routes, path weight (calculated using the product method) becomes very small. In this case, scaling each weight using a non-linear method may solve this problem to some extent. This can be done by applying $W_i' = 1 - (1 - W_i)^2$. Hence; path quality can be calculated as $\prod_{i=1}^{n} W_i'$. When applying this technique to the last example, $\prod_{i=1}^{n} W_i^1 = 0.3164$ and $\prod_{i=1}^{n} W_i^2 = 0.4219$. The value of path quality increased from 0.0625 to 0.3164 due to the scaling operation. This operation becomes very important when individual weights on a path are small. However; it is important to note that this method might also increase the end-to-end delay due to the additional computation.

4.3. Avoiding Route Loops

Consider a route with n nodes N_i between source and destination. Where $i = 1, 2, 3, \cdots, n$. If there is a loop in the route, then the path would look like $R = \{N_1, N_2, N_3, \cdots, N_n, N_2, \cdots\}$. If a source node was in the looping route, it can easily terminate the loop by simply looking at the source address. For values $0 < V_i < 1$ at each node i in the route R, the product of values up to the node $n > 1$ becomes $P_n = \prod_{i=1}^{n} V_i$. Note that $V_1 = P_2$, and product of values at each node in R looks like $V_1 > P_2 > P_3 > P_4 > P_5 \cdots > P_n$. Hence; an intermediate node N_2 can detect a looping situation as the new product of values is smaller than the previous $P_2 < V_1$. Hence; it is possible to identify a loop the moment a packet reaches its previously navigated node. It is also important to know that this technique can only be applied in a single route per single destination routing tables.

Consider a looping path $\{E, F, G, D, F\}$ from the previous network graph. Metric values V_i at each node are represented with W_i. Can a packet containing weight, be cycling the route infinitely? No. Because, the moment a packet reaches its previously navigated node, the product of weights in the loop $(0.5 \times 0.5 \times 0.5 \times 0.2) = 0.025$ is smaller than the previous weight recorded for the node E. (i.e. 0.5)

4.4. Routing Using the Ants' Way

Before diving deep into the operational details, we would like to note that the proposed algorithm is an extension of the current AODV protocol. Hence; the searching mechanism uses the expanding ring searching technique.

Route Discovery. When a source node wants to communicate with a destination, it first checks its routing table. If it could not find a route to the destination, it computes node level weight using the mechanism stated in section 4.1 and encapsulates this value in a route request ant. Finally, the route request ant is broadcasted to all neighboring nodes. The source also keeps track of its current weight until the next update. It should be noted that nodes will keep packets in a queue until a route to the destination is found. Recipient of a request ant that is 1 Hop away, will maintain only one route to their neighbor. When a node broadcasts a request ant to all of its neighbors, it puts its weight on the node weight and path weight fields. The following figure shows the newly added header fields in the request ant.

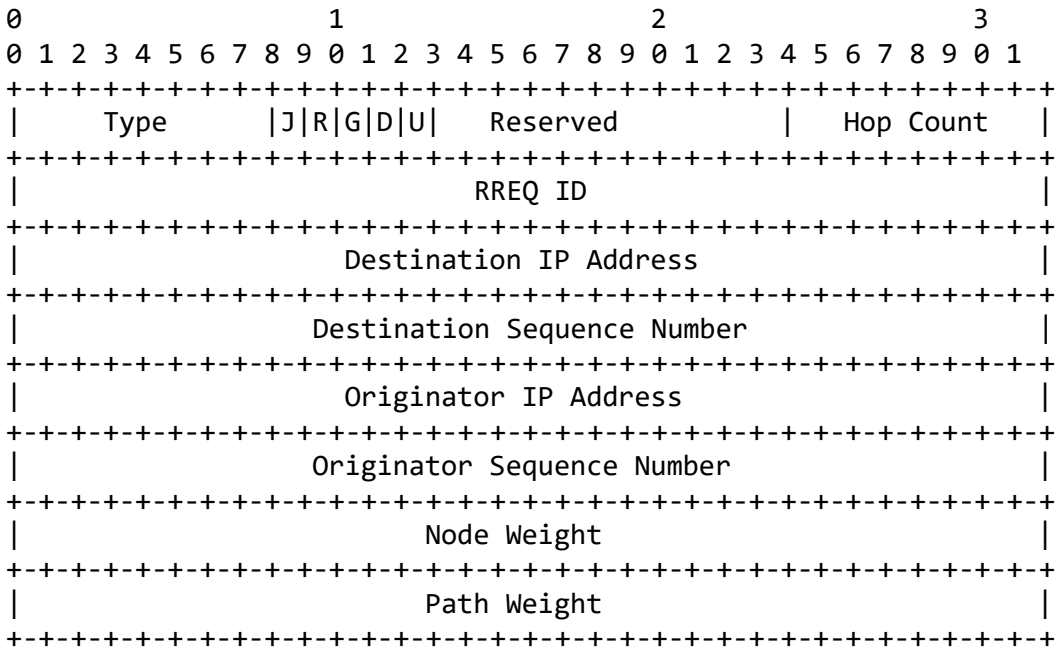


Figure 4.2 Request ant header fields.

An intermediate node, which received a request ant and is neither the destination nor the source, checks its routing table for the destination and the source. If no record is found, then the request-ant hadn't been received before and the node does not know the destination either. In this case, the node saves routing information towards the source from the ant. Then, the node computes route heuristic from the total weight $\eta_n^s = \prod_{i=1}^{n} W_i'$ and rebroadcasts the ant if the Hop count in the packet is greater than 2. If no record is found about the destination but the source is known, then the request-ant had been received before. In this case, an intermediate node behaves differently in the two operational modes. During the reactive mode, a node maintains one route per destination. Since there can only be one route, the pheromone level has no importance. When $\alpha = 0$, β loses its importance too; and

if no alternative neighborhood information is there (because the node is in reactive mode), $\eta^{s}_{n,d}$ from equation 2.8 becomes determinant of the routing decision. Hence; it compares the total weight (η^{s}_{n}) in the newly received ant with the recorded value in the routing table. If the total weight from the newly received ant is higher than the recorded value, or the Hop count from the newly received ant is smaller than the recorded value, it replaces the recorded route with values from the new ant and rebroadcasts the ant. During the proactive mode, a node maintains two routes per destination. When a new ant is received by an intermediate node, a route is created in the routing table if the total weight from the newly received ant is higher than the recorded route value**s** or Hop count from the newly received ant is smaller than both the recorded route value**s**. However; since there are two routes recorded and the weight from the newly received ant is lower than the best route but higher than the second, then the second route will be replaced. A routing table contains the following additional information:

- Total Weight
- Pheromone

Pheromone level of all routes are initialized from zero $\left(\tau^{0}_{i,j} = 0\right)$. When a node saves routing information from a request-ant for the first time, it saves the amount of pheromone laid by the ant as $\left(\tau_{n,d} = 0 + \Gamma_{n,d}\right)$.

Eventually; after repeating similar steps, the ant reaches its destination. Similar to an intermediate node, a destination node in the reactive mode maintains one route per destination and returns a single reply ant. This route is the one with the highest total weight. In the proactive mode, the node maintains two routes per destination. Hence; reply ants are sent to all the neighbors from which the request ant was received. Similar to request ants, routing information is stored when a reply ant traverses back to the source. One of the common fields in both the request and reply ant is neighbor weight, which represents the

quality of a neighbor for a transmission. The following figure shows the newly added header fields in the reply ant.

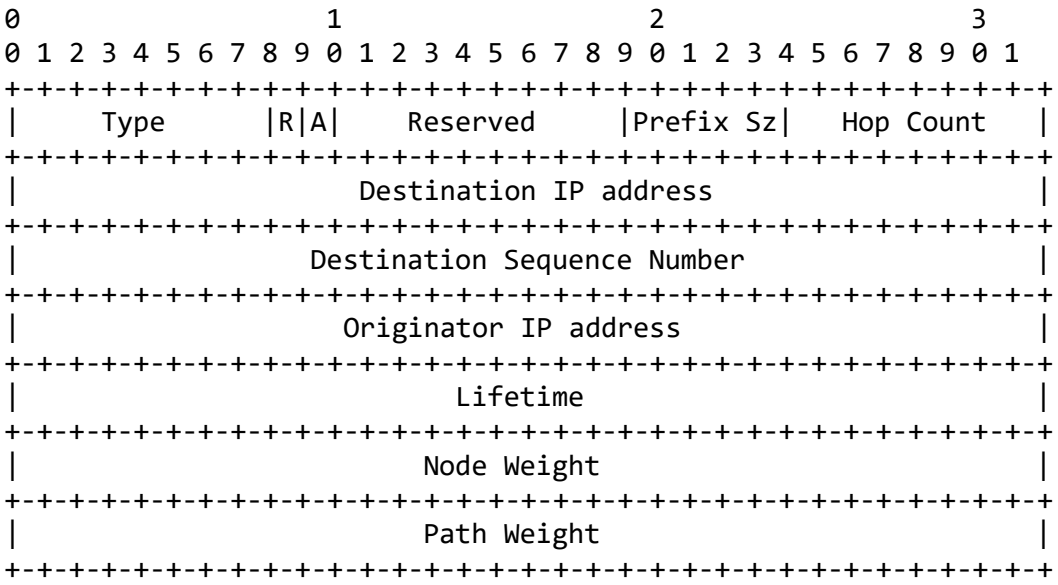


Figure 4.3 Reply ant header fields.

Data Transmission. After recording routing information about a destination from the received reply ant, a source node transmits its data in two ways. If a source is in reactive mode, it sends the data to the next node along the best route. This is because $\alpha = 0$, no alternative route is maintained, and the value of β does not matter in equation 2.8. Hence; the probability of the route is always one $(P_{n,d} = 1)$. However; in a proactive mode, the source computes the probability of each route using equation 2.8 and sends the data through the link with high probability. Intermediate and destination nodes that receive a data/control packet perform two operations. These nodes evaporate the pheromone level of all routes that lead to the source using equation 2.9 and update the pheromone value of the route through which the data/control packet was received using equation 2.10.

In proactive mode, balancing the network load can be achieved using the following mechanism. First, the node must decide when load balancing should be performed. Let $\{R_1, R_2\}$ be routes leading to a destination node where their order is based on probability $P_{n,d}$, and the currently active route is $R_a = R_1$. If the ratio between path weight of alternate and active route is less than a predefined value $\left(\frac{\eta_2}{\eta_1}\right) \geq \eta_{Th}$, then data transmission is safe to be routed through R_2 at any given time. Hence; data transmission is routed in such a way that makes the pheromone value of each route equal ($\tau_1 = \tau_2$). Otherwise; data transmission will not be routed through R_2 until its pheromone level upon the next transmission is zero.

Whenever pheromone level to a destination node is noticed to be zero on the next transmission, any current transmission should be routed through R_2. This is because if the packet is not routed through R_2, the route will be deleted when the next packet from this destination comes in. (Because, whenever a packet is received, pheromone values of all routes leading to that source are updated. This makes pheromone level of R_2 to be zero, and it will be deleted) Hence; packet should be sent through this route before it gets deleted, and the pheromone value of a route should be checked every time there is a packet to be sent to a destination.

CHAPTER FIVE

5. SIMULATION AND RESULTS

5.1. Implementation Details

This section presents the implementation detail and results generated from the simulations. In our implementation, nodes were made to measure their remaining energy. Then, the ratio of the remaining energy to the initial energy was taken as a node weight. Once each node measures its weight, the value can be included in the transmitted packets. In our case, when a request ant is sent by a node, it records this value to a field. The node weight and path weight fields are set to be the same as the calculated weight. The last two lines in the following code perform this operation.

```
// Measure node weight
Ptr<EnergySourceContainer> sources = m_ipv4->GetObject<Node>()->GetObject<EnergySourceContainer>();
Ptr<BasicEnergySource> basicSourcePtr = DynamicCast<BasicEnergySource>(sources->Get(0));
double weight = basicSourcePtr->GetRemainingEnergy() / basicSourcePtr->GetInitialEnergy();
// Set packet fields with the node weight
rreqHeader.SetWeight(weight);
rreqHeader.SetPathWeight(weight);
```

Nodes receiving the request ant extract the weight and use it when creating/updating route to the neighbor. Similarly, the path weight is used when creating/updating route to the request ant source. However; when rebroadcasting the request ant to neighbors, nodes replace the node weight field with their weight. Then, they use two methods to set path

weight to the source of the request ant. These are the “mean” and “product methods”. The following code was used to perform these operations.

```
// Measure node weight
Ptr<EnergySourceContainer> sources = m_ipv4->GetObject<Node>()->GetObject<EnergySourceContainer>();
Ptr<BasicEnergySource> basicSourcePtr = DynamicCast<BasicEnergySource>(sources->Get(0));
double my_weight = basicSourcePtr->GetRemainingEnergy() / basicSourcePtr->GetInitialEnergy();
// Set packet fields during rebroadcast
rreqHeader.SetWeight(my_weight);
double path_weight = rreqHeader.GetPathWeight();
// Set path weight when using the mean method
rreqHeader.SetPathWeight(path_weight + my_weight);
// Set path weight when using the product method
rreqHeader.SetPathWeight(path_weight * my_weight);
```

Unlike AODV, our method does not just drop duplicate request ants automatically. Instead, it checks to see if the weight of the path through which the duplicate packet traversed was better. If the weight collected by the duplicate request ant is higher, then the node updates

its current routing entry by the new value from the request ant. The following code was used to update routes using weights from the duplicate packets.

```
/*
 * Checks if Request Ant was received with the same Source IP
and Request ID.
 * If so, compare the weight from the newly received Request
Ant with the previous one.
 */
if (m_rreqIdCache.IsDuplicate(origin, id))
{
      // When using the product method
      if (rreqHeader.GetPathWeight() >
routeToOrigin.GetWeight())
      {
            // Update route to Origin
            // Update route to the Neighbor
      }

      // When using the mean method
      if ((rreqHeader.GetPathWeight() /
(rreqHeader.GetHopCount() + 1)) > (routeToOrigin.GetWeight() /
routeToOrigin.GetHop()))
      {
            // Update route to Origin
            // Update route to the Neighbor
      }
      // Drop the ant
}
```

Just as sending a request ant, sending a reply ant to a source follows similar steps. However; when nodes receive a reply ant, they update their routing table in two ways.

The code below shows how nodes update their routing table using the “mean” and “product” methods respectively.

```
uint8_t hop = rrepHeader.GetHopCount() + 1;
double weight_toDst = rrepHeader.GetPathWeight();
// Using the mean method
if ((rrepHeader.GetDstSeqno() == toDst.GetSeqNo()) &&
((weight_toDst / hop) > (toDst.GetWeight() / toDst.GetHop())))
{
      // Update route to the source of the Reply Ant
}
// Using the product method
if ((rrepHeader.GetDstSeqno() == toDst.GetSeqNo()) &&
(weight_toDst > toDst.GetWeight()))
{
      // Update route to the source of the Reply Ant
}
```

Just like the way request ants header fields are updated, the headers of reply ants are updated using the same technique. It is also important to mention that the implementation was only focused on solving the first problem we had stated. Hence; the implementation does not provide alternative routes leading to destinations. As the decision to make changes to the routing table was based on the employed “mean” and “product” methods, simulation of the proposed approach was tested in the two scenarios.

5.1.1. Node Deployment Details

The physical layer standard used in the simulation was the IEEE 802.11 standard. This is the base standard upon which the infrastructure-based and Ad hoc modes of Wireless Local Area Network (WLAN) were established. The physical layer model used was the (Yet Another Network Simulator) YansWifiPhy [35], which is the default model for Ad hoc in NS3. The MAC layer model used in the simulation was the ns3::AdhocWifiMac. This allows the nodes to establish a Wi-Fi network topology of the form, non-access-point station (STA) within an Independent Basic Service Set (IBSS). AODV uses the transport layer protocol User Datagram Protocol (UDP), hence; we have also used the same protocol at the transport layer of the simulated nodes. Initially, the application layer was made to generate packets at a Constant Bit Rate (CBR) of 2 kbps. However; the packet size was increased from the default 64 to 256, and 512 Byte to see its effect on the simulation performance.

A total of 50 nodes were randomly placed within a 300 X 1500 m^2 area. To see the effect of higher network load resulting from the increased number of communicating nodes, we have gradually increased the number of sink nodes from 10 to 20, and 25. The movement of these nodes were managed using the Random WayPoint mobility model discussed in section 2.3 of chapter 2. The movement of nodes was made to be in a non-stop motion with a zero pausing time. The pausing time was set to 0 because we wanted to test the performance of our methods in a dynamically changing network topology. The following figure shows the random deployment of the simulated nodes.

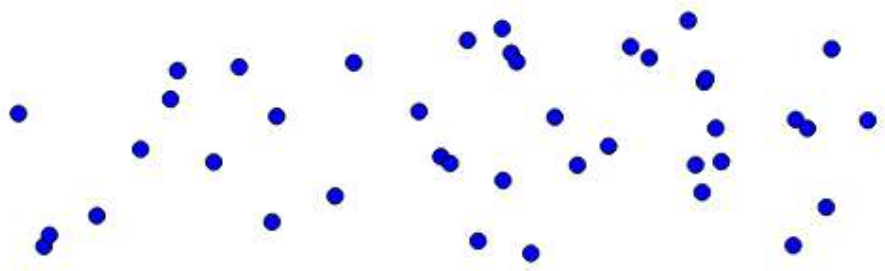


Figure 5.1 Deployment topology of the simulated nodes.

Furthermore; a single energy source, with an initial energy source of 10.1 Joules, was attached on all nodes in the simulation. The physical layer of all nodes draws the default transmission current of 0.0174 Amperes from the energy source.

Simulations were run using the AODV, BIO-AODV product method, and BIO-AODV mean method protocol under the following parameters.

Table 5.1 Summary of simulation parameters.

Parameter	Value
Total Number of Nodes	50
Number of Sink Nodes	10, 20, 25
Application Layer Data Rate	2 Kbps
Application Layer Packet Size	64, 256, 512 Byte
Maximum Node Speed	20, 30 m/s
Pause Time	0 Second
Physical Layer Standard	802.11b
Simulation Area	300 × 1500 m
Mobility Model	Random Waypoint
Initial Node Energy	10.1 J
Transport Layer Protocol	UDP

5.1.2. Simulation Performance Measurement

The performance metrics used to compare the simulation results were average packet receiving throughput, end-to-end delay, and packet delivery ratio. These evaluation metrics were calculated using the collected data at the network layer in a per flow manner. A flow is represented as a communication between pair of source and destination nodes. In particular, a flow can be represented as the pair (Protocol, source (IP, Port), destination (IP, Port)). The average throughput of all flows was calculated by taking the mean of all throughputs. Throughput can be defined as the number of bits received in a given period of time. This period in a flow is the amount of time between the last and first received packets. Mathematically it is represented as:

$$Throughput_{mean} = \frac{1}{n} \sum_{flow=1}^{n} \frac{No.Received\ Bits}{(T_{Last} - T_{First})} \quad 5.1$$

The average end-to-end delay in a flow was measured as the ratio between total delay and the number of packets received. The overall average end-to-end delay is the mean of all average end-to-end delays. Mathematically it is represented as:

$$Delay_{mean} = \frac{1}{n} \sum_{flow=1}^{n} \frac{Delay\ Sum}{No.\ Received\ Packets} \quad 5.2$$

Packet delivery ratio of a flow can be defined as the amount of received packets from the total number of sent packets. Hence; the average packet delivery ratio was computed by taking the mean of all packet delivery ratios. Mathematically it is represented as:

$$Delivery\ Ratio_{mean} = \frac{1}{n} \sum_{flow=1}^{n} \frac{No.\ Received\ Packets}{No.\ Transmitted\ Packets} \quad 5.3$$

Where n is the total number of flows in the network. In general, a total of 36 simulations were run to compare only the product and mean methods. Appendix E presents the output of these simulation results.

5.2. Comparison of the Mean and Product Methods

5.2.1. Mean End-to-End Delay

Discussion. When comparing the average end-to-end delay that occurred on each flow, it appears that the mean method creates a higher delay than the product method. For 10 source/sink nodes from 50, each moving at the maximum speed of 20 m/s, routes chosen using the product method seems to have a 69 milliseconds average delay. But, routes chosen using the mean method seem to have an average delay of 88 milliseconds. When the maximum speed of all the nodes was increased to 30 m/s, the average end-to-end delay also increased. But, still, the product method provides a lower average end-to-end delay per flow.

The following two graphs show the average end-to-end delay of flows when 10 of the 50 nodes, moving at a maximum speed of 20 m/s, are sending and receiving.

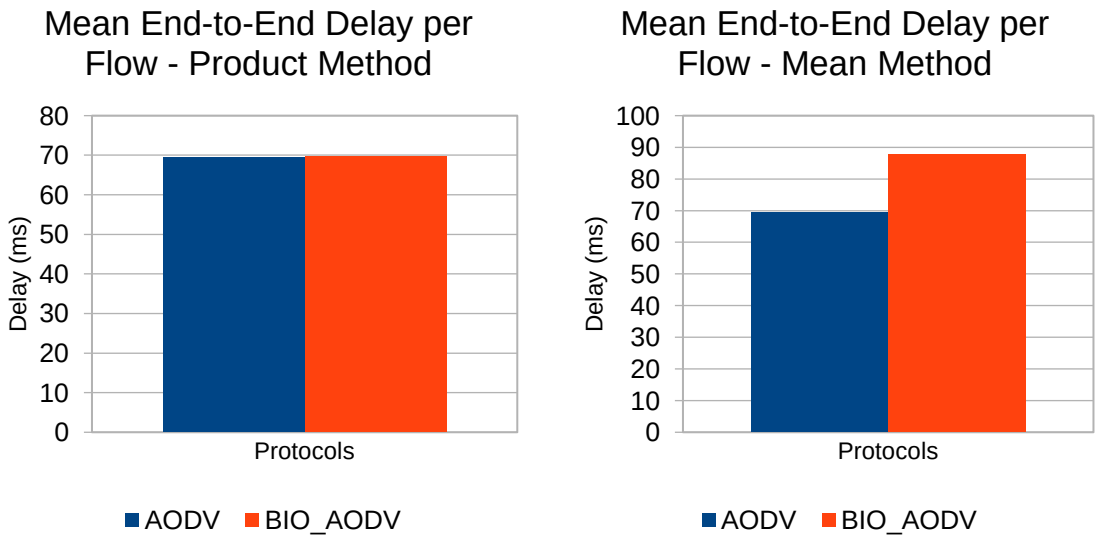


Figure 5.2 Average end-to-end delay of the product and mean methods. (Packet size: 64 Byte)

The following two graphs show the average end-to-end delay of flows when 10 of the 50 nodes, moving at a maximum speed of 30 m/s, are sending and receiving.

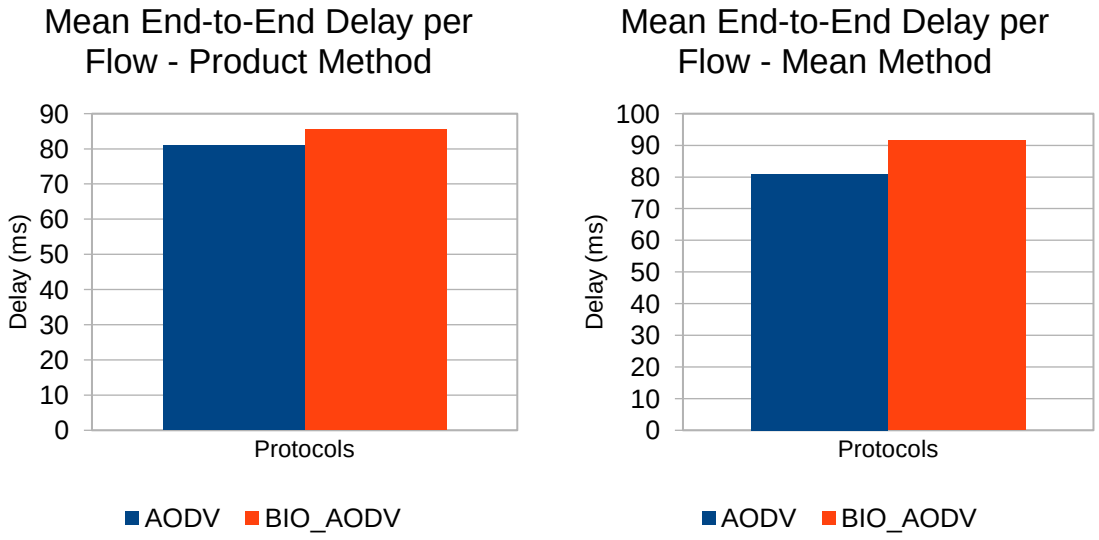


Figure 5.3 Average end-to-end delay of the product and mean methods.

The above four graphs show that route decisions based on the product method are more efficient than the traditional mean method.

Discussion. When the number of communicating pairs was increased from 10 to 20, where nodes are moving at a maximum speed of 20 m/s, the average end-to-end delay of the mean and product methods was 80 milliseconds. And, there seems to be no significant difference between our two methods of route selection in terms of average end-to-end delay. The following two graphs show the average end-to-end delay of all flows when 20 of the 50 nodes, moving at a maximum speed of 20 m/s, are sending and receiving.

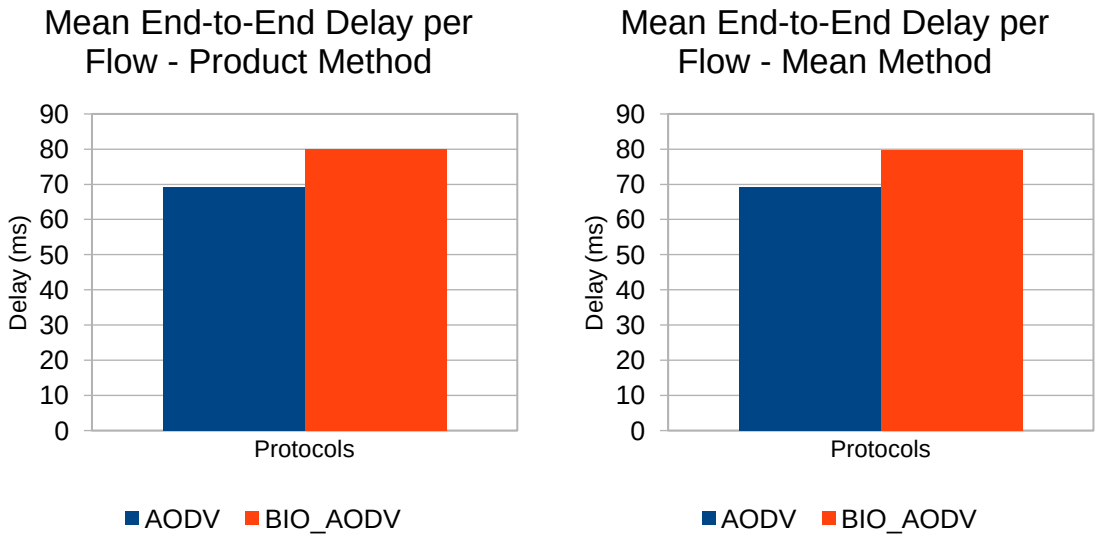


Figure 5.4 Average end-to-end delay of the product and mean methods.

However; when the maximum speed of all the nodes was increased from 20 to 30 m/s, the average end-to-end delay of our methods also increased. But, the increase in the mean delay of the product method was lower than the mean method. The following two graphs show

the average end-to-end delay of all flows when 20 of the 50 nodes, moving at a maximum speed of 30 m/s, are sending and receiving.

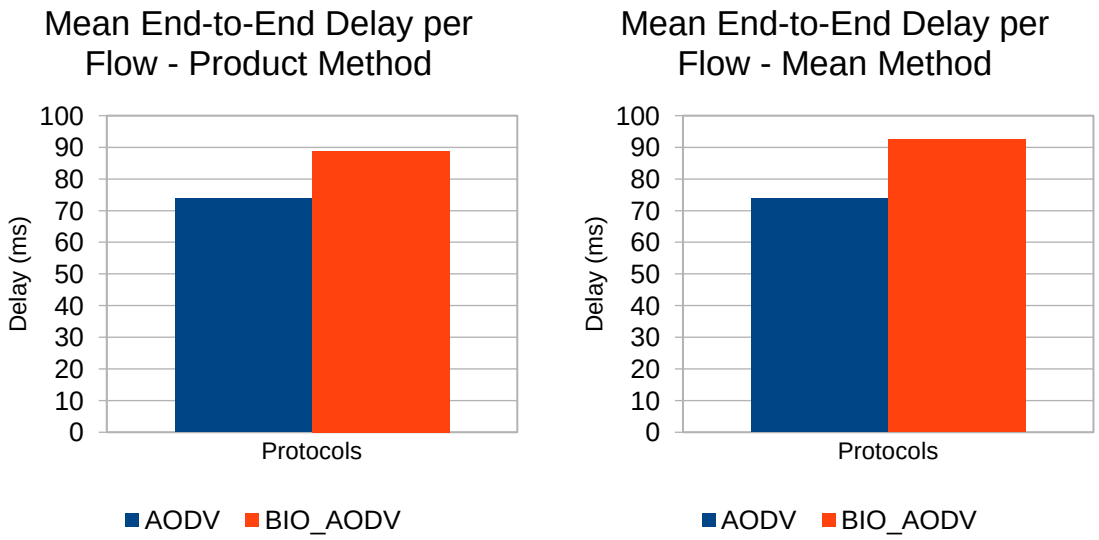


Figure 5.5 Average end-to-end delay of the product and mean methods.

Summary. In general, end-to-end delay is affected by factors such as processing and transmission delays. Note that AODV simply discards duplicate request packets. Hence; the observed increase in the average end-to-end delay in our approach seems to be because of the added logic that updates the routing table using the weight from the duplicate request ants. Furthermore; it was seen that the product method performs better than the mean method in terms of mean delay.

5.2.2. Mean Packet Receiving Throughput and Delivery Ratio

The average receiving throughput of our method seems higher than AODV's. The average number of bits received at each destination per a given time was higher in our method. However; when comparing our two methods, the product method performs better in the average number of packets delivered. The following four graphs show that the product method in our proposed solution out-performs both the mean method and AODV. The

following results are from the simulations of 10 source and sink node moving at speed of 20 m/s and.

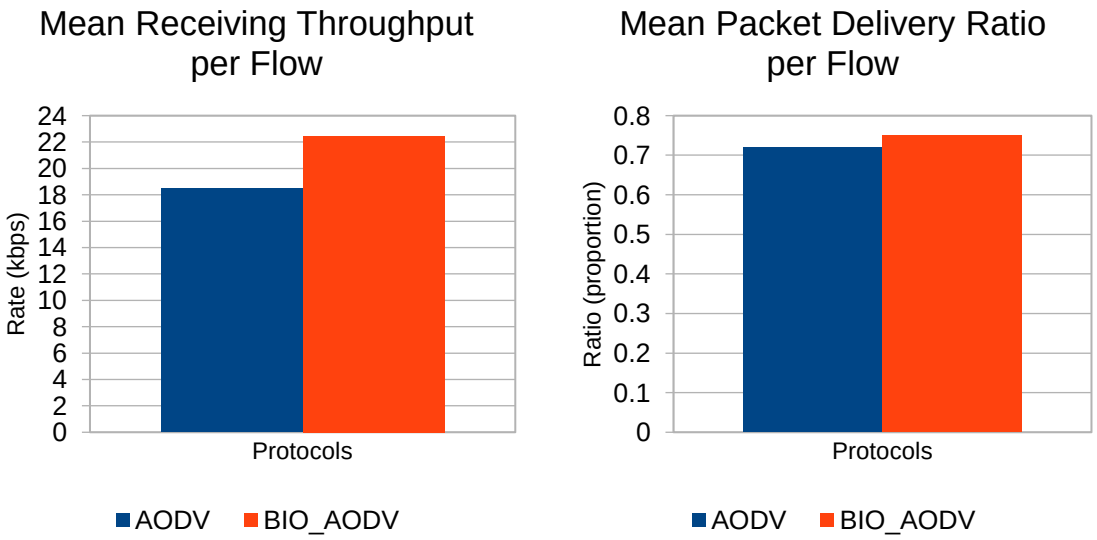


Figure 5.6 Average throughput and delivery ratio when using the product method.

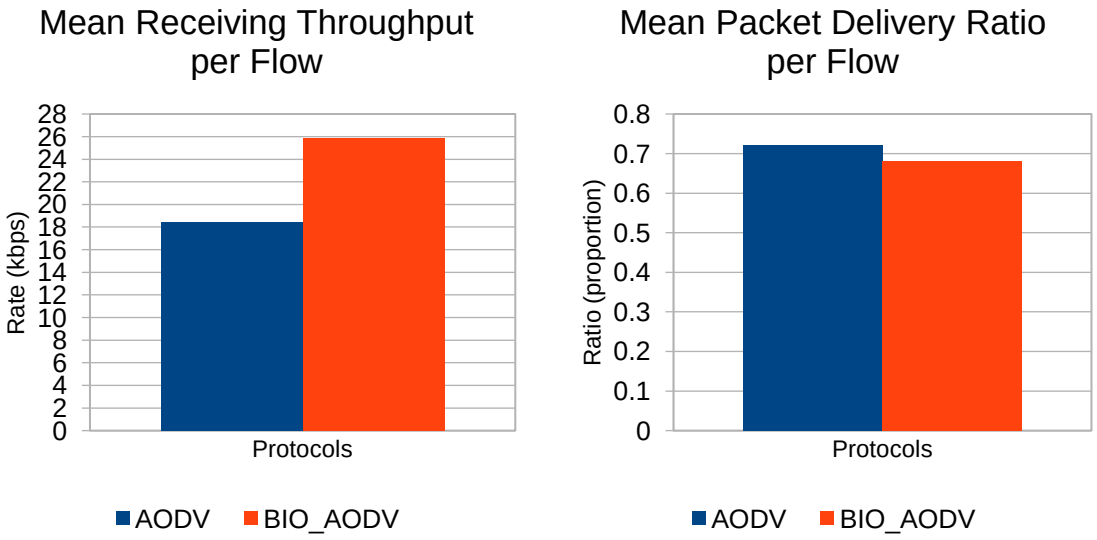


Figure 5.7 Average throughput and delivery ratio when using the mean method.

As shown in the above graphs, the increase in the data rate of the mean method did not increase the delivery ratio. Hence; the mean method seems to create a higher drop rate. This is because a large number of packets are generated but are not being delivered properly. This is clearly shown in the following four graphs where the number of communicating nodes was increased to 20.

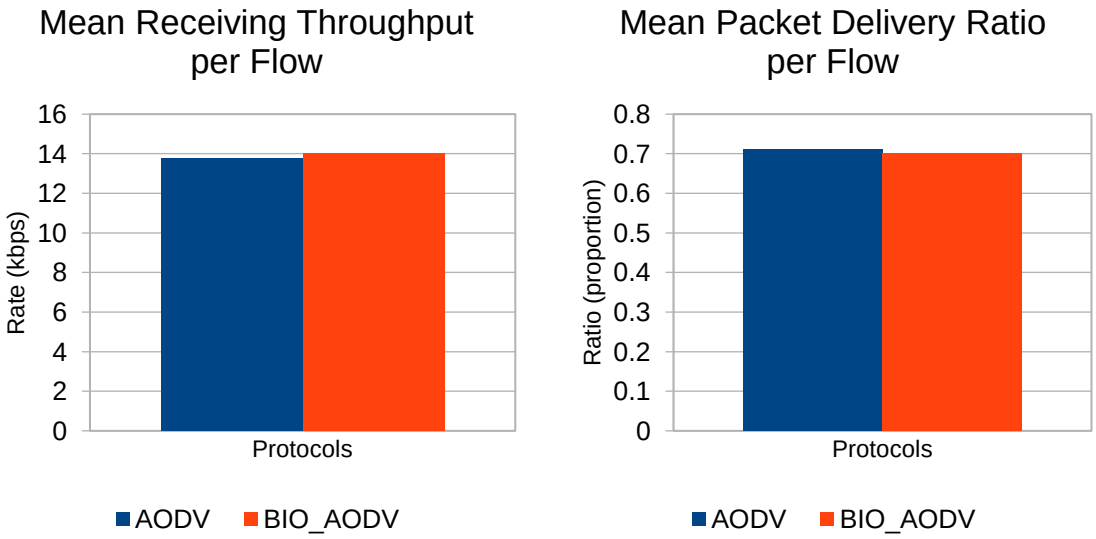


Figure 5.8 Average throughput and delivery ratio when using the product method.

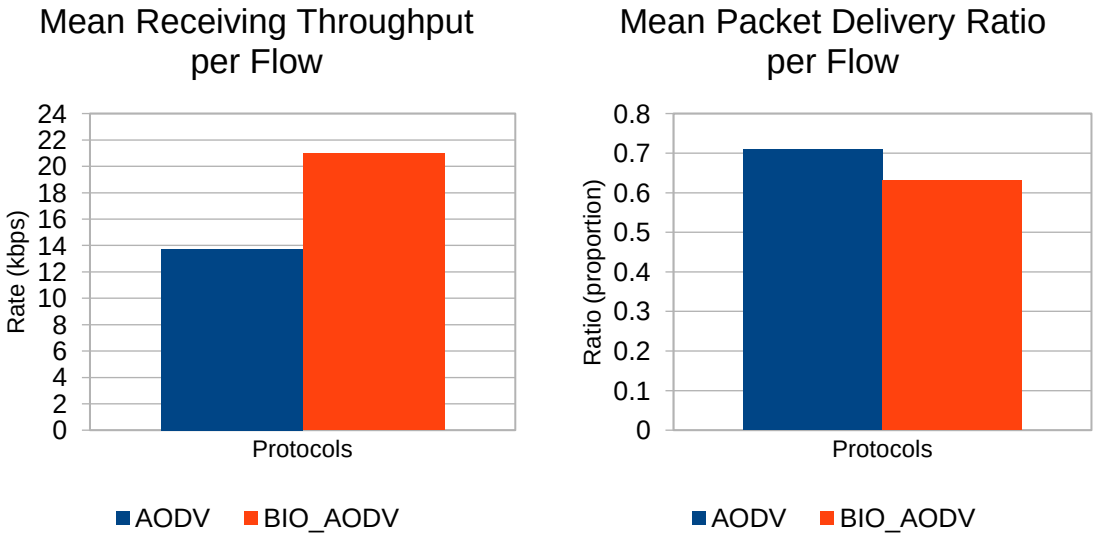


Figure 5.9 Average throughput and delivery ratio when using the mean method.

Finally, when all the nodes are communicating while moving at the speed of 30 m/s, the performance of our method (the product method) is higher than AODV's and the mean method in terms of both throughput and delivery ratio. The following four graphs show this finding.

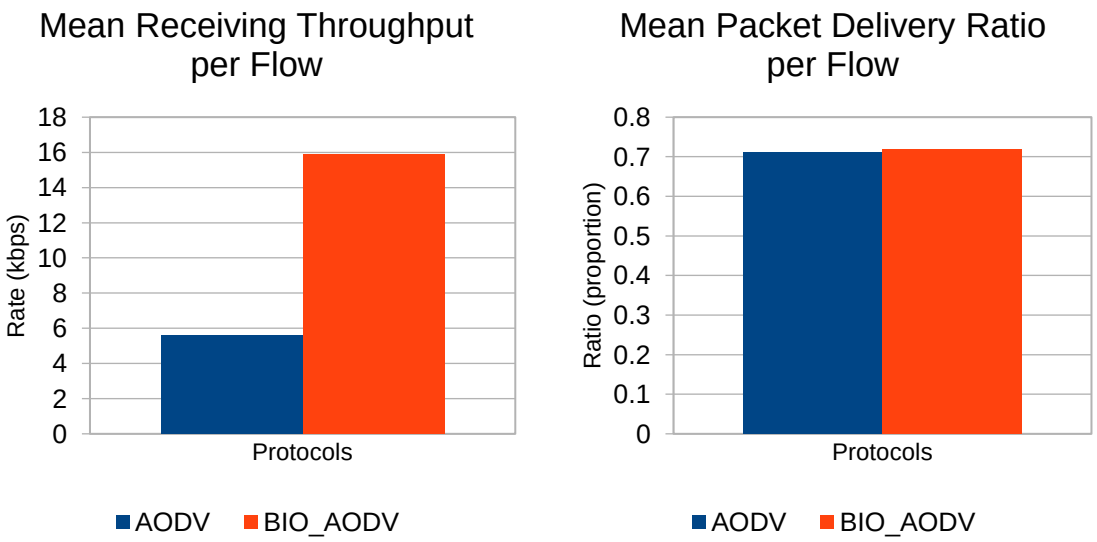


Figure 5.10 Average throughput and delivery ratio when using the product method.

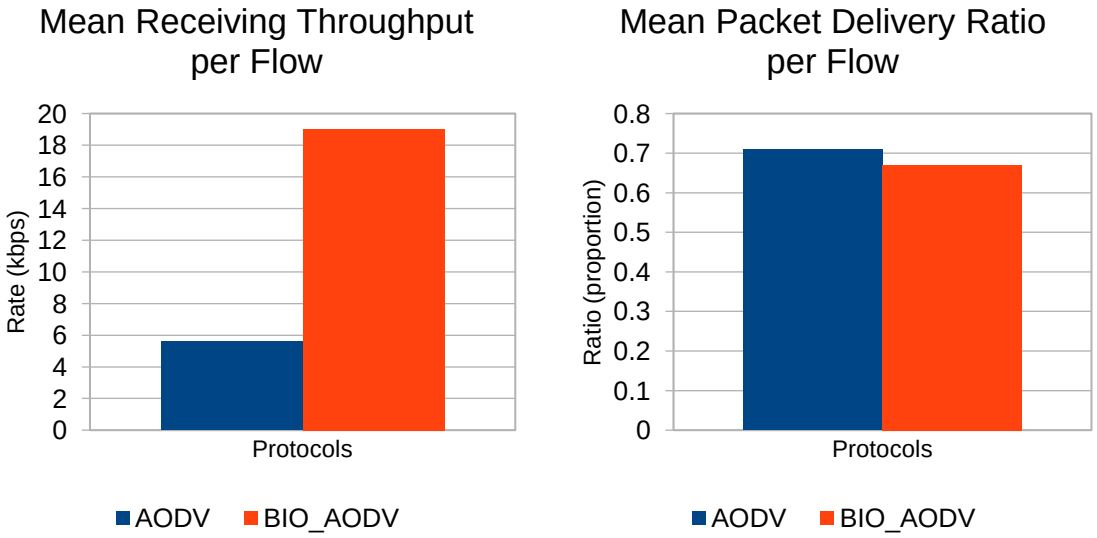


Figure 5.11 Average throughput and delivery ratio when using the mean method.

In general, packet delivery ratio and throughput are related to network parameters. Hence; an increase in one should increase the other one. This seems to be the reason for the observed improvement in our product method of route selection when compared with the current implementation of AODV. However; unlike the product method, the mean method generating a higher number of bits without improving the actual delivery ratio.

When simulating our method of route selection, we have also measured the average percentage of lost packets. The mathematical calculation for this metric and performance of our result can be seen from Appendix F.

5.3. Validation of the Results

To prove validity of our results, we performed a *t-test* to compare the mean and product methods. We had proved, using mathematical and simulations, that the product method is better in terms of the average throughput, end-to-end delay, and packet delivery ratio. But, was the result significantly higher? To answer this question we set up a null (H_0) and alternative hypothesis (H_1). In our hypothesis test, we denoted our methods- the mean and product- methods as m_1 and m_2 respectively.

5.3.1. Hypothesis Test 1

The null hypothesis (H_0) can be stated as "the average end-to-end delay of all flows when using the mean method is lower or equal to the average end-to-end delay when using the product method." Then, the alternative hypothesis (H_1) becomes "the average end-to-end delay of all flows when using the mean method is higher than the average end-to-end delay when using the product method."

When stated mathematically:

$$H_0: \mu_1 - \mu_2 \leq 0 \quad (5.4)$$

$$H_1: \mu_1 - \mu_2 > 0 \quad (5.5)$$

The amount of type 1 error allowed (significance level) was 0.05. As shown in the following table, the p-value calculated was 0.0. And, with 95% confidence level the mean method has an average of additional delay between 4.93-15.44 milliseconds. Hence; we reject the null hypothesis in fever of the alternative hypothesis.

Table 5.2 Hypothesis test results for the average end-to-end delay.

Compared Methods	p-value	Mean Difference	Standard Deviation	95% Confidence Interval (lower, upper)	Compared Parameter
Mean and Product Methods	0.0	10.25 ms	2.68	(4.93, 15.44) ms	Difference in the Average End-to-End Delay

In general, when the probability of getting our sample statistic or more extreme values under the null hypothesis is zero, we reject the null hypothesis. In our case, the probability that

there will be no difference between the two methods in terms of average end-to-end delay is 0.0. This indicates that, the average delay created by the product method is lower than the mean method. Furthermore; the performance improvement seen in the product method was statistically significant.

5.3.2. Hypothesis Test 2

The null hypothesis (H_0) can be stated as "the average packet delivery ratio of all flows when using the mean method is higher or equal to the average packet delivery ratio when using the product method." Then, the alternative hypothesis (H_1) becomes "the average packet delivery ratio of all flows when using the mean method is lower than the average packet delivery ratio when using the product method."

When stated mathematically:

$$H_0: \mu_1 - \mu_2 \geq 0 \quad 5.6$$

$$H_1: \mu_1 - \mu_2 < 0 \quad 5.7$$

Again, the amount of type 1 error allowed was 0.05. As shown in the following table, the p-value calculated was 0.0204. And, with 95% confidence level the mean method has an average of lower delivery ratio of 0.1667-3.11%. Hence; we rejected the null hypothesis in fever of the alternative hypothesis.

Table 5.3 Hypothesis test results for the average packet delivery ratio.

Compared Methods	p-value	Mean Difference	Standard Deviation	95% Confidence Interval (lower, upper)	Compared Parameter
Mean and Product Methods	0.0204	-1.56%	0.01	(-3.11, -0.1667) %	Difference in the Average Packet Delivery Ratio

In general, when the probability of getting our sample statistic or more extreme values under the null hypothesis is lower than the preset threshold level (0.05), we reject the null hypothesis. In our case, the probability that there will be no difference between the two methods in terms of average packet delivery ratio is 0.0204. This indicated that, the average packet delivery ratio of the product method is higher than the mean method.

CHAPTER SIX

6. CONCLUSION AND FUTURE WORK

In this research two problems were raised, one, the inability of AODV to measure and utilize node level quality parameters, second, failure to provide an alternative route. To answer these research questions we have adopted the popular Ant Colony Optimization algorithm. Furthermore; we have proposed a new method to get accurate route quality using node-level measurements. Simulations were tested on two of our best route selection mechanisms. These are the product and mean methods of selection.

The comparison of AODV and our approach shows that our methods create a higher average end-to-end delay between communicating pairs. This becomes even higher when the number of communicating nodes and their maximum speed increases. However; when comparing our two methods of choosing the best route, the product method was found to be the best mechanism to accurately model route quality.

On the other hand; the comparison of AODV and our approaches in terms of average throughput and packet delivery ratio shows that the product method performs better than AODV. Even though both of our methods produce higher throughput, the packet delivery ratio of the mean method is lower than AODV. In a stressful situation where all nodes are communicating and moving at the speed of 30 m/s, the product method created an average of a 1% increase in the packet delivery ratio than AODV.

One of the fundamental findings of this research was disproving the accuracy of a well-known method of measuring a route quality. Traditionally, the summation of individual values was used to measure route quality. However; as shown in this research, using the product of individual values can better model the underlying truth about the network.

REFERENCES

[1] M. Corson, J. Maker and J. Cernicione, "Internet-based Mobile Ad Hoc Networking," *IEEE Internet Computing*, pp. 63-70, 1999.

[2] M. Conti, "Body, Personal, and Local Ad Hoc Wireless Networks," in *The Handbook of Ad hoc Wireless Networks*, CRC Press LLC., 2003, pp. 13-34.

[3] J. P. Macker and M. S. Corson, "Mobile Ad hoc Networks (MANETs): Routing Technology for Dynamic Wireless Networking," in *Mobile Ad Hoc Networking*, Institute of Electrical and Electronics Engineers, 2004, pp. 255-273.

[4] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations (RFC: 2501)," The Internet Society, 1999.

[5] I. Leontiadis, E. Ferranti, C. Mascolo, L. McNamara, B. Pasztor, N. Trigoni and S. Waharte, "APPLICATION SCENARIOS," in *Mobile Ad Hoc Networking: Cutting Edge Directions*, New Jersey, JohnWiley & Sons, Inc., 2013, pp. 77-105.

[6] G. E. Rolader, J. Rogers and J. Batteh, "Self-healing minefield," *Journal of Battlespace Digitization and Network-Centric Systems,* pp. 13-24, 2004.

[7] Y. Feng, B. Zhang, S. Chai, L. Cui and Q. Li, "An Optimized AODV Protocol Based on Clustering for WSNs," in *6th International Conference on Computer Science and Network Technology(ICCSNT)*, 2017.

[8] C. E. Perkins, E. M. Royer, S. R. Das and M. K. Marina, "Performance Comparison of Two On-demand Routing Protocols for Ad-hoc Networks," *IEEE Personal Communications,* pp. 16-28, 2001.

[9] E. Weingartner, H. v. Lehn and K. Wehrle, "A performance Comparison of Recent Network Simulators," in *IEEE International Conference on Comunications*, Dresden, 2009.

[10] S. Glisic, "Ad Hoc Networks," in *Advanced Wireless Networks: Technology and Business Models*, John Wiley & Sons, Ltd., 2016, pp. 126-193.

[11] E. M. Royer and C. K. Toh, "A Review of Current Routing Protocols for Ad-hoc Mobile Wireless Networks," *IEEE Personal Communications*, pp. 46-55, 1999.

[12] T. Melodia, H. Kulhandjian, L.-C. Kuo and E. Demirors, "Advances in UnderWater Acoustic Networking," in *Mobile Ad hoc Networking: the Cutting Edge Directions*, New Jersey, John Wiley & Sons, Inc., 2013, pp. 804-852.

[13] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks," *Mobile Computing*, p. 153–181, 1996.

[14] C. E. Perkins and E. M. Royer, "Ad-hoc On-demand Distance Vector Routing," in *Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[15] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing (RFC: 3561)," The Internet Society, 2003.

[16] "ns3::aodv::RreqHeader Class Reference," NS-3, 21 October 2020. [Online]. Available: https://www.nsnam.org/doxygen/classns3_1_1aodv_1_1_rreq_header.html#details. [Accessed 21 October 2020].

[17] "ns3::aodv::RrepHeader Class Reference," NS-3, 21 October 2020. [Online]. Available: https://www.nsnam.org/doxygen/classns3_1_1aodv_1_1_rrep_header.html#details. [Accessed 21 October 2020].

[18] T. Nieberg and J. Hurink, "Wireless Communication Graphs," in *Proceedings of the 2004 Intelligent Sensors, Sensor Networks and Information Processing Conference*, Melbourne, 2004.

[19] T. Camp, J. Boleng and V. Davies, "A Survey of Mobility Models for Ad hoc Network Research," *Wireless Communications and Mobile Computing*, vol. 2, p. 483–502, 2002.

[20] C. Chiang, "Wireless Network Multicasting," University of California, Los Angeles, 1998.

[21] J. Bishop, "Stochastic Searching Networks," in *Proceedings of 1st IEE International Conference Artificial Neural Networks*, London, 1989.

[22] M. M. al-Rifaie and J. M. Bishop, "Stochastic Diffusion Search Review," *PALADYN Journal of Behavioral Robotics*, vol. 4, no. 3, pp. 155-173, 2013.

[23] A. Colorni, M. Dorigo and V. Maniezzo, "Distributed Optimization by Ant Colonies.," in *Proceedings of the First European Conference on Artificial Life*, 1991.

[24] J. Kennedy and R. Eberhart, "Particle Swarm Optimisation," in *Proceedings of the IEEE International Conference on Neural Networks*, Piscataway, 1995.

[25] C. Blum and A. Roli, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268-308, 2003.

[26] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," in *IEEE International Conference on Evolutionary Computational Proceedings*, Anchorage, 1998.

[27] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Reading, Massachusetts: Addison-Wesley Publishing Company Inc., 1989.

[28] K. D. Meyer, "Foundations of Stochastic Diffusion Search," Ph.D. Thesis Submitted to the University of Reading, 2004.

[29] E. Dijkstra, "A Note on Two Problems in Connexion With Graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269-271, 1959.

[30] P. Vijayalakshmi, S. A. J. Francis and J. A. Dinakaran, "A Robust Energy Efficient Ant Colony Optimization Routing Algorithm for Multi-hop Ad hoc Networks in MANETs," *Springer Wireless Networks*, vol. 22, no. 6, pp. 2081-2100, 2016.

[31] H. Zhang, X. Wang, P. Memarmoshrefi and D. Hogrefe, "A Survey of Ant Colony Optimization Based Routing Protocols for Mobile Ad Hoc Networks," *IEEE Access*, vol. 5, pp. 24139-24161, 2017.

[32] C. Shubhajeet and D. Swagatam, "Ant Colony Optimization Based Enhanced Dynamic Source Routing Algorithm for Mobile Ad-hoc Network," *Information Sciences*, pp. 67-90, 2015.

[33] J. Kang, Y. Zhang and B. Nath, "Accurate and Energy-efficient Congestion Level Measurement in Ad Hoc Networks," in *IEEE Wireless Communication and Networking Conference*, New Orleans, 2005.

[34] A. Al-Ani and J. Seitz, "QoS-aware Routing in Multi-rate Ad hoc Networks Based on Ant Colony Optimization," *Network Protocols and Algorithms,* vol. 7, no. 4, pp. 1-25, 2015.

[35] M. a. H. T. R. Lacage, "Yet Another Network Simulator," in *Proceeding from the 2006 Workshop on Ns-2: The IP Network Simulator*, Pisa, Association for Computing Machinery (ACM), 2006, pp. 12-22.

[36] X. Hong, M. Gerla, G. Pei and C. Chiang, "A Group Mobility Model for Ad-Hoc Wireless Networks," in *Proceedings of the ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM)*, 1999.

[37] A. Yadav, Y. N. Singh and R. R. Singh, "Improving Routing Performance in AODV with Link Prediction in Mobile Adhoc Networks," *Wireless Personal Communications*, 2015.

[38] B. R. Hanji and R. Shettar, "Improved AODV with Restricted Route Discovery Area," in *International Conference on Computer Communication and Informatics (ICCCI - 2015)*, Coimbatore,, 2015.

[39] AndreaGorrieri and G. Ferrari, "Irresponsible AODV routing," *Vehicular Communications,* vol. 2, pp. 47-57, 2015.

[40] M. Yoshimachi and Y. Manabe, "A New AODV Route Discovery Protocol to Achieve Fair Routing for Mobile Ad Hoc Networks," in *6th International Conference on Information Communication and Management*, 2016.

[41] J. Ichikawa, S. Sakata and N. Komuro, "Fair Energy Consumption-Based Routing Control Considering Data Traficc for Mobile Ad hoc Networks," 2013.

[42] A. Shamsoshoara and Y. Darmani, "Enhanced Multi-Route Ad Hoc On-Demand Distance Vector Routing," in *23rd Iranian Conference on Electrical Engineering*, 2016.

APPENDIX

Appendix A. NS-3 code used to set/get neighbor and path weight in the packet headers.

```
/**
   * \brief Set the weight
   * \param weight the weight
   */
  void SetWeight (double weight)
  {
    m_weight = weight;
  }
  /**
   * \brief Get the weight
   * \return the weight
   */
  double GetWeight () const
  {
    return m_weight;
  }
  /**
   * \brief Set the path weight
   * \param weight the path weight
   */
  void SetPathWeight (double path_weight)
  {
    m_path_weight = path_weight;
  }
  /**
   * \brief Get the path weight
   * \return the path weight
   */
  double GetPathWeight () const
  {
    return m_path_weight;
  }
```

Appendix B. NS-3 code used to set/get path weight in the routing table.

```
/**
   * Set the path weight
   * \param weight The path weight
   */
  void SetWeight (double weight)
  {
    m_weight = weight;
  }
  /**
   * Get the weight
   * \returns the weight
   */
  double GetWeight () const
  {
    return m_weight;
  }
```

Appendix C. NS-3 code used to initialize a routing entry in the routing table.

```
/**
   * constructor
   *
   * \param dev the device
   * \param dst the destination IP address
   * \param vSeqNo verify sequence number flag
   * \param seqNo the sequence number
   * \param iface the interface
   * \param hops the number of hops
   * \param nextHop the IP address of the next hop
   * \param lifetime the lifetime of the entry
   * \param weight the path weight to the destination
   */
  RoutingTableEntry (Ptr<NetDevice> dev = 0,Ipv4Address dst =
Ipv4Address (), bool vSeqNo = false, uint32_t seqNo = 0,
Ipv4InterfaceAddress iface = Ipv4InterfaceAddress (), uint16_t  hops
= 0, Ipv4Address nextHop = Ipv4Address (), Time lifetime =
Simulator::Now (), double weight = 0.0);
```

Appendix D. Table that presents the average end-to-end delay and loss ratio of all the 12 simulations.

Mean End-to-End Delay per Flow (ms)		Mean Loss Ratio per Flow (%)		Number of Sink Nodes	Maximum Node Speed (m/s)	Best Path Selection Method
AODV	BIO_AODV	AODV	BIO_AODV			
69.36	87.71	5.02	3.65	10	20	Mean
69.18	79.75	4.87	4.38	20		
67.23	75.33	4.93	3.75	25		
81	91.57	5.07	3.39	10	30	
73.83	92.54	5.47	4.84	20		
70.64	111.46	4.82	4.71	25		
69.36	69.67	5.02	4.01	10	20	Product
69.18	79.86	4.87	4.88	20		
67.23	71.75	4.93	4.36	25		
81	85.48	5.07	4.4	10	30	
73.83	88.77	5.47	5.08	20		
70.64	79.14	4.82	5.39	25		

The following table presents the average packet delivery ratio and throughput of all the 12 simulations.

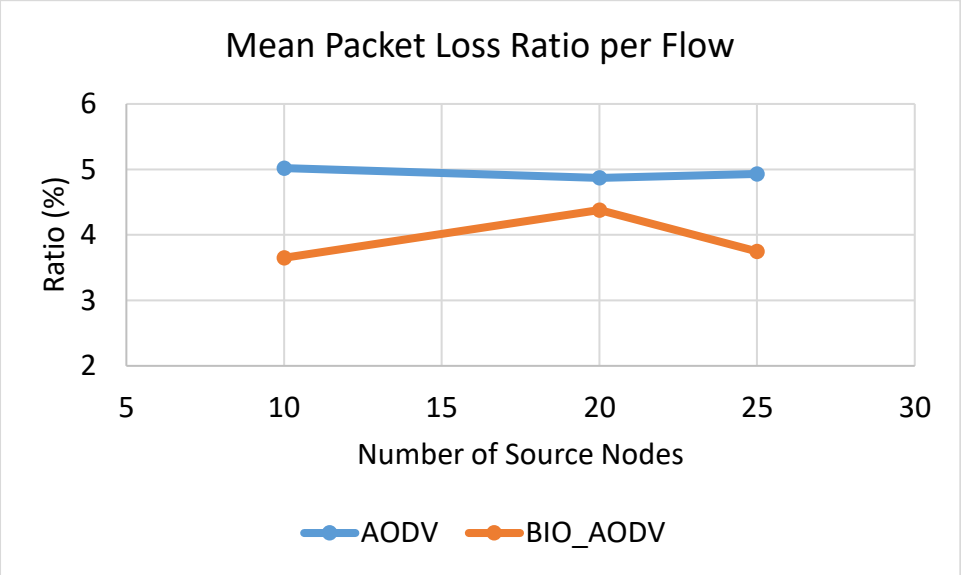
Mean Packet Delivery Ratio per Flow (ms)		Mean Throughput per Flow (%)		Number of Sink Nodes	Maximum Node Speed (m/s)	Best Path Selection Method
AODV	BIO_AODV	AODV	BIO_AODV			
0.72	0.68	18.47	25.84	10	20	Mean
0.71	0.63	13.75	20.98	20		
0.73	0.63	12.28	16.58	25		
0.70	0.57	19.27	34.43	10	30	
0.68	0.64	12.02	11.7	20		
0.71	0.67	5.61	19.03	25		
0.72	0.75	18.47	22.46	10	20	Product
0.71	0.70	13.75	14	20		
0.73	0.70	12.28	17.04	25		
0.70	0.61	19.27	26.39	10	30	
0.68	0.67	12.02	17.68	20		
0.71	0.72	5.61	15.88	25		

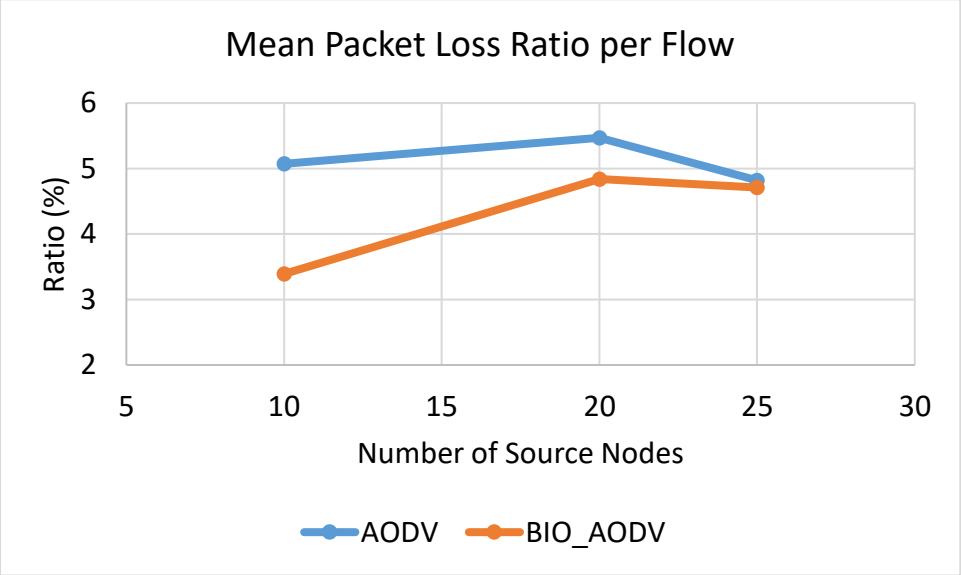
Appendix E. Measuring average packet loss ratio.

The average percentage of lost packets can be calculated as the ratio of lost packets to the sum of lost and received packets. This is mathematically illustrated as:

$$Loss\ Ratio_{mean} = \frac{1}{n} \sum_{flow=1}^{n} \frac{No.Lost\ Packets}{No.Received + Lost\ Packets}$$

Where n represents the total number of flows. The following two graphs compare performance of the mean method with AODV for nodes moving at the speed of 20 and 30 m/s respectively.





These graphs show that the mean method of selecting the best route is good at reducing the percentage of lost packets. As shown in the following two graphs, the product method

also performs better than AODV under less-stressful situations. These are results of the simulation where nodes were moving at the speed of 20 and 30 m/s respectively.