Improving Route Stability of the AODV Routing Protocol Using Fuzzy Approach

Nihad Ibrahim Abbas

Submitted to the
Institute of Graduate study and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Computer Engineering

Eastern Mediterranean University
October 2016
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Mustafa Tümer
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Doctor of Philosophy in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Doctor of Philosophy in Computer Engineering.

Assoc. Prof. Dr. Mustafa İlkan
Co-Supervisor

Asst. Prof. Dr. Emre Özen
Supervisor

Examining Committee

1. Prof. Dr. Muhammet Ali Akçayol _____
2. Prof. Dr. Mehmet Ufuk Çağlayan _____
3. Assoc. Prof. Dr. Gürcü Öz _____
4. Asst. Prof. Dr. Adnan Acan _____
5. Asst. Prof. Dr. Emre Özen _____

ABSTRACT

There are many studies that focus on improving source–destination route stability and lifetime by modifying the existing wireless mobile Ad Hoc networks (MANETs) routing protocols. In this study, a fuzzy based approach is proposed to enhance the Ad Hoc on-demand distance vector (AODV) routing protocol's performance by selecting the most trusted nodes to construct a route between the source and the final destination nodes. In this scheme, the nodes' parameters, like residual energy, node mobility, and hop count are fed through a fuzzy logic inference system to compute the value of the node trust level, which can be used as a metric to construct an optimal path from source to destination.

The simulation results show that the proposed Fuzzy AODV approach performs better than the classical AODV routing protocol. The average routing load and average end to end delay metrics were improved in all cases of different network parameter changes as node speed, number of nodes, and pause times for both of classical AODV and Fuzzy AODV routing protocols. Whereas the Packet Delivery Ratio (PDR) and average network throughput were slightly varied in different network parameter changes. As shown in simulation results, the effects of fuzzy logic inference modifications that have improved the route selection procedure of classical AODV. This route selection scheme reduces the rebroadcasting of overhead route control (RREQ) packets at each intermediate node which receives more than one copy of the same RREQ packets. This will reduce the number of overhead route control packets flooding throughout the network in route discovery stages of classical

AODV routing protocol. As a result, it decreases the network congestion and helps the network to retain their limited network resources.

Keywords: AODV, routing protocol, MANET, fuzzy logic, trusted nodes.

ÖZ

Plansız mobil ağlarda mevcut yönlendirme protokollerini modifiye ederek kaynak-hedef rotasındaki istikrarı sağlamak ve rotanın ömrünü uzatmak amacını taşıyan çok sayıda çalışma mevcuttur. Bu çalışma, plansız mobil ağlar için geliştirilen AODV (ad hoc on-demand distance vector) yönlendirme protokolünün performansını bulanık tabanlı bir yaklaşımla geliştirip iyileştirmeyi hedeflemektedir. Bu amaçla, kaynak ve hedef arasındaki rotayı belirlemek için en güvenilir bağlantı noktaları seçilmektedir. Bağlantı noktalarının güvenilirliğini, bağlantı noktasının mevcut enerjisi, hızı ve bağlantı noktasına kadar oluşturulan rotada kullanılan bağlantı noktası sayısı belirlemektedir. Bahsedilen parametreler bulanık mantık çıkarım sistemi tarafından değerlendirilip bağlantı noktasının güvenilirliği hesaplandıktan sonra kaynak ve hedef arasındaki en uygun rotanın belirlenmesinde kullanılmaktadır.

Simülasyon sonuçları, önerilen Bulanık AODV yaklaşımının performansının klasik AODV yönlendirme protokolünden daha iyi olduğunu göstermektedir. Uçlar arası veri transfer zamanı ve ortalama yönlendirme yükü çalışılan tüm senaryolarda klasik AODV'ye göre iyileşme göstermektedir. Buna karşılık, veri paketi teslim oranı ve kurulan plansız ağın ortalama iş çıkarma oranı farklı senaryolarda değişiklik göstermektedir. Simülasyon sonuçlarından da anlaşılabileceği üzere, bulanık mantıkla modifiye edilen AODV yönlendirme protokolü kaynak-hedef arasındaki rota seçimini iyileştirmiştir. İyileştirilen rota seçimi yöntemi, bağlantı noktaları arasında kullanılan RREQ paketlerinin tekrarlanma sayısında düşüşe neden olarak ağ içerisinde tekrar tekrar yayınlanan rota kontrol mesajlarının, ağdaki iletişimi engelleme oranını düşürmüştür. Sonuç olarak ortak kullanılan sınırlı iletişim

ortamında, ağın tıkanma oranında düşüşe neden olunmuş ve sınırlı ağ kaynaklarının korunmasında iyileştirme yapılmıştır.

Anahtar kelimeler: AODV, yönlendirme protokolü, bulanık mantık, MANET, güvenilir bağlantı noktaları.

ACKNOWLEDGEMENT

First, I would like to express my grateful and thankful to Dr. Emre Özen for his efforts and supervision to complete this work. Also, my deep grateful to Dr. Mustafa İlkan for his advice and revision of my draft manuscripts. I would also like to thank my wife and son for their patience and encouraging me to continue to completion the work.

Deep respect and love to my parents and my family for their understanding and helping. Their love motivates me to complete this stage of my study of the PhD degree level

To My Family

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AP	Access Point
AODV	Ad Hoc On Demand Distance Vector
BS	Base Station
CBR	Constant Bit Rate
CI	Confidence Interval
CL	Confidence Level
CPU	Central Processing Unit
DSDV	Destination Sequenced Distance Vector
DSR	Dynamic Source Routing
FIS	Fuzzy Inference System
FTP	File Transport protocol
ID	Broadcasting Identification
IP	Internet Protocol
LAN	Local Area Network
MAC	Medium Access Control
MANET	Mobile Ad Hoc Network
NS-2	Network Simulator 2
OLSR	Optimized Link State Routing
OPNET	Optimized Network Engineering Tool
PDA	Personal Digital Assistant
QoS	Quality of Service
RERR	Route Error
RREP	Route Reply

RREQ	Route Request
RSS	Received Signal Strength
RWP	Random Waypoint
SP	Shortest Path
TCP	Transmission Control Protocol

Chapter 1

INTRODUCTION

For the last decades, digital communication networks have demonstrated the communications world over the classical analogue communication networks. The availability of cheap data processors and advances in digital electronic technologies led to surge in using data communication networks. A network can be defined as a group of digital devices tend to share information between them. It relays the data information by using physical mediums to achieve specific tasks. Generally, devices (like computer, laptops, mobile phones, printers, etc.), are used as a network hosts (nodes) and the copper wires, optical fiber, and free space channels are used as data communication mediums. At the beginning, computer networks were designed to share expensive equipment (i.e. printer, scanner and else) as well as to exchange files between the network hosts. After then, these tasks became more popular and versatile by sharing different applications and commercial business [1].

A computer network can be characterized into two groups: wired networks and wireless networks. Wired computer network hosts are connected each other with physical medium. Whereas, the wireless networks have no physical connectivity between network nodes. In the last years, wireless computer networks became more popular than wired networks due to the many attractive features accompanied by the wireless networks. These features can be enumerated as node mobility, flexible topology structure, leave and join the networks freely. The wireless communication

offers several services such as; network satellite services, cellular communication services, and wireless Ad Hoc network services. Wireless Ad Hoc network utilizes bands of radio frequencies for transmitting and receiving information. It is basically formed by autonomous hosts without needing to use any fixed infrastructure; in contrast with the other wireless communication systems like cellular networks and satellite systems that their need to some of pre-existing equipment and controlling units to implement their functions correctly. There are many challenges facing the wireless communication systems that doesn't present in wired systems. Usually, wired computer networks such as local area networks (LANs) has a lower transmitting bit error compared with wireless networks, also it’s easy to detect the collision occurred in wired channel. On the other side, wireless networks, usually, exhibit high collision probabilities between broadcasting packets and they are more difficult to detect [2].

Wireless Ad Hoc networks comprise mobile nodes as well as fixed nodes that are engaged with the network to improve its flexibility. Moreover, it makes the network more dynamic and attractive. These new versions of wireless networks, which did not need any administration controllers for their communication operations, were called mobile Ad Hoc networks (MANETs). MANETs were introduced to serve as a peer to peer service for specific purposes. Easy node deployment and self-connectivity properties extend the network applications for disaster fields and emergency operations. One of the main challenges of MANETs is to find the efficient scheme to connect the nodes due to the frequent network topology changes. So a number of routing protocols have been proposed to accomplish the nodes connection task efficiently. Further information about routing protocols presented in the forthcoming chapters.

1.1 Problem Statement

MANETs operations are influenced by different network environment factors. Node mobility, network load density, scalability, traffic loads among others re some factors that effectively have impacts on the MANETs performance. For that reason, computer network developers suggested different kinds of routing protocols to improve the network performance and to satisfy the different environmental operating conditions in real world. So far, routing protocol researchers have been studying them and suggested routing schemes to enhance the routing performance under various environmental aspects by using different network simulators. Unfortunately, there's no routing protocol found that satisfy all requirements for efficient MANET operations under different environment conditions [3].

MANETs considered as an important network for short distance communication with distinct applications. MANETs performance improvement of the current routing protocols is still prominent in the areas of continuous researching field.

1.2 Motivations

The rapid growth of wireless mobile communication technologies and applications provides an intelligent, fast and variety of communication services to the users. Recent advancements of MANETs allow the user applications to access to the internet services irrespective of their positions. These services encourage the researchers to design a newrouting protocol, or to improve existing ones. These issues were considered as a good reason behind studying, improving and testing various aspects of MANETs protocols.

1.3 The Aim and Contributions

The purpose of this work is to improve the performance of Ad Hoc On Demand Distance Vector (AODV) routing protocol by improving the selection method used to construct a stable route (more route stability selection) from source to destination nodes in a MANET. A fuzzy logic inference system is proposed to achieve this task by selecting the best route scheme. The following objectives are addressed through this research:

- Study the background of the wireless communication networks and mobile Ad Hoc routing protocols characteristics in order to understand the features of routing protocols used in the MANET and its operation.
- Survey the route stability and fuzzy logic techniques literatures that have been used in improving various aspects of MANETs routing protocol.
- Implement and investigate the proposed Fuzzy Inference System (FIS) added to the AODV routing protocol and modify it by using C++.
- Analyze the performance metrics (packet delivery ratio, average throughput, average end to end delay, and average routing load) obtained from the simulation study of different network parameters (number of nodes, nodes speed, and node pause times) results and discuss to conclude.

Chapter 2

BACKGROUND OF WIRELESS MOBILE AD HOC NETWORKS

2.1 Introduction

Todays, communication technologies are considered to be one of the main progression criteria used in determining a country's welfare. Computer communication networks are rapidly evolved and are extensively used in various fields of human activities. Computer networks typically consist of a number of digital equipment (e. g. Personal computers, printers, mass memory devices, etc.), that are connected to form a temporary data communication network to exchange and share resources between them as shown in Figure 2.1.

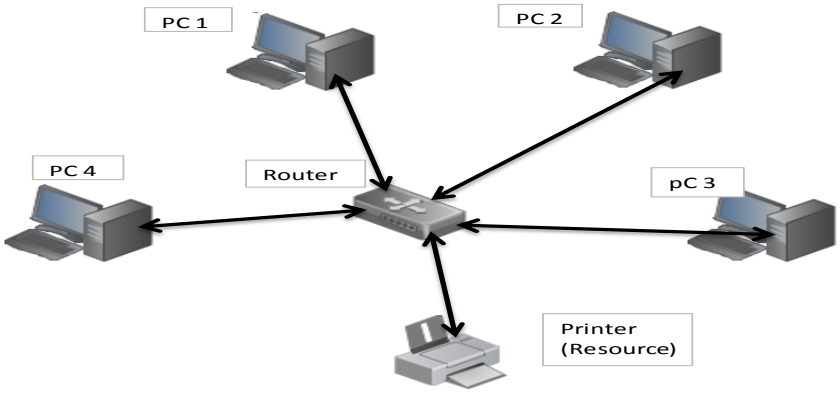


Figure 2.1: Classical wired computer network

In the last decade, links between computers changed from wired links to wireless. Wireless communication networks, nowadays become more popular networks. They have significant importance in computing and research community environments. Wireless networks perform a wider coverage range of communication that makes the

contacts between users easier at anytime and anywhere. Also, wireless networks introduced powerful and flexible communication services to the users. Computer networks have enormously extended their services to the several of data applications. At the beginning, the users' apparatuses have to connect to the base stations wirelessly. Then, the base station towers provide a facility to access to the other users at different places around the world. However, the wireless channel offers a lot of opportunities for wireless communication services. Ad Hoc networks introduced as an example of exploiting the wireless feature to connect a large number of digital equipment to form a temporary wireless network. These networks have attracted the computer developers' attention about the concepts and ideas of new applications that might be created. They have utilized them in different industrial and practical application fields, such as robots, banking operations, e-learning systems, distance conferencing and meetings.

The infrastructure wireless network generally consists of mobile devices connected wirelessly through each other via a centralized controller called Base Station (BS) or access point (AP) as shown in Figure 2.2. The centralized controller can provide the abilities for nodes to connect with other networks through it. Also, the wireless connections permit the nodes to move freely within the coverage communication range of the base station.

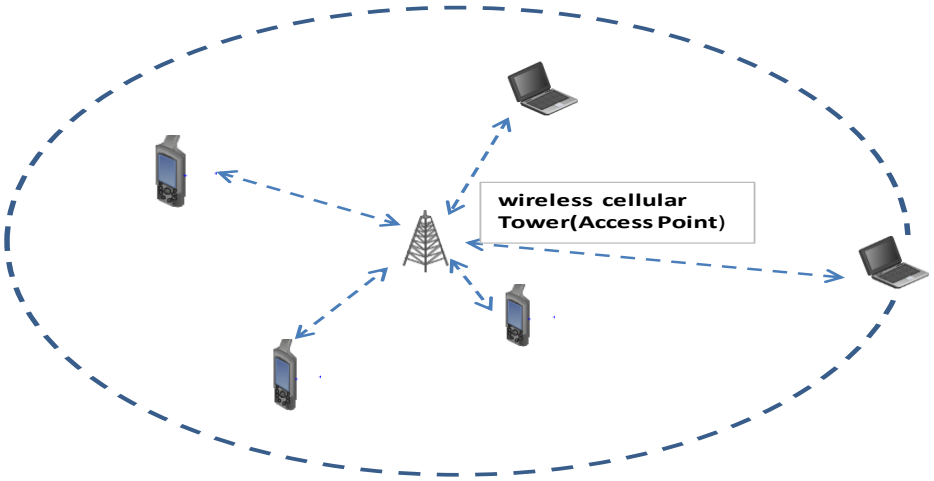


Figure 2.2: Infrastructural wireless network

Wireless communication environments face different problems related to the electromagnetic wave propagation in free space medium, such as reflection, diffraction, and scattering. These problems may degrade the network performance in wireless environments in terms of increasing transmission bit errors and minimizing data rates. MANETs comprise of a group of wireless stations (nodes) that are communicating with each other to form a short live wireless network without having to use a centralized administration and without the need for any pre-existing communication infrastructure (infrastructure less network). The nodes in MANETs possess the ability to create their own wireless network instantaneously. The nodes can randomly move in any speed or direction within the networks. As a result, the node location would be changed and thereby changes the communication links between the nodes in MANETs.

MANETs have received considerable attention due to the rapid deployment of wireless mobile networks in many emergency cases, such as disaster areas, search and rescue operations, conferences and battlefield operations make these networks

more practical and attractive, where there is a little or no time available to build a service communication infrastructure network as shown in Figure 2.3.

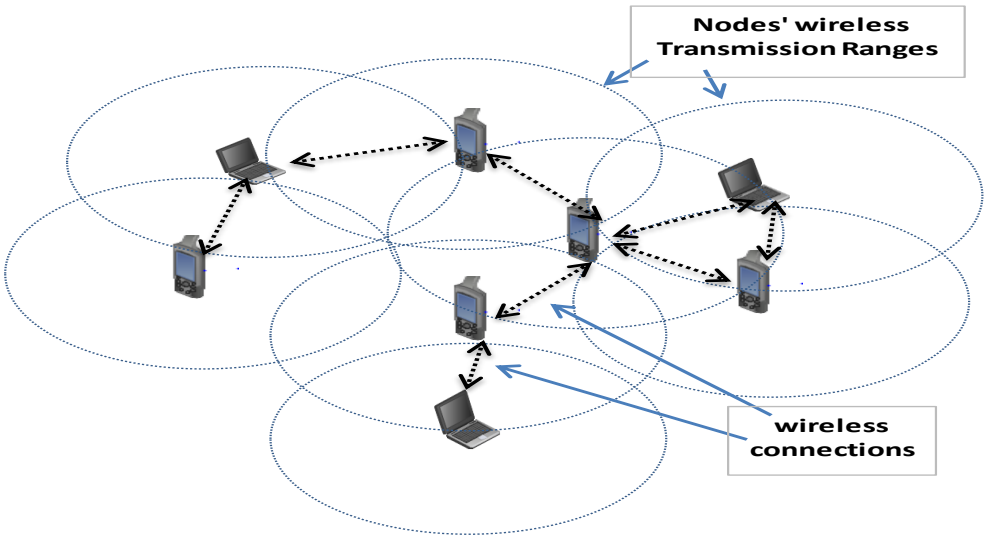


Figure 2.3: Wireless mobile Ad Hoc network (MANET)

Usually, the nodes in wireless mobile Ad Hoc networks can connect and interact with each other via wireless multi-hop scheme, in contrast, with the classical wireless networks which uses a single hop scheme to communicate between the users and network base stations. The multi hop communication scheme assists the wireless network nodes to preserve their energy and prolong the network lifetime [4].

There are many objectives needed to be considered in MANET's performance, like network throughput or packet delays which directly affect the multimedia application quality. Node mobility may cause different problems, for example it can increase the probability of packet delays and decrease the network throughput. Consequently, degrading the performance of online service application. Furthermore, channel capacity does not fully exploit in wireless networks because of exposed and hidden node phenomenon problems [5].

The conservation of node's energy is crucial in wireless Ad Hoc networks that assist to prolong the network life, especially when the node operates in disaster fields environments. As the nodes in mobile Ad Hoc networks consume energy from its limited resource "battery energy", the network partitioning can occur. So, there will be more than one individual wireless networks within the limited area and the network may no longer fulfill its intended functions.

2.2 Characteristics of Mobile Ad Hoc Networks

A MANET is an autonomous wireless communication system. There exist a lot of queries about the effective operability of wireless connections of mobile nodes under different environments difficulties. It's clear that many consideration factors must be taken into account, that is inherited directly to the wireless Ad Hoc networks such as: the amount of bit errors occurring at the receivers, due to nodes mobility compared with classical base station infrastructure networks, the lack of security in MANETs, which its ease to anyone to join or to be a member of a MANET's group, asymmetric of link's channel conditions, and limited bandwidth available for data transferring within the network. Depending on the construction topology nature of mobile Ad Hoc networks, the networks have several features and notable characteristics that can be summarized as follows [6]:

- Limited energy resources; wireless devices energies due to limited battery resource may have short lifetimes and it is difficult to replace the battery in some specific environments as in disaster areas or battlefield operations.
- Infrastructure-less communication system; the wireless Ad Hoc mobile system is an infrastructure-less system and the nodes are self-organized themselves arbitrarily to form a wireless network. Also, the structure of this

network would be in decentralized case where there's no administration or controller exits.

- Wireless multiple hops scheme connection; there are a lot of energy consumed from node battery resource when packets transmitting through the free space channel and its consuming power would be proportional to the squared of the distance between receiver and transmitter nodes [7].
- Nodes operation roles as a host or a router; each node in the wireless Ad Hoc network can act as a router, by forwarding information and data packets to neighbor nodes, and the MANET node can also act as a host node
- Variable link capacity with limited bandwidth constrains; wireless links in MANETs, in general, have lower capacity compared to wired link networks. The effects of the wireless environment (noise, multiple access, fading, and conditions of propagation interferences) result in reducing the amount of data packets received by receiver in MANETs. Besides, many of routing protocols used consumes considerable channel bandwidth in discovering route process [8].
- Node mobility and dynamic network topologies; in MANETs, free movement of nodes with different speed and direction within the network, leads to unpredictable network topology changes over time. The dynamic topology feature of MANETs has to be supported by the decentralized mechanism that helps MANETs to be robust network against the single point failure problems [9].
- Limited physical security; wireless mobile Ad Hoc networks are easier to attack by malicious threats than wired networks due to the fact that a node can join and leave randomly without verifying its identity before allowing to

connect. The lack of centralized control prevents to distinguish the nature of attacks because it is difficult to monitor all data traffic exchanges between highly dynamic topology networks [10].

Wireless mobile Ad Hoc network characteristics are representing a big challenge that faces routing protocol designers. Designers should consider these characteristics in order to enhance the routing protocol performance and to treat the weakness of any protocol that would be designed in the future.

2.3 Mobile Ad Hoc Network Applications

There are numerous fields extended for wireless communication network applications, especially for MANETs and some typical applications that can be mentioned are as follows [11]:

- Military operations: battlefields are one of the most dangerous places that one needs the advantages of mobile Ad Hoc communication technology to keep information exchanges between the soldiers and their commander.
- Conferences and urgent meetings; a local level communication application, requires to share the information between peoples in the classroom or conference, this can be done using a MANET service which provides instant connections between users with multimedia services.
- Bluetooth and personal services network; Bluetooth is a short range network, where various digital equipment (PDA, Laptop, Mobile Phone, printer, etc.), can be interconnected between them to form a simple wireless Ad Hoc network.
- Public Sector; MANETs may be used in rescue and emergency operations, where there are no communication services available or that

damaged accidently. The rapid network establishment is needed for assisting in natural accident, such as earthquake places, floods, and forest fires.

2.4 Challenges and Complexities of Mobile Ad Hoc Networks

Although, MANETs are expected to be more popular and versatile network compared with the other networks, still, there are some challenges and complexities that should be solved to enhance the network performance. These challenges and complexities are discussed below [12]:

- Routing: is one of MANETs protocol issues that determine the best path that should be established to relay packets between the nodes in MANETs. Frequent topology changes, network security and node residual energy are enumerated as some of the challenges and complexities facing MANETs routing developers. The routing protocol developers should consider such challenges when they designs and tests a routing protocol.
- Node's power consumption: most nodes are energized with limited power resource (battery powered). In order to prolong MANETs lifetime, it's required to design an efficient routing protocol that conserves the nodes' resource energy and consume minimum possible power of MANET nodes.
- Quality of Service (QoS): Multimedia services qualities provided by mobile Ad Hoc networks have to maintain at an accepted level of quality for different applications. This is a big challenge in MANETs that operated in different interfering operating environments.
- Reliability and security: due to wireless connection nature, mobile Ad Hoc networks have many problems related to vulnerability and security issues. The operation of distribution form feature of nodes in MANETs that exhibits

authentication security difficulties with key management. Furthermore, hidden and exposed terminals in wireless network communication introduce an addition problem with the reliability issues.

- Network scalability: nodes in MANETs can join or depart arbitrarily, so mobile Ad Hoc network needs to be flexible in handling the network scalability without degrading the network performance.

The development of mobile wireless Ad Hoc networks in the research community are subjected to a large number of challenges corresponding the routing protocols, mobile devices, services and applications.

2.5 MANET Ad Hoc Routing Protocols

A routing protocol is a core element of a wireless Ad Hoc network. A routing protocol is an algorithm needed whenever a node in a MANET has information to relay to another node through the intermediate nodes within the network. The function of the routing protocol is to guide the source packets to the final destination by finding the best route available to the correct destination node. Different concepts of routing protocols are studied to improve its performance in order to design an ideal routing protocol that satisfy all requirements of a wireless network communication environments. Desired properties of routing protocols should meet some specific application and network topology requirements. In order to achieve the optimal design of routing protocol, the main properties of required routing protocols should be studied. Some of the desired properties required for different application kinds can be defined as follows [13], [14]:

- Nodes energy conservation techniques: nodes in MANETs, in general, are light weight devices (mobile phone, Laptops, PDAs and so on) that have

limited energy resource, for this reason, sorts of power saving techniques are used in designing routing protocols, for example, by supporting methods of nodes sleeping states in MANET nodes.

- Loop free: to avoid any additional nodes' CPU cycling resource and waste the network bandwidth or avoiding extra overhead control packets broadcasted through the networks, the routing algorithms should be loop-free algorithms. This surely leads to enhancing the routing performance. Thus, it is preferred the protocol to be in reactive behavior protocols, that means the protocol reacts and activates only when there's a need to transmit packets.
- Distributed operation: The desired routing protocol has to operate in distributed form, because the nodes in MANETs may join or depart the network randomly and the network may be partitioned at any time. The distributed operation without any controller centralization is suitable in MANETs.
- Multiple valid routes: In order to make routing protocols more flexible and efficient, multiple routes are important issue which improves the route choices. Due to the rapid changes of network topology and congestion occurrence in wireless Ad Hoc networks, when a route is broken unpredictably, to use another previously stored route will be possible. The multiple route will prevent initiating the route discovery process many times during a transmission and prevents wasting network resources.
- Quality of Service (QoS) requirements: routing protocol may support some of the QoS features that are required for the protocols' quality assessment. This may help to realize many real time applications such as audio–video services.

- Network security issues: radio propagation environment characteristics makes network vulnerable to attacks. Security problems in MANETs can be solved partially by using Authentication and Encryption schemes provided to MANET node software that falls within a key distribution among MANETs.

Unfortunately up to now, none of the available proposed routing protocols of wireless Ad Hoc network have all desired properties required. The routing protocol is still under development and attracts researchers'. Also the routing protocols are perhaps extended with additional functionalities.

Depending on their properties, routing protocols may be classified into categories as [15] [16].

- Centralized vs. Distributed protocols.
- Static vs. Adaptive protocols.
- Reactive vs. Proactive protocols.

The administrator node in centralized algorithms is responsible from all decisions for the route selection, where, in distributed algorithms, all nodes are contributed in the computation for the route selection among choices.

The adaptive routing protocol related to the interaction with the network traffic densities, selects the best routes as response to the minimum traffic network load, on the other hand, the static routing algorithms where the source-destination route pairs are unaffected by the traffic load conditions, route changes only as a response to the failures in node operation and/or link state conditions. To investigate the optimal

routing performance, most of the computer networks utilize adaptive algorithms depending on the changes in the traffic congestion state.

A third classification of routing protocols based on the instants when the data is available at a node and it is needed to send the data to another node within the network. These kinds of algorithms are classified as proactive and reactive routing protocols. Reactive routing protocols run the route establishment process whenever the data packets are ready to relay from source to the final destination. Reactive protocols are used in broadcasting or flooding schemes for route searching technique and they are related to the on-demand algorithm protocols. On the other hand, the proactive routing algorithm continuously updates the information of route states. Proactive algorithms choose one of the previously stored routes in the nodes' route table to transmit source packets immediately. The families of Distance-Vector protocols are an example of the proactive routing protocols. Proactive routing protocols, in general, have a small delay for packet relaying to destination as compared to the high delay times that the data packets last to arrive to the final destination due to the route discovery process achieved by the reactive routing algorithms group. In contrast, proactive routing protocol algorithms try to converge to a steady state at each change occurrence in network topology. This can cause serious problems if the network topology has frequent changes.

A hybrid routing protocol combines the features of reactive and proactive routing protocols. It takes the advantages of proactive discovery property within the local neighborhood nodes (the first hop neighbor nodes) and uses an on demand protocol property for connection between these neighborhoods. Zone routing protocol (ZRP)

is one such hybrid routing protocol implementation. Figure 2.4 shows a list of routing protocols gathered by Halvardsson and Lindberg in [17].

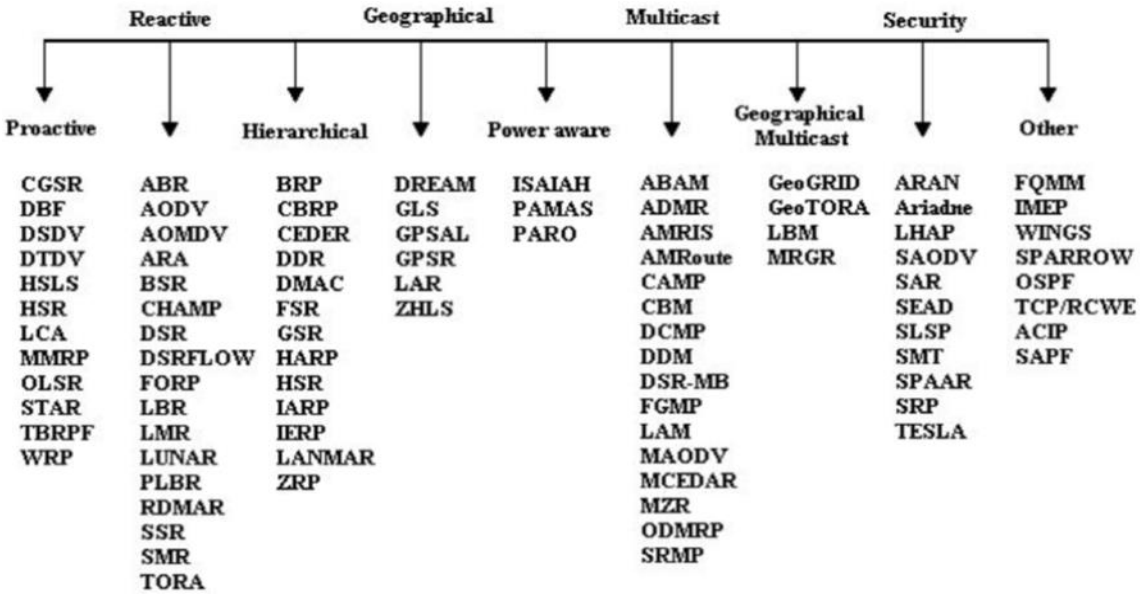


Figure 2.4: Ad Hoc routing protocols overview [17]

Chapter 3

AD HOC ON-DEMAND DISTANCE VECTOR ROUTING PROTOCOL (AODV)

3.1 Overview of Classical AODV Routing Protocol

AODV is one of the most popular wireless mobile reactive routing protocols used in the mobile wireless research environment. It supports multicast and unicast routing protocols. It can act as a reactive routing protocol based on (On-Demand) algorithm. AODV algorithm starts when a source node has data packets to transmit to some destination nodes. At the beginning, all nodes in the network periodically send a Hello message to neighbors to check neighbor's node connectivity. Then, AODV protocol initiates route discovery process; it constructs routes between nodes only when the source nodes ask for. The two main phases proposed in AODV are route discovery phase and maintenance phase. The protocol keeps these routes as long as necessary. The source node starts a route discovery process whenever it has data packets to be sent. It floods a route request packet (RREQ) to all neighbor nodes in coverage transmission ranges. The node receives the RREQ packet, determines if it has a fresh route to the destination or is itself the destination, it replies back in unicast form a route replay packet (RREP) to the source (initiator) node. If a neighbor node does not has information of the route to the destination node, it will rebroadcast the RREQ packet to its neighbor nodes and so on until it arrives to a node that has fresh route information to the destination or itself is the destination as shown in Figure 3.1.

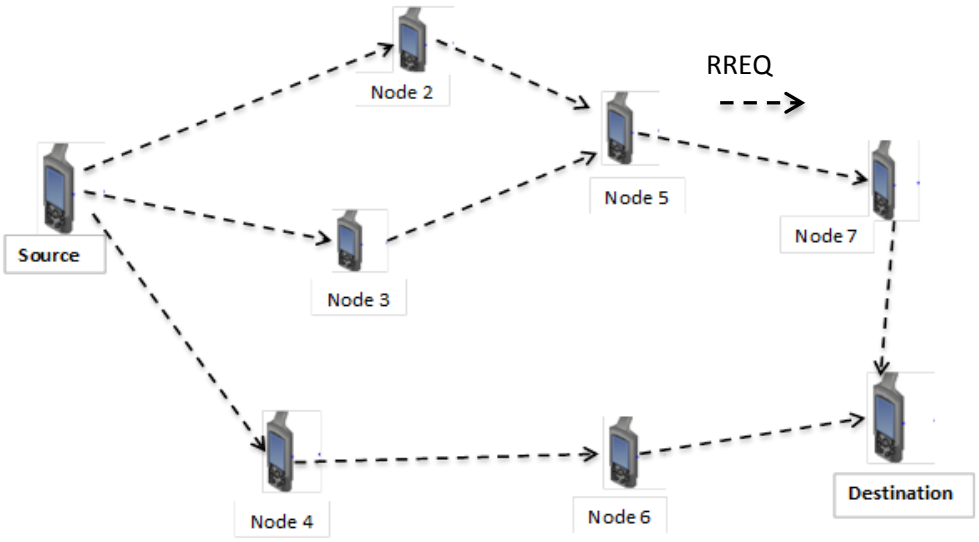


Figure 3.1: Propagate RREQ packet

Intermediate nodes, that forward a RREQ message, stores in its own routing table the addresses of the nodes that the RREQ packet came from. Each node includes two counters, one for counting the node's sequence number (to avoid the loop problems) and the other one for the broadcasting identification (ID) which is incremented when a broadcast is initiated in the node. To identify just one RREQ packet, the ID and the address of the source node are used. The RREQ packet format includes source's address and sequence number, broadcast ID, destination's address and sequence number, and the hop count, as shown in Table 3.1.

Table 3.1: RREQ packet format

Source Address	Source Sequence No.	Broadcast Identification	Destination Address.	Destination Sequence No.	Hop Count

The intermediate nodes which receives the RREQ packet and have information for the required path to destination, replies with the RREP packet. The RREP packet will be transmitted in reverse unicast route to the initiator source node. RREP packet

includes information about the fresh route available to destination that may be used to transmit source data packets. The fresh route to destination is investigated only when the intermediate node routing table contains a destination sequence number equal or greater than the sequence number carried by the RREQ packet (same or more recent sequence number). The intermediate nodes increment the hop count number during the broadcast of RREQ packet. Also, it stores in its own routing table, the address of the neighbor node which sent the RREQ packet to provide a return back route. Same RREQ copies received later, which are coming from the other neighbors, may be discarded if it has higher hop count number than the previous RREQ packet received. Figure 3.2 shows the reverse unicast path of the replied RREP packet.

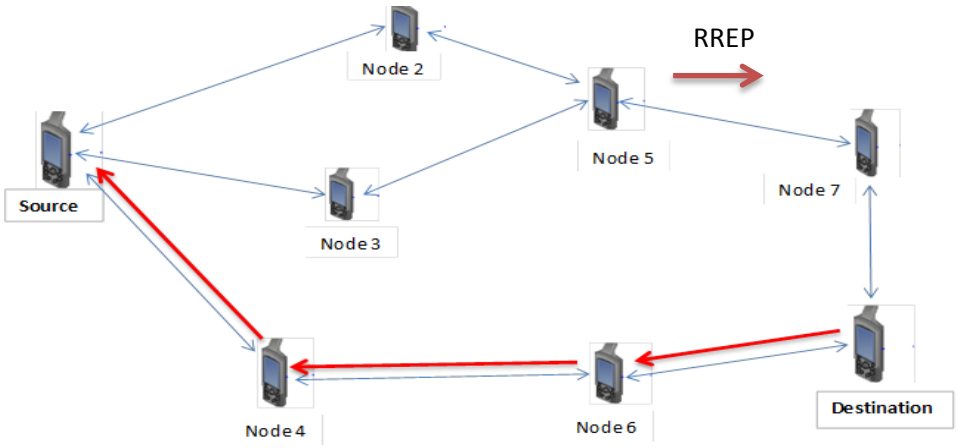


Figure 3.2: Path of the RREP packet

The RREP packet format includes information of source and destination addresses, number of hops to the destination, new destination sequence number, and reverse path expire time, as shown in Table 3.2.

Table 3.2: RREP packet format

Source Address	Destination Address	Hop Count	Destination Sequence Number	Expire Time of Reverse Path

Each intermediate node forwarding the RREP updates the return path information as the freshest route to the final destination node. Thus, AODV utilizes a bidirectional links channel. Route failure occurs when the nodes depart out of the transmission coverage area of neighbors in constructed route, then the route error (RRER) packet created. RRER is used to inform the source that the constructed route is no longer valid. Source node will start a route discovery process in order to find a new path to destination if there is more data to send or the route to destination is still needed [18] [19] [20].

The classical AODV protocol is a single metric routing protocol that uses the minimum hop count (shortest path) parameter to select the route to the destination. This selection occurs without considering the nodes' specifications and abilities to construct a long life or trusty route. AODV protocol, sometimes, called a shortest path (SPAODV) routing protocol [17]. If multiple RREQ packets arrive to the source node, then the source node will select the shortest hop count route.

Usually, intermediate nodes receive many of RREQ packets of same Identification (ID) and sequence number but with different hop count values from its neighbors. Hence, the node examines each RREQ packet individually. If it has a lesser hop count value than previously received RREQ packets with the same ID, then the node updates its reverse route table and rebroadcasts the RREQ packet. Otherwise, it discards the RREQ. For that reason, an intermediate node may propagate the same

identification RREQ packet more than once, as illustrated in Figure 3.3. Thus, one of major drawbacks of the classical AODV protocol is unnecessarily consume more energy, waste network bandwidth resources, and increase network traffic, especially in high density wireless mobile Ad Hoc networks [21]. Figure 3.4, shows a flowchart of classical AODV algorithm.

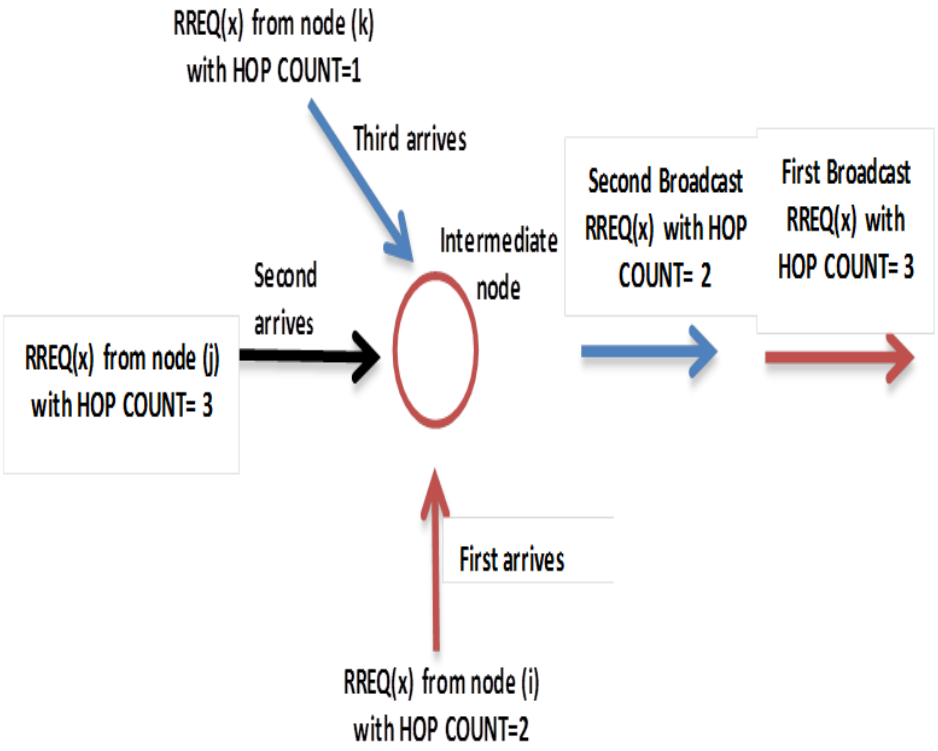


Figure 3.3: Intermediate node RREQ broadcasting in classical AODV protocol

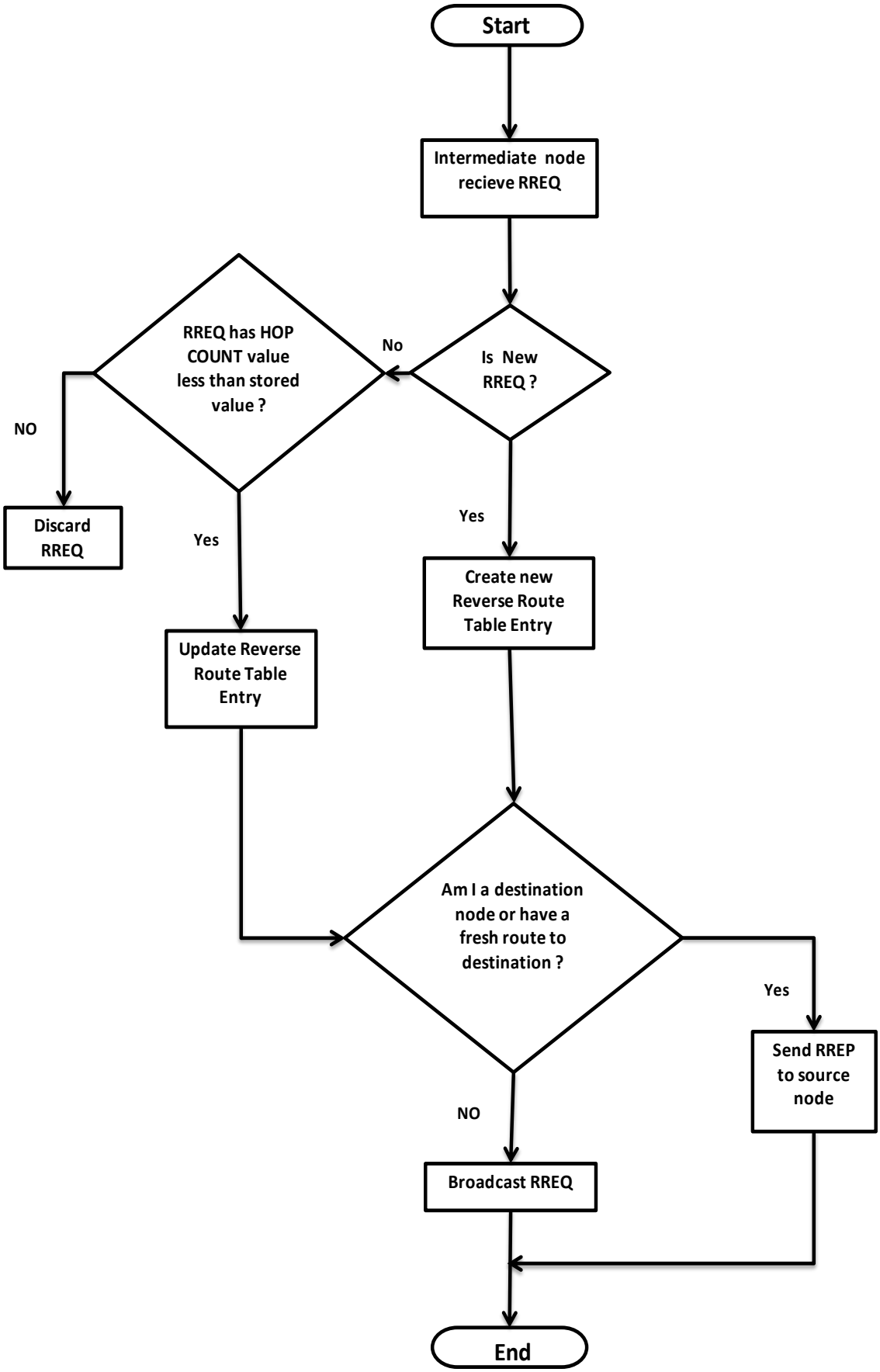


Figure 3.4: Flowchart of route requesting in classical AODV algorithm

3.2 Drawbacks of Classical AODV Routing Protocol

AODV routing protocol presents some problems related to reactive (on-demand) scheme nature. These problems can be summarized as follows [22]:

- Redundancy of route discovery: AODV discovery stage, usually, requires broadcasting a lot of control packets to achieve the path discovery process correctly. High amount of flooded RREQ packets cause unnecessary load and consume limited network resources, the amount of control packets increase proportionally with the network's node densities. Several of the RREQ packets may retransmitted again, due to the packet collisions and channel occupation.
- Message duplication: intermediate nodes receive multiple RREQ packets of the same identification from neighbor nodes. The intermediate nodes have to (in specific conditions) rebroadcast them again. This rebroadcasting scheme of the same identification RREQ packets will increase network traffic load and consumes extra battery energy.
- Insufficient metrics for route selection: classical AODV utilizes a minimum hop count (shortest path) as a decision metric to construct a route between source and destination without considering other important network parameters such as node residual energy, node speed, the strength of received signal among others. Although, this metric may achieve a short delay for packet's transmission, it could not establish a robust and prolongs life route, because it does not consider important parameters in selecting trusty nodes in route construction process. That's, surely, cause frequent route links breaks through the route's life period.

Chapter 4

LITERATURE REVIEW OF ROUTE STABILITY TECHNIQUES

Route Stability represents the quality and life time of the established route between source-destination pair nodes which confirm the consistency of the network environment. It addresses to how a stable route has been built and which parameters can support the route prolong in MANETs. Selection of a stable route from source to destination nodes is considered as an important issue in wireless mobile Ad Hoc networks. Variations of the network parameters such as node mobility, residual energy and environment signal interference cause frequent change of the network topology. So, constructed route has no longer valid and alternate route must be established. In order to avoid MANETs performance degrading, several strategies have been proposed considering different schemes to improve the route stability in MANETs.

Received Signal Strength (RSS) schemes have been proposed in [23], [24] as an indicator to the link lifetime. In this approach the intermediate node forward the receiving control packets only when the packets signal strength exceeds the predetermined signal strength threshold, otherwise it discards the packets. The routing protocols adopted this approach uses MAC layer's signal strength values to determine the neighbor's link stability. It contributed to utilize minimum control overhead packets. The authors in [25], suggested a new method to reduce the effects of link failure in mobile Ad Hoc network. They defined a signal strength parameter

in determining a stable path for packet transmission. High speed stable routes are required to ensure a better packet delivery ratio between the network nodes. So, dynamic switching between the nodes introduced by the authors. Also, they suggested a method to select a neighboring node with maximum signal strength for data transmission. The scheme used to ensure the stable route path, and reduces the hop counts between the source - destination pairs. However, in the urban area, shadow effects may influence effectively on the degree of the signal strength received by the intermediate nodes, which increases the probability of errors in computing the RSS values and consequently, fail to predict the link stability.

Pilot signal or Hello packet based scheme proposed in [26], [27], [28]. The periodic broadcasting of Hello packets in AODV routing protocol could be used to verify the link connectivity of the neighbor nodes. The nodes in AODV routing protocol broadcast Hello messages periodically to identify them for one hop neighbors. The continuous receiving of these Hello packets pointed to the existence of its neighbor's nodes. When a node leaves out of neighbor's nodes transmission ranges, the receiving nodes of Hello packets would record the link failure of this node. The concept of this approach is to construct routes with more stationary nodes over the less stationary ones. So, the route's lifetime of stationary nodes tend to be longer than the route's lifetime constructed with high mobility nodes, hence it's considered to be more stable route.

Quality of Service (QoS) route stability scheme suggested in [29]. The proposed route algorithm avoids the weak links during forwarding the route request packets across all possible paths available from source to destination nodes. In this scheme, the source node generates a QoS packet (QRREQ), which relays through the

network. Each intermediate node receives the QRREQ, drops this packet if its signal strength value is less than the specific threshold value. Otherwise, the intermediate node save the address of the node which the QRREQ packet coming as the reverse route path. At the destination node, a timer is set up for a fixed period of time (called Route Reply Latency (RRL)) when the first QRREQ packet received. Then, the destination node selects the best reverse path among all feasible paths after the timer expired. The selected path has the highest route stability value compared with the other reverse paths from destination to source. This will assist increasing the network throughput and decreasing the packet delay.

Power aware route stability schemes are suggested in [30], [31]. The proposed schemes based on examining the route link stability, residual node energy, and then predict the probability of route failure. The authors suggested an algorithm to calculate the link stability, maximum mobile nodes life time and minimum energy consumptions in order to select the optimal route to destination. In [30], authors divide the node transmission ranges into three coverage zones. Stable zone (which has highest stable link connection with the neighbor node), warning zone (which has a worst link connection with neighbor node), and the buffer zone (which has a critical link connection with neighbor node). Also, they suggest a mathematical expression to calculate the link stability which uses the relative velocity between one hop neighbor nodes. They concluded that the link stability is inversely proportional with the relative velocity of one hop neighbor nodes.

Many routing protocol algorithms such as Ad Hoc on Demand Distance Vector (AODV) and Dynamic Source Routing (DSR) protocols utilized a single metric in determining route between source-destination nodes. They are not considering

effective parameters which influence the network performance such as node mobility or energy aware in their routing protocols algorithms. These schemes may lead to select unreliable route, which lead to the frequent network partitioning and minimizing the route lifetime.

Route selection that satisfies a multiple objective metrics is a hard computational task which requires some approximate and heuristics solutions [32]. The mechanism of a stable route selection requires different information about the intermediate nodes and route environments such as nodes remaining energy, route traffic congestion, mobility, number of intermediate hop count, and signal propagation medium. In multiple objective routing schemes, each objective links to different network metrics. For example, an end to end delay objective metric depends on the route traffic congestion and the numbers of hops from source to destination which are directly influence the frequent route failures. The failure of routes is the major reason that stimulates the routing protocol to discover a new route to destination. This will increase the packets waiting intervals in the sender's buffer before resending the packets again. Control overhead packet is another important objective, which limits the network scalability. It depends on the route length and the route stability. Route failures increase the amount of the packet control overhead broadcasting and it reduces the probability of network scale.

Multiple objective route selection became a more complicated issue when there is no rigid information available about the node and links environments. Under the uncertainties conditions and randomness of the intermediate nodes environments, it's better to combine several metrics using fuzzy logic inference system to improve the performance of routing protocols. Fuzzy logic theory is proved to be a good approach

for routing in the Ad Hoc networks. The advantages of fuzzy logic are its simplicity, flexibility of combining conventional control techniques, ability to model nonlinear functions and imprecise information, use of empirical knowledge and dependency on heuristics. Fuzzy logic can be used to solve the problem of routing in Ad Hoc networks where the final outcome is based on the factors with uncertainty [33]. Developing a fuzzy based protocol for mobile Ad Hoc networks has been proposed as an adaptive field research in the few past years. Table 4.1, summarizes the comparison of various fuzzy based routing protocols utilized to enhance different MANETs objective performances including the proposed Fuzzy AODV.

Table 4.1: Comparison of fuzzy logic based routing protocols

Approach protocol name	**Base protocol**	**Input fuzzification metrics**	**Output defuzzifica-tion**	**Remarks**
FCMQR [34]	AODV	Band Width, Delay, Hop Count	higher link stability, lower cost	Approach is used to select stable and least congestion route
FMRM [35]	AODV, AOMDV	Expiry Time, Data Rate, Queue Length	average link-connect time, the success rate to find the path	Approach is used to reduce the number of route reconstruction.
FLEAMR [36]	AOMDV	Delay, Avg. Load, Band Width, Residual Energy	load distribution possibility	Proposed approach determine the traffic distribution over fail-safe multiple routes to reduce the load at a congested node
FBERP [37]	AOMDV	Packet Loss Rate, Communication Rate, Energy, Delay	priority of a node (a node with maximum throughput is selected)	FBERP is used for route discovery and maintains the route dynamically in case of node failure.
FLBSRP [38]	AOMDV	Mobility factor, Residual Energy	probability of link stability	proposed protocol measures link and node stability together using two metrics

FRPM [39]	MANET protocol	Delivery Predictability, Power Remaining, Number of Packet Copies	probability to send a packet copy	The approach find good packet routes that maximize their delivery probability and minimize the delivery costs.
Fuzzy-ABR [40]	MANET protocol	Route Reply, Route Request, Route Error, Data Delivery Misbehavior	trust of a particular node	Routing approach to improve QoS and to mitigate network attacks
FBSRA [41]	DSR	Velocity Neighbor Nodes, Distance Between Neighbor Nodes	link stability index	The approach is used to increase the reliability during the routing selection and reduce the number of broken routes efficiently
AODVFHI [42]	AODV	Transmission Power, Mobility	value of Hello interval	an efficient approach to optimize the frequency of sending hello message
RRAF [43]	AODV	Trust Value, Energy Value	reliability value	The approach increases packet delivery ratio in the face of node mobility and route breaks
ERPN [44]	DSR	Noise Factor, Signal Strength	probability	The approach is an efficient routing for transmission of data packets.
FLBDR [45]	MANET protocol	Signal Power, Bandwidth, Mobility, Packet Forwarding Ratio	optimal path	new dynamic routing protocol is proposed that has the capability of intelligently selecting an optimal route.
FQURM [46]	unicast routing	Band Width, Link Delay, Link Reliability	link status	The approach evaluated QoS acceptance ratio, route discovery time and bandwidth utilization
OMDRP [47]	OMDRP	Data Rate, Expiry Time, Queue Length	priority index	The approach is used to schedule the data packets based on their respective priority index
FLGBRB [48]	MANET protocol	Node Degree, Residual Energy, Node Velocity	retransmission probability	The approach technique for gossip based reliable broadcasting in MANETs
Fuzzy AODV (proposed algorithm) [56]	AODV	Node Speed, Residual Energy, Hop Count.	node trust value (a node having the lowest probability of broken connection)	Approach constructs the most stable route via calculating the node trust values at each hop.

The proposed algorithm (Fuzzy AODV) changes the selection method used to construct a route from source to destination nodes in MANETs via taking into account the parameters: residual energy, hop count and node speeds. A fuzzy logic inference system is proposed to achieve the task of selecting the route scheme. Fuzzy AODV explained in details in Chapter 5.

Chapter 5

FUZZY APPROACH: IMPROVING AODV ROUTING PROTOCOL

In this chapter, fuzzy logic concepts are presented to modify and improve the decision making of route selection strategy of the classical AODV routing protocol. The fuzzy logic membership parameters are introduced with fuzzy based rule explanations too.

5.1 Introduction

MANETs have received considerable attention over the past few decades. The rapid deployment of wireless mobile nodes of a MANET in many emergency cases such as disaster areas, rescue operations, conference meetings, and battlefield operations make these sorts of computer networks more attractive to a wide range of application usages. A MANET is composed of mobile nodes that temporarily communicate to form a special sort of wireless network. The nodes in the MANET organize and configure themselves dynamically without the need of an administrator or central controller system. Each node in MANETs can join or leave the network arbitrarily and is free to move at any speed and in any direction independently. Battery powered devices, such as laptops, PDAs, or smartphones, are widely used in MANETs as mobile nodes. The limited energy resources of these devices forces MANETs developers to adapt a multi-hop communication strategy in order to preserve the node's energy and prolong MANETs lifetime [49]. Unfortunately, route failures frequently occur in MANETs because of the node's mobility, limited energy resources, and electromagnetic propagation interference among other reasons. Due to

this fact, the routing protocol algorithms should react rapidly to any environmental changes and reconnect the broken path links efficiently.

Many simple MANETs' reactive routing protocols use a single metric like the shortest path (SP), signal strength, or node battery's residual energy to construct the route for data transmission. This single-metric route selection is not sufficient to construct a stable route because it may cause frequent route failures that stimulate the routing protocol algorithms to rediscover a new route each time a route is broken. The route discovery operations consume extra network resources, degrading network performance, minimizing network lifetime, and leading to network partitioning problems. In contrast, improving the efficiency of the route selection scheme in a MANET can be achieved by combining multiple routing metrics using an adaptive intelligent tool to choose the most trustworthy nodes from which the best route to a destination can be constructed [50].

AODV routing protocol is one of the well-known reactive routing protocols in wireless mobile Ad Hoc networks. It uses the minimum hop count criteria (Shortest Path) to select the route for data transmission without taking into account a path's link stability factor or nodes' quality when constructing the route from source to destination. A node in a simulated network running the AODV protocol must flood routing control packets over the network each time it needs to discover a route to a destination. Such nodes are likely to exhaust their energy resources and deplete their battery power rapidly. Hence, node cooperation is needed to preserve wireless Ad Hoc network resources and support the wireless network performance effectively [51]. This ideal cooperative environment, generally, is not achieved in classical,

simple MANETs routing protocols. The behavior of MANET nodes changes continuously over time, depending on the wireless network environment. However, a variety concepts, schemes, and models have been proposed to achieve intelligent services and networks. Adding open programming and management abilities to the nodes can enhance the new network services. This feature of programmable network elements moves the control and managing network system toward an adaptively evolutionary computing system with a variety of genetic algorithms and evolutionary programming [52].

In this work, a Fuzzy Inference System is proposed as an adaptive computational approach to compute a node's trust value (stable nodes) and introduce an efficient routing scheme by selecting the most trustworthy nodes to establish a stable route. Using the concept of node trust when building stable routes decreases the probability of route breaks during the data relay period. This, consequently, minimizes the amount of unnecessary overhead control packets transmitted over the network in the route discovery stage. In addition, it preserves network resources and improves network performance.

5.2 Fuzzy Logic Concepts

Fuzzy logic is introduced and formulated by Lotfi Zadah in mid 1965s at California University in Berkeley [53]. Fuzzy logic theory is an extended version of classical Boolean logic algebra that based on the fuzzy sets of the mathematical theory. Introduction of the membership degree concepts of linguistic variables provides a good flexibility for intuitive reasoning with multi-level logic states (multi-valued states ranging between 0 and 1), rather than two levels (True and False) as in classical logic system. This will help to have and consider a partial truth instead of

absolute truth for representing uncertainties and inaccuracies. One of the benefits of using fuzzy logic sets is to formalize the human natural reasoning in a set of IF-THEN rules in the form of the human natural language; for example, if the weather is raining and the car's tire is bad then drive with Low speed. Also, if the weather is sunny and the car's tire is good then drive with High Speed. So, it's noted that the weather variable is categorized to raining and sunny, and car's tire is categorized to good and bad. The weather and tire condition represent the input variables and the speed represents the output variable [54].

A fuzzy logic scheme that deals with the reasoning algorithms is classified as a branch of artificial intelligence. It emulates the human thoughts and making a decision in many controlling machines. Usually, the fuzzy logic algorithms utilizes with the application environments where the data to process cannot be supplied in the digital binary formats.

5.2.1 Linguistic Variables

Linguistic variables are fuzzy logic variables that are used to represent a non-numeric variable. Sentences and words of natural human language can express the magnitude and importance of fuzzy logic variables. For example a temperature can be expressed as a linguistic value of Hot, Cold, Very Cold, etc. Instead of using numeric degree value at 40 ^{o}C . The fuzzy logic service variable can be expressed as Excellent, Good, Bad, etc., where service variable cannot be represented in numerical form.

5.2.2 Membership Functions

A membership function is a graphical representation of fuzzy logic variables. Each input and output fuzzy logic variable is defined by the value of membership's

functions that describe the linguistic variable graphically. The membership function is ranged between 0 and 1 and each point of the membership function curve represents the input degree (or degree of membership). The fuzzy input value, in some cases, can be a member of more than one membership functions at the same time. As an example, if temperature is categorized with two membership functions as Hot and Cold, as shown in Figure 5.1, then it can have two values at the same time for input temperature ranging between ($30 \leq T$ (temperature) ≤ 60), but with different degree of memberships. For example, temperature variable (Temp. = 38 ^{O}C) has two membership degree values equals to 0.22 Hot and 0.78 Cold at the same time as shown in Figure 5.1.

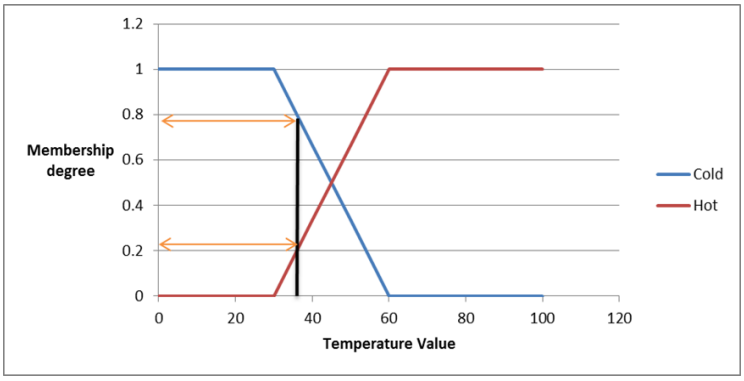


Figure 5.1: Two membership functions of temperature (fuzzy linguistic variable)

5.2.3 Fuzzy Logic Operators

The operators of the traditional two values logic system that applies in classical Boolean operations are AND, OR, and NOT. The logic operators can perform the operations of Intersection, Union, and complement respectively. Table 5.1, shows the truth table of the classical Boolean operations.

Table 5.1: Classical Boolean logic operations

A	B	A.AND.B	A.OR.B	NOT A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Fuzzy logic needs more additional operations that can consider all possible values represented by the membership functions, which includes the ranges of values between 0 and 1. The fuzzy logic operators are described in another mathematical notion to distinguish them over the classic Boolean operators' notion as shown in Table 5.2.

Table 5.2: Classical Boolean and fuzzy logic operators

A .AND. B	min(A,B)
A .OR. B	max(A,B)
NOT A	1 - A

Table 5.3, describes the operation of a fuzzy logic operator that covers the classic Boolean logic values and fuzzy logic operations [54].

Table 5.3: Truth table of fuzzy logic operations

A	B	min(A,B)	max(A,B)	1 - A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0
0.2	0.6	0.2	0.6	0.8
0.8	0.3	0.3	0.8	0.2
0.6	0.6	0.6	0.6	0.4

5.3 Proposed Fuzzy Based AODV Algorithm

In classical AODV, the minimum number of hops metric is used to make a decision about the route selection, but this is not a sufficient parameter for constructing the best route to destination in MANETs [55]. It does not consider other factors that may affect the route quality, like the received signal strength, node mobility, or node residual energy.

In our proposed Fuzzy AODV, important parameters such as node residual energy and node mobility are considered to construct a reliable route. Besides, the selection of high quality nodes will help to minimize the probability of route failure during data packet transmission. The choice of trustworthy nodes used to build a stable route in the proposed fuzzy algorithm is based on the nodes that have higher residual energy level and move with slower speed.

The proposed approach uses fuzzy logic techniques to determine a node's trust value by combining the residual energy and speed of each node in MANETs. The nodes with the highest trust values are selected to establish the best route available to the destination node. Each intermediate node calculates its trust value whenever it receives the RREQ packet. The intermediate node initiates a timer if the RREQ packet has not been previously received. During the timer duration, the intermediate node receives more RREQ packets (of the same identification ID and sequence number) from its neighbors. The intermediate node selects the node with the highest trust value (carried by the received RREQ packets) to update its reverse route table, which will be used to construct the reverse unicast route as a part of a reliable route establishment between source and destination. After the timers' expiration, the

intermediate node forwards the RREQ, carrying the intermediate node's trust value to other neighbors, as shown in Figure 5.2.

The timer is used to examine the same RREQ packets that arrive at different times to the intermediate node and then the one with the highest trust value is forwarded. This procedure, to select the best path using trustworthy nodes, minimizes the amount of overhead control packets flooded throughout the network and reduces the probability of network traffic congestion. Figure 5.3 shows the proposed fuzzy flowchart [56].

The fuzzy based modification algorithm started by acquiring the intermediate node's speed, remaining energy and hop count value. Then, the timer is initiated when the new RREQ packet is received and creates a reverse route table; otherwise the node compares the stored trust value with the trust value carried by RREQ packet. If the stored value is smaller than received one then the reverse routing table updated with RREQ packet information. The intermediate node checks if it is the destination or if it has a fresh route to destination. Depending on intermediate node's decision, it sends a route reply RREP packet to initiator source node using unicast form path or it broadcasts the RREQ packet to neighbor nodes again. Fuzzy AODV source code modifications changes appear in Appendix C.

5.3.1 Fuzzy AODV Algorithm Steps

The following steps explain the route requesting steps in the proposed Fuzzy AODV algorithm:

1- Receive REEQ
2- Calculate Trust Value
3- IF (new RREQ) THEN
Set TIMER for new RREQ
Create reverse route table entry
Else
IF (node trust value improved) THEN
Update reverse route table entry
ELSE
Discard RREQ
4- WHILE (TIMER not expired)
GO TO step 1
5- IF (I am a destination node or have new route to destination) THEN
Send RREP
ELSE
Broadcast RREQ
6- END

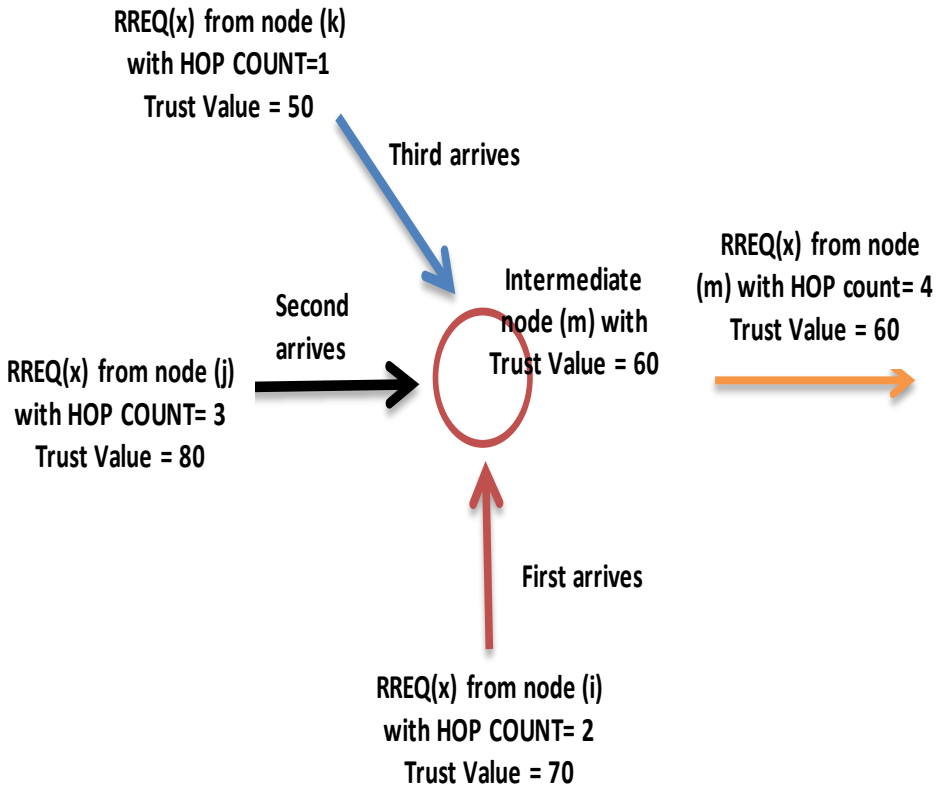


Figure 5.2: Intermediate node RREQ broadcasting in proposed Fuzzy AODV

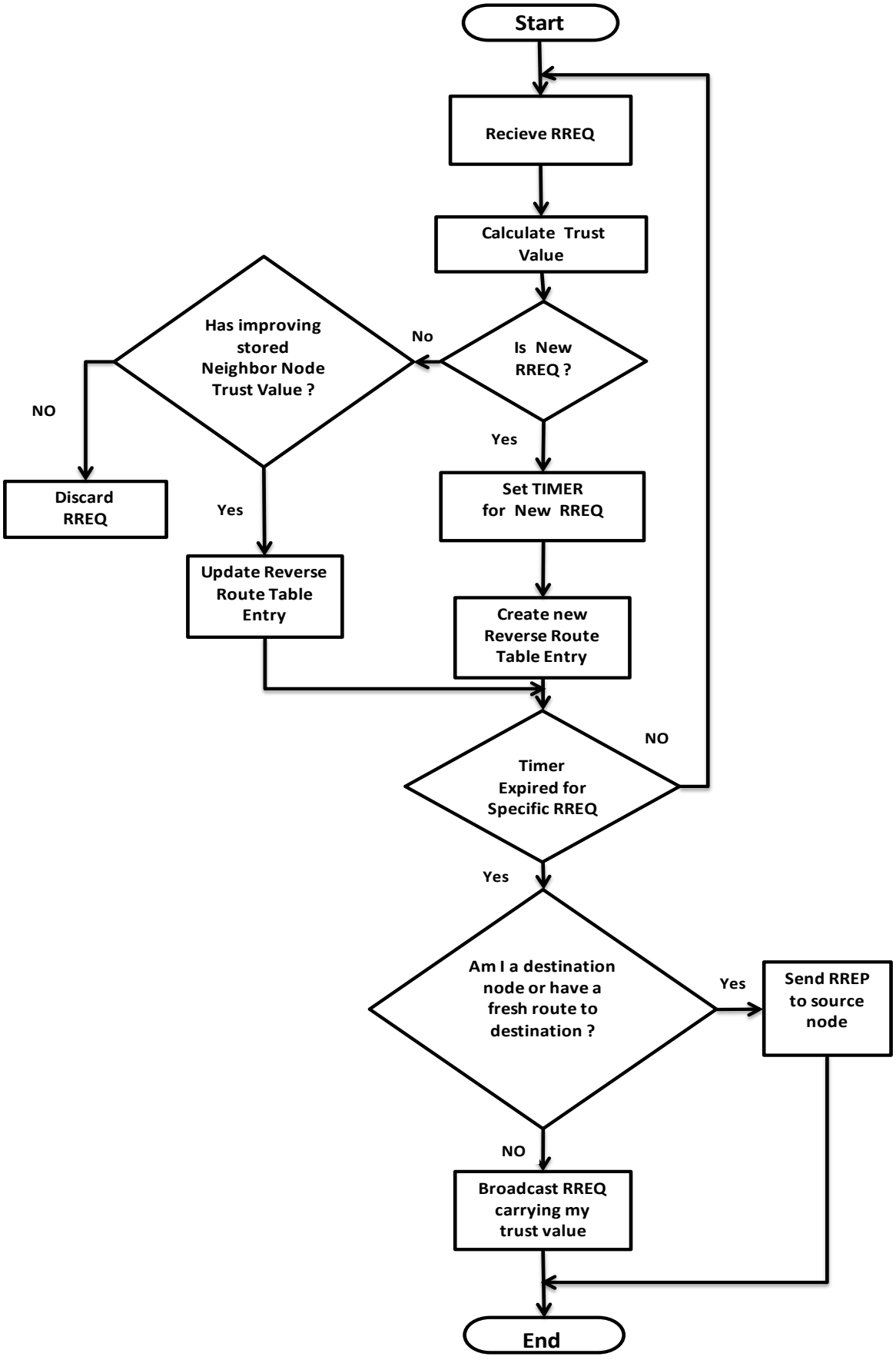


Figure 5.3: Flowchart of route requesting for intermediate node in the proposed Fuzzy AODV algorithm

5.3.2 Fuzzy Based Trust Value Computations

Computational intelligence techniques have been extensively used in various fields of engineering research and control engineering and provides very promising approaches in computer communication routing algorithms [57], [58]. The basic fuzzy system shown in Figure 5.4 is suited for decision making techniques. A fuzzy logic system describes the relationship between crisp inputs and output variables with the help of IF-THEN based rules provided by the fuzzy system designer. A fuzzy system consists of three main parts: Fuzzification, Defuzzification, and a fuzzy inference engine with IF-THEN based rules. Fuzzification is responsible for representing decisive input variables in terms of fuzzy set membership functions. Defuzzification converts the fuzzy output to decisive values using a mathematical formula, while the inference engine calculates the fuzzy output depending on the IF-THEN based rules as provided in Table 5.6.

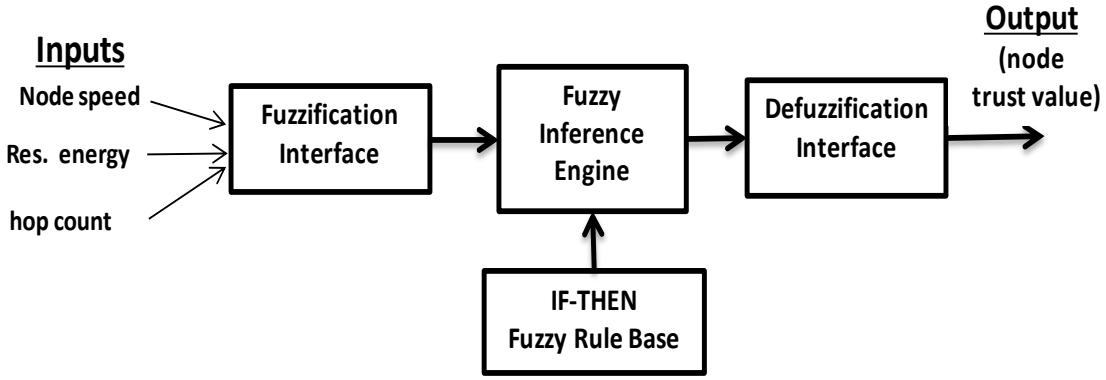
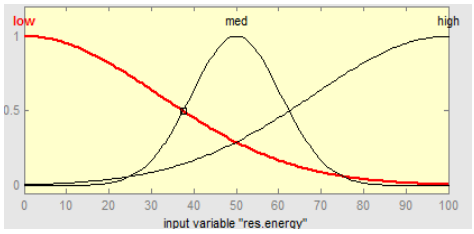


Figure 5.4: Block diagram of fuzzy logic system

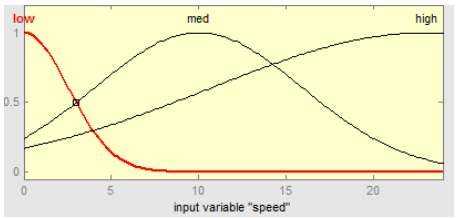
Because of the correlation between MANET parameters, which have a range of values, the fuzzy logic system describes the effects of the different parameter interactions. Hence, to develop a Fuzzy Inference System, the input and output variables should be defined as membership functions. Fuzzy rules (IF-THEN) that

connect the input memberships with the output membership are then suggested [59]. The membership function is a graphical interpretation of the input and output linguistic variables. The inputs, in our case, are node residual energy, node speed, and hop count values and the output represent the node trust value (node quality).

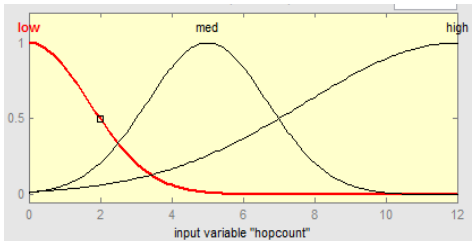
There are various representations of membership functions. Most popular member functions used are: piecewise linear, trapezoidal, triangular, and Gaussian membership functions. Table 5.4 shows a computational comparison results for different crisp input values for IF-THEN based rules shown in Table 5.6, and by using different types of membership functions. The computation results of the nodes trust values shown in Table 5.4, by applying Gaussian membership functions, shown in Figure 5.5, and triangular-trapezoidal membership functions, shown in Figure 5.6.



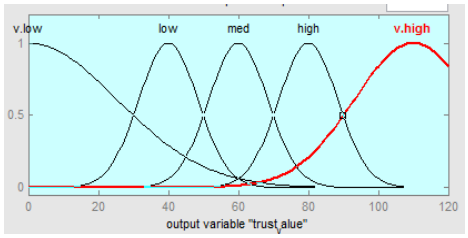
(a) Residual energy



(b) Node speed



(c) Hop count



(d) Trust value

Figure 5.5: Gaussian membership functions used to calculate node trust value

The computations are achieved using MATLAB 7.6.0 (R2008a) package under Window 7 Professional, processor of Intel Core i7, 2.4 GHz and 64 bits Operating system.

Table 5.4: Node trust values comparison using different membership functions

Residual Energy (%)	**Node Speed (m/sec)**	**Hop Count**	**Trust value (Triangular-Trapezoid membership)**	**Trust value (Gaussian membership)**
10	1	1	57.1	53.5
10	1	5	57.2	52.6
10	1	10	37	32.7
10	10	1	40.4	40
10	10	5	40.3	39.6
10	10	10	17.9	19.3
10	20	1	40.4	40
10	20	5	19.3	21.2
10	20	10	18.3	19.6
50	1	1	77.7	74.3
50	1	5	77.9	73.3
50	1	10	73.9	66.1
50	10	1	60.1	60.4
50	10	5	60.1	59.8
50	10	10	45.3	47
50	20	1	60.1	60.4
50	20	5	45.3	47.5
50	20	10	45.3	47.5
90	1	1	96.9	98.1
90	1	5	96.9	98.2
90	1	10	96.9	98.2
90	10	1	80.8	79.7
90	10	5	80.8	79.7
90	10	10	80.8	79.7
90	20	1	80.8	79.7
90	20	5	60.1	65.6
90	20	10	60.1	65.6

Since the results as demonstrated in Table 5.4 are showing very similar trust values for different membership functions via checking only the trust values, it is not

possible to decide which one should be applied. Due to their simple structures and linear expressions, triangular - trapezoid membership functions are widely used in fuzzy controller theory and applications and finds different ranges of interest in theoretical researches and industrial fields [60], [61], [62].

In our work, triangular - trapezoid membership functions applied because they are extensively used in real time operations. Also, the triangular - trapezoid functions are achieved with simple formulas and provide computational efficiency that we are needed for our node trust value computations [63]. High computational complexity is an important issue and needed to avoid in order to not adding extra delays for intermediate nodes reply. Triangles and trapezoid membership functions, formulated by equations (1) and (2), are used to describe the input and output membership degrees of the input and output variables of FIS as shown in Figure 5.6.

Node residual energy parameter, which directly affects the lifetime of a MANET, has an important influence on the node electromagnetic communication abilities, packet transmission, reception, and internal computing processes [64]. Hence, it is treated as a key input variable in the fuzzy node trust value calculation. The node speed parameter also has a considerable effect on route stability; when the selected node moves rapidly out of the communication range of the neighbor nodes, the links are broken. Hence, nodes with the highest speed increase the probability of a route failure which in turn increases the overhead control packets retransmission for route discovery process. The third parameter of fuzzy input variable used is the number of hop count values included in the RREQ packet, which represents the route length. Generally, the route with the minimum hops is the best route if all nodes participating in the established shortest route have maximum residual energy and

low speed. So, hop count parameter has the least significant effect on the output node trust value.

Triangle membership function

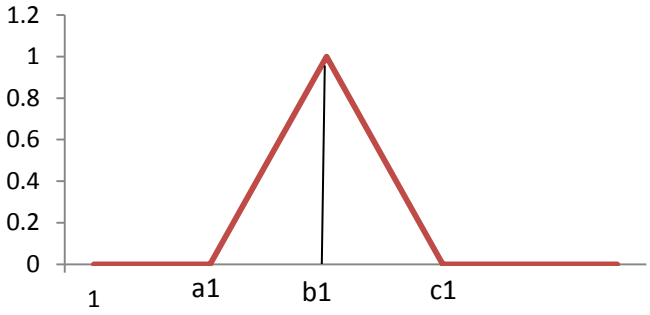


Figure 5.6: Triangular membership function

The triangular membership function defined as**:**

$$\mu_{A1}(x) = \begin{cases} 0 & x \le a1 \\ \frac{x - a1}{b1 - a1} & a1 \le x \le b1 \\ \frac{c1 - x}{c1 - b1} & b1 \le x \le c1 \\ 0 & x \ge c1 \end{cases} \quad (1)$$

Trapezoid membership function

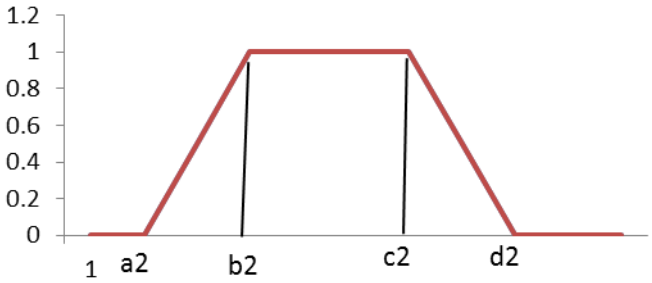
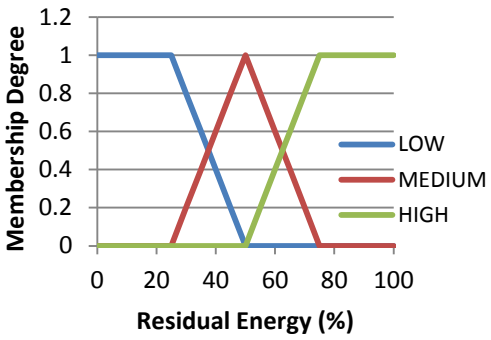


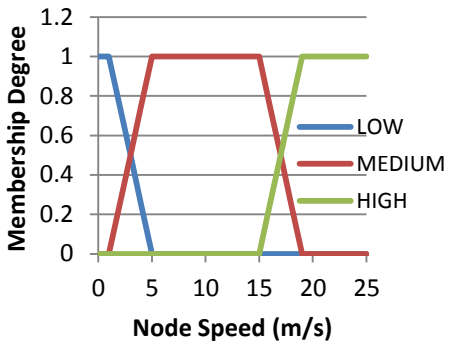
Figure 5.7: Trapezoid membership function

The trapezoid membership function defined as:

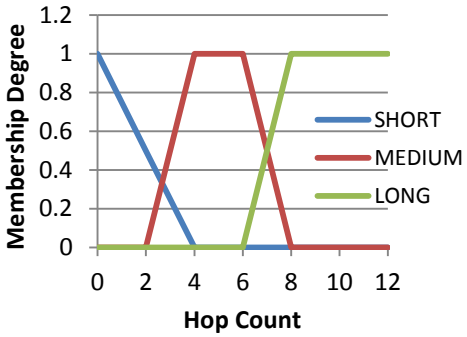
$$\mu A2(x) = \begin{cases} 0 & x \le a2 \\ \frac{x - a2}{b2 - a2} & a2 \le x \le b2 \\ 1 & b2 \le x \le c2 \\ \frac{d2 - x}{d2 - c2} & c2 \le x \le d2 \\ 0 & x \ge d2 \end{cases} \quad (2)$$



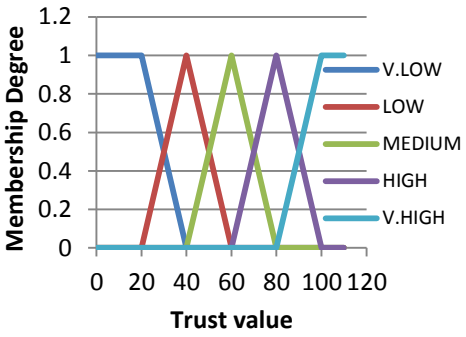
(a) Membership function of residual energy input



(b) Membership function of node speed input



(c) Membership function of the hop count input



(d) Output membership function of node trust value

Figure 5.8: Fuzzy membership functions used for node trust calculation

5.3.3 Operation of Fuzzy Logic Algorithm

The description of the fuzzy logic algorithm can be divided into four basic steps of fuzzification, IF-THEN rule evaluation, outputs aggregation, and defuzzification to calculate the crisp value. These steps are described as follows:

Step 1: Fuzzification of input crisp parameter values

The input parameters, in our case, are node residual energy, node speed, and number of hop count is defined by their membership functions as shown in Figure 5.6. Depending on the three input crisp values, we can find the membership degree of each input by intersecting the input value with the membership function.

Step 2: Evaluation of IF-THEN rules

The membership degrees found in step 1 are fed to IF-THEN based rules to determine the output fuzzy set. The AND operator is used to select the minimum membership values out of the three input membership values.

Step 3: Aggregation of outputs

In this step, the system collects, in the union form, all outputs that results from applying the IF-THEN rules, then apply OR operator to these outputs to select the maximum evaluating values to construct a new aggregate fuzzy set.

Step 4: Defuzzification process

The centroid method (center of gravity) is applied to the new aggregate function obtained in the step 3 to calculate the node trust value by using (3).

Defuzzification method is a mathematical approach to extract the crisp output value from the fuzzy aggregation output representation. There are various defuzzification methods that can be used to find the crisp value from output inference system, which have different conflict resolution schemes , such as First of Maxima (FOM), Last of Maxima (LOM), Mean of Maxima (MOM), Centroid method (also called Center of Gravity (COG)) and weighted average method [65]. Max and Mean of Max membership methods are limited to peaked output membership functions. Usually, the aggregation output memberships have multiple peaks rather than a single peak point. Weighted average method defuzzification is one of the frequently used ones in fuzzy system applications due to its efficient computation schemes. But, the disadvantage of this method is the restriction to symmetrical membership output functions [66].

Table 5.5 shows the comparison results of node trust value calculated by applying COG, LOM, and MOM defuzzification methods for our FIS and crisp inputs given in

the table. It is noted that the COG has a better output resolution for each distinct inputs compared to the LOM and MOM defuzzification methods. Using random input values with the fixed simulation parameters; wireless mac layer protocol (IEEE802.11), simulation area (900mx900m), transmission range (250m), mobility model (RWP), simulation time (300sec), application (FTP), size of packets (512bytes/sec) and interface queue size(50), it is observed that COG has the highest precision on outputs in overall. As a sample, as demonstrated in Table 5.5, for inputs; residual energy, node speed and hop count, where the values are 2, 2, 1 and 8, 2, 2 respectively, MOM returns the same output value 60, for inputs 8, 2, 2 and 10, 4, 2 respectively, LOM returns the same output value 49.2 however COG returns different outputs which in short means that methods MOM and LOM are not sensitive to the inputs as much as COG method.

Table 5.5: Comparison of node trust value using different defuzzification methods

Inputs			**Outputs**		
Residual Energy (%)	**Node Speed (m/sec)**	**Hop Count**	**Node trust computation using COG defuzzification**	**Node trust computation using MOM defuzzification**	**Node trust computation using LOM defuzzification**
2	2	1	54.2	60	64.8
8	2	2	53.1	60	49.2
10	4	2	48.9	39.6	49.2
6	12	6	40	39.6	39.6
12	20	3	23.9	15	30
15	3	12	27.3	24.6	49.2
11	7	10	15.2	9.6	19.2
18	5	6	40	39.6	39.6
22	2	6	54.2	60	64.8
26	14	3	41.5	39.6	49.2
34	2	2	61.9	60	69.6
37	8	3	49.1	39.6	49.2
42	18	10	31.5	39.6	45.6
44	1	1	74.4	79.8	84
47	6	8	36.5	40.2	42
51	12	4	61.1	60	60
57	3	7	67.7	60	90
62	15	3	69.8	60	69.6

66	4	2	79.2	80.4	90
72	2	4	95	100	100
74	12	8	77.1	79.8	80.4
78	5	3	80	80.4	90
82	3	6	92	95 4	100
85	2	6	98.5	100	100
88	6	3	80	80.4	90
89	16	5	74.2	79.8	84
90	4	2	88.1	80.4	90
92	12	6	80	80.4	80.4
98	5	7	96.1	80.4	90
100	2	4	98.5	100	100
99	8	6	80	80.4	80.4
67	13	3	72	80.4	90
88	5	8	80	80.4	80.4
54	6	6	64	60	62.4
43	4	3	59.4	60	69.6
32	6	4	46.3	40.2	45.6
28	9	2	44	39.5	49.2

The Centroid defuzzification is adopted in our proposed model because it is the most commonly used one and is very accurate and has more consistency in results. Also, this scheme represents the most prevalent and physical appealing of all the defuzzification schemes [67], [68], [69]. The mathematical expression of centroid defuzzification method symbolized in (3).

$$Center\ Of\ Gravity\ (COG) = \frac{\int \mu_A(x) * x\ dx}{\int \mu_A(x)\ dx} \quad (3)$$

Here, $\mu_A(x)$ represents the weight of the output membership function defined in (1) and (2), x denotes the centroid of each output membership function, and COG denotes the crisp value of the defuzzifier output [70].

5.3.4 Fuzzy IF-THEN Based Rules

Fuzzy-based rules map the input and output membership functions. The fuzzy inference engine is based on fuzzy IF-THEN based rules, which are ultimately written by a professional designer in the related field. The rules of the fuzzy based

system hold at most ($n \times m \times k$) IF-THEN rules, where n, m, and k are the numbers of membership functions characterized by the input variables. These memberships are connected using special fuzzy logic operators. In our case, (AND operator used (minimum (x, y, z))), 27 rules for our fuzzy inference engine, as shown in Table 5.6. Besides being differences at the output membership functions and the input parameters, 27 rules for fuzzy interference engine has been used in various studies also [71] [72] [73] [74] [75].

For example, as shown in Table 5.6, IF the node residual energy is HIGH AND node speed is LOW AND hop count is SHORT, THEN the node trust value is VERY HIGH. This means that this node is a trusted node (highly qualified) to be a part of a stable route. In contrast, IF the node residual energy is LOW AND node speed is HIGH AND the hop count is LONG, THEN the node trust value is VERY LOW. This means that this node is not a qualified node and it could cause established routes to fail if it is used. The numerical samples of the fuzzy system computation results shown in Table 5.7.

Appendix F shows the comparison results of using different numbers of membership functions used to calculate the node trust value and the effect of using two, three and four membership functions on the performance of proposed Fuzzy AODV algorithm. Appendix G explains the train and test phases of the fuzzy logic system.

Table 5.6: Fuzzy based rules set

Inputs			Output	Inputs			Output	Inputs			Output
Residual Energy	Node Speed	Hop Count	Trust Value	Residual Energy	Node Speed	Hop Count	Trust Value	Residual Energy	Node Speed	Hop Count	Trust Value
Low	Low	short	Med	Med	Low	short	High	High	Low	short	v. High
Low	Low	Med	Med	Med	Low	Med	High	High	Low	Med	v. High
Low	Low	Long	Low	Med	Low	Long	High	High	Low	Long	v. High
Low	Med	short	Low	Med	Med	short	Med.	High	Med	short	High
Low	Med	Med	Low	Med	Med	Med	Med.	High	Med	Med	High
Low	Med	Long	v. Low	Med	Med	Long	Low	High	Med	Long	High
Low	High	short	Low	Med	High	short	Med.	High	High	short	High
Low	High	Med	v. Low	Med	High	Med	Low	High	High	Med	Med
Low	High	Long	v. Low	Med	High	Long	Low	High	High	Long	Med

Table 5.7: Numeric samples of fuzzy system calculations

Inputs			Output	Inputs			Output	Inputs			Output
Residual Energy	Node Speed	Hop Count	Trust Value	Residual Energy	Node Speed	Hop Count	Trust Value	Residual Energy	Node Speed	Hop Count	Trust Value
< 25	< 1	< 1	45	50	< 1	< 1	65	> 75	< 1	< 1	85
< 25	< 1	]4, 6[	45	50	< 1	]4, 6[	65	> 75	< 1	]4, 6[	85
< 25	< 1	> 8	30	50	< 1	> 8	65	> 75	< 1	> 8	85
< 25	]5, 15[	< 1	30	50	]5, 15[	< 1	45	> 75	]5, 15[	< 1	65
< 25	]5, 15[	]4, 6[	30	50	]5, 15[	]4, 6[	45	> 75	]5, 15[	]4, 6[	65
< 25	]5, 15[	> 8	9.75	50	]5, 15[	> 8	30	> 75	]5, 15[	> 8	65
< 25	> 19	< 1	30	50	> 19	< 1	45	> 75	> 19	< 1	65
< 25	> 19	]4, 6[	9.75	50	> 19	]4, 6[	30	> 75	> 19	]4, 6[	45
< 25	> 19	> 8	9.75	50	> 19	> 8	30	> 75	> 19	> 8	45

Note that the mathematical operators : < less than, > greater than,] a, b [where {x ∈ R:a< x <b

5.3.5 Complexity Analysis of Fuzzy Inference System

Complexity analysis of algorithm is one of the most complicated branch of mathematic topics that deals with the number of steps (time consuming) and memory space needed for solving problems under the worst case condition (maximum number of steps required to solve the problem). An algorithm may be defined as a number of procedural and computational steps used to solve a specific problem. The

algorithm usually consists of three parts; inputs, outputs, and a sequence of finite procedural logic (number of steps) that transform the inputs to the outputs in a finite amount of time and memory spaces. There are two kinds of algorithms ; informal algorithms (which are a well-defined computational procedure that takes some values or set of values as input and produce some values or set of values as output). and the formal algorithms (which are the corrected solution for any problem could be found by applying a finite sequence of unambiguous steps and terminated after a finite amount of memory and time for all possible inputs) [76].

In our work, the major modification was adding fuzzy logic inference steps to modify the original AODV routing protocol. These commands were used to calculate the trust value of nodes, so the complexity analysis of the modified AODV routing protocol can be done by analyzing the complexity of fuzzy logic system.

A fuzzy logic system basically consists of three parts; fuzzification part, inference engine part, and defuzzification part. Each part can be implemented using several techniques, so there exists different characteristics of fuzzy logic algorithms. The use of triangular fuzzification, minimum inference engine base rules, and center of gravity in our Fuzzy Inference System is one of such example. Table5.8 abbreviates the parameters and symbols used in our fuzzy complexity analysis.

Table 5.8: Fuzzy logic parameter abbreviations

Symbol	**Description**
N_i	Number of crisp inputs
N_{iF}	Number of input fuzzy sets
N_{iD}	Number of membership function discretization
N_R	Number of inference rules
N_o	Number of crisp outputs
N_{oF}	Number of output fuzzy sets
N_{oD}	Number of membership function discretization

5.3.5.1 Number of Operations of General Fuzzy Logic Algorithm

General Fuzzy Inference System has computation time related to the fuzzy parts operations. Each part of fuzzy logic system has a number of computation steps and it can be described as followings.

a) **Fuzzification Part**

Fuzzification part is used to transform the crisp input variable into fuzzy memberships function sets. One of the widely transformation method used is triangular membership fuzzy sets. By considering triangular membership functions and to estimate the number of operations for fuzzification process, equation (4) can be used to compute the number of operations [77]:

No. of fuzzification operations for triangular shaped = $(59+31N_{iF})\ N_i$ (4)

Where N_{iF} is assumed to be equal for all input variables.

When a nonspecific membership function shaped fuzzy set used, the number of operations required for fuzzification process will be as the following Equation:

No. of fuzzification operations of nonspecific shape = $(70N_{iF}+29N_{iF}N_{iD}+8)\ N_i$ (5)

b) Inference Part

The inference engine represents the main part of fuzzy logic system. There are many schemes used to implement inference base rule processes. So, different number of operations required for each scheme. The number of operation required for minimum inference process will be as the following equation:

No. of operation of minimum inference = $(63N_{oD}+37N_i+19)\ N_R+6$ (6)

c) Defuzzification Part

Defuzzification part is responsible from extracting the final crisp output value of fuzzy logic system based on the inference result of each rule. Center of Gravity method is carried out by calculating the center of area of the resulting fuzzy sets

aggregated from the inference base rule stage. The number of operation required is given by the following equation:

No. of operation of COG defuzzification = $(39N_{oD}+5)\ N_R+15$ (7)

Table 5.9, estimates the number of operations required for each part of fuzzy inference logic system. Table 5.10 describes the total number of basic operations required for Fuzzy Inference System [77].

Table 5.9: Number of operations required for FIS

Fuzzy system part	**Method**	**Number of operation**
Fuzzification	Triangular membership Non-specific	$(59+31\ N_{iF})$ Ni $(70\ N_{iF} + 29\ N_{iF}\ N_{iD} + 8)\ N_i$
Inference engine	Minimum inference Product inference	$(63\ N_{oD} + 37\ N_i +19)\ N_R + 6$ $(88\ N_{oD} + 37\ N_i +20)\ N_R + 6$
Defuzzification	Mean of maximum Center of Gravity	$(25\ N_{oD} + 5\ )\ N_R +15$ $(39\ N_{oD} +5)\ N_R +15$

Table 5.10: Number of basic operations required for each Fuzzy Inference operation based on [77]

Method	**Addition**	**Subtraction**	**Comparison**	**Multiplication**	**Division**
Triangular fuzzy set	$(4N_{iF}$ $+6)N_i$		$3N_{iF}\ N_i$	$(2N_{iF}\ +2)N_i$	
Non-specificc fuzzy set	$(9N_{iF}$ $+4N_{iF}$ $N_{iD})N_i$	N_{iF} Ni	$(N_{iF}$ $+3N_{iF}$ $N_{iD})N_i$	$(4N_{iF}$ $+2N_{iF}$ $Ni_D)$Ni	$N_{iF}\ N_i$
Minimum inference	$(9N_{oD}$ $+6N_i$ $+3)N_R$	$4N_{oD}\ N_R$	$(2N_{oD}$ $+N_i$ $+1)N_R$		
COG	$(5N_{oD}$ $+1)N_R$	$2N_{oD}\ N_R$	$(N_{oD}$ $+1)\ N_R$	$2N_{oD}\ N_R$	1

Table 5.11 shows the FIS's computational costs for different crisp input values. Three sets of triangular membership functions were used for each input. Inputs and output memberships discretized set to 10. Center of Gravity used as the

defuzzification method. Also, one output of five triangular membership functions was used.

Table 5.11: Computational cost of FIS for different crisp inputs

No. of crisp inputs	No. of rules	Total number of operations required	Computational cost
1	3	3416	$5.86 * 10^6$
2	9	10387	$5.39 * 10^7$
3	27	31662	$5.00 * 10^8$
4	81	97181	$4.67 * 10^9$
5	241	296970	$4.30 * 10^{10}$

In Figure 5.7, computational cost of FIS with different number of crisp inputs is shown. Figure 5.8 presents the total number of operations required for FIS versus different number of crisp inputs. Cubic spline interpolation method, which has been fitted with Matlab Curve Fitting tool, with zero error expresses the correlation between the number of crisp inputs vs. number of required operations and the number of crisp inputs vs. computational cost.

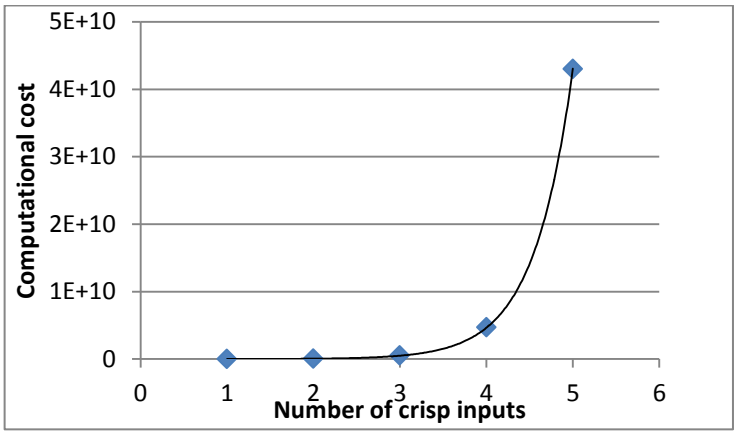


Figure 5.9: Computational cost of FIS with different crisp inputs

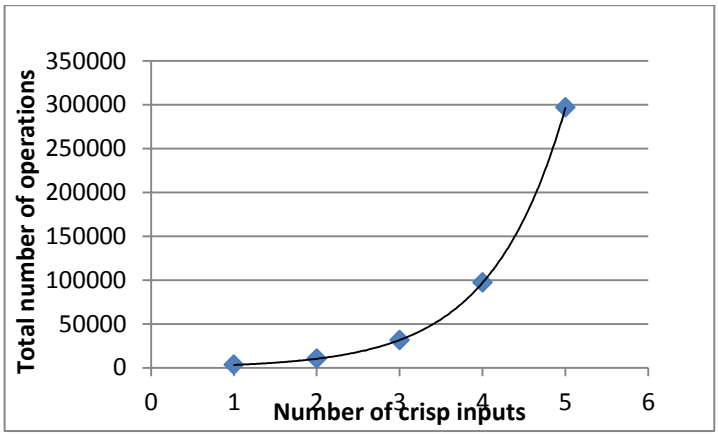


Figure 5.10: Total number of operations of FIS with different crisp inputs

Table 5.12, Shows computational costs of the FIS of three crisp inputs with different number of triangular memberships function sets. The numbers of discretization for input and outputs memberships are set to 10 and Center of Gravity method used for defuzzification. Also one output of 5 triangular membership functions was used.

Table 5.12: Computational cost of FIS for different number of membership functions

No. of membership function set	**No. of rules**	**Total number of operations required**	**Computational cost**
1	1	1446	$0.88 * 10^{6}$
2	8	9624	$4.44 * 10^{7}$
3	27	31662	$5.00 * 10^{8}$
4	64	74490	$2.80 * 10^{9}$
5	125	145038	$1.07 * 10^{10}$

Figure 5.9, shows the computational cost of FIS with fixed number of crisp inputs (set to 3 inputs) and different number of input membership function sets. In Figure 5.10, the total number of operations required for FIS with different inputs membership functions presented. Cubic spline interpolation method that has been fitted with Matlab Curve Fitting tool, with zero error expresses the correlation

between the number of membership function sets vs. number of required operations and the number of membership function sets vs. computational cost.

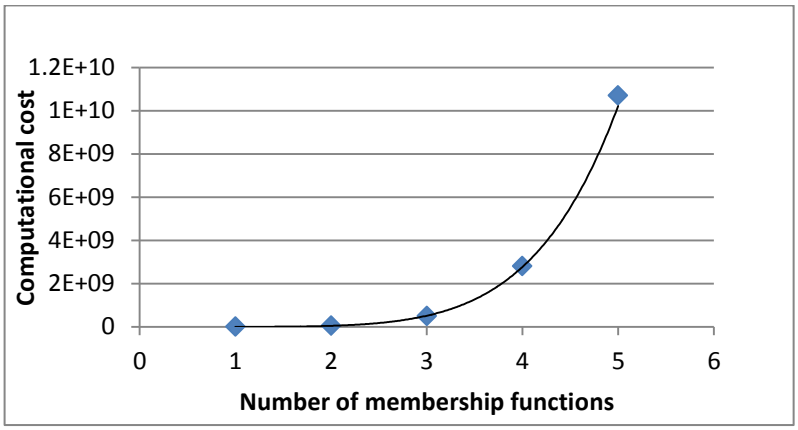


Figure 5.11: Computational cost of FIS with different input membership functions

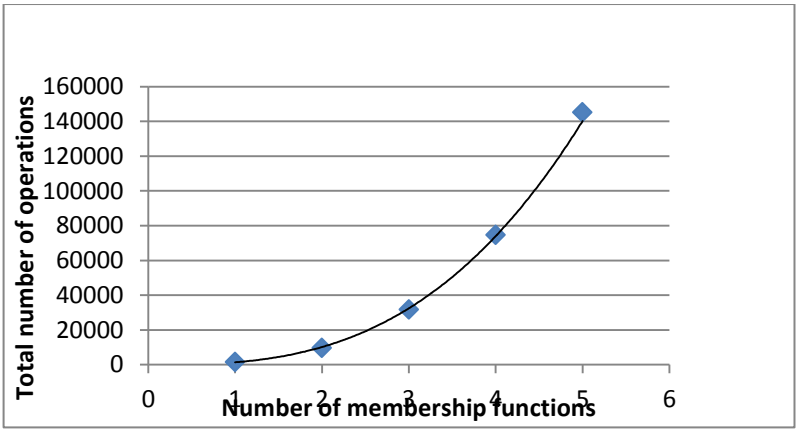


Figure 5.12: Total number of operations of FIS with different input membership functions

In Table 5.13, computational costs of the FIS for different fuzzy logic inference inputs with varying numbers of memberships function sets demonstrated. Number of discretization of inputs and output memberships is set to 10 and Center of Gravity used as the defuzzification method; also one output of five triangular membership functions was adopted.

Table 5.13: Computational cost of FIS for different number of inputs and membership functions

Number of Crisp inputs	Number of membership function sets	Computational cost
1	1	685038
	2	$2.63 * 10^6$
	3	$5.86 * 10^6$
	4	$1.03 * 10^7$
	5	$1.61 * 10^7$
2	1	763985
	2	$1.08 * 10^7$
	3	$5.39 * 10^7$
	4	$1.69 * 10^8$
	5	$4.13 * 10^8$
3	1	880116
	2	$4.44 * 10^7$
	3	$5.00 * 10^8$
	4	$2.80 * 10^9$
	5	$1.07 * 10^{10}$
4	1	$1.03 * 10^6$
	2	$4.64 * 10^7$
	3	$5.20 * 10^8$
	4	$2.92 * 10^9$
	5	$1.11 * 10^{10}$
5	1	$1.22 * 10^6$
	2	$4.85 * 10^7$
	3	$5.42 * 10^8$
	4	$3.03 * 10^9$
	5	$1.15 * 10^{10}$

Figure F3 in appendix F demonstrates the effect of using two, three and four membership functions on the performance of the proposed Fuzzy AODV algorithm. As illustrated in Figure F3(a) when node speed increases packet delivery ratio of Fuzzy AODV with four membership functions and three membership functions converges to each other and shows an increasing percentage where the packet delivery ratio of Fuzzy AODV with two membership functions decreases.

Figure F3(b) presents the throughput of Fuzzy AODV under various node speeds. With three membership functions and four membership functions Fuzzy AODV shows similar and increasing behavior while Fuzzy AODV with two membership functions has lower throughput under increasing node speeds. Figure F3(c) demonstrates the average routing load under different node speeds. Fuzzy AODV with two membership function has the worst routing load, while three and four membership Fuzzy AODV shows almost the same routing load. Four membership function Fuzzy AODV has highest end to end delay when compared to Fuzzy AODV with two and three membership functions as presented in Figure F3(d).

According to the analysis demonstrated in Figure F3, Fuzzy AODV with Four and three membership functions shows almost the same performance where the Fuzzy AODV with two membership function shows poor performance compared to other versions of Fuzzy AODV. Complexity analysis shows that the computational cost of Fuzzy AODV with three membership functions is better when compared to Fuzzy AODV with four membership functions. We can conclude from the analysis made Fuzzy AODV with three membership functions will be the best to choose among the three membership function comparisons.

Chapter 6

SIMULATION ENVIRONMENTS

This chapter presents network simulation environments for AODV and the proposed Fuzzy AODV routing protocols, simulation tools and network parameters.

6.1 Simulation Model

Discrete event network simulator NS-2 package version NS-2.35 has been used to evaluate the performances of AODV and Fuzzy AODV routing protocols [78]. NS-2 is one of the most popular open source object-oriented software network simulators. It has a large library that supports various aspects of computer network simulations environments such as routing protocols, mobility models, network layers and network configurations.

NS-2 simulator includes different categories of wireless and wired routing protocols. C++ programming language represents the core language used in NS-2 simulator. Tool Command Language (TCL) is the interactive user interface language for NS-2 which helps the users to modify the configuration of the network parameters without the need of recompilation.

6.1.1 Network Animator

Network animator (NAM) is a visualization tool that animates the nodes movement patterns. It helps researchers to build their simulation terrain network and traces the packet relaying within the network area. NS-2 cooperates with the visual output network animator as shown in Figure 6.1. It supports the users via satisfying the

opportunity to track the nodes positions and packet activities like node movements, node transmission coverage boundary, packet relaying, packet queuing and dropping that one cannot see or feel it in the real world [79].

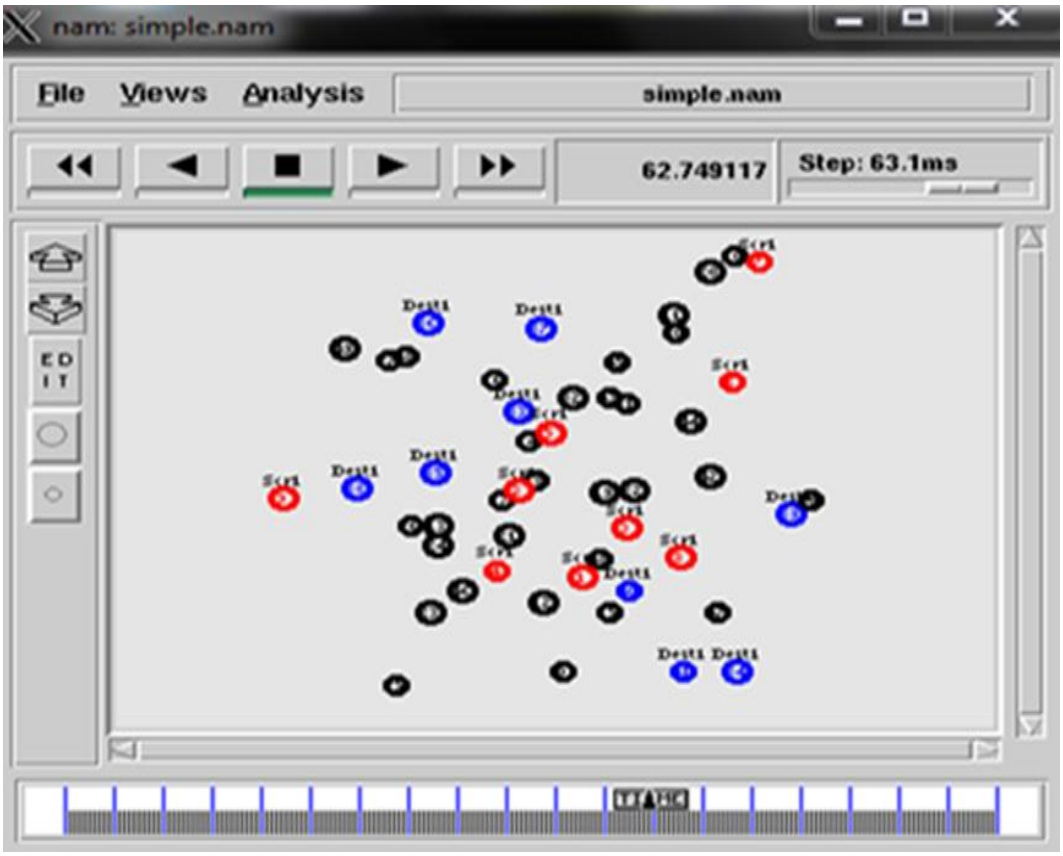


Figure 6.1: Snapshot of NS-2 NAM

6.1.2 Mobility Model

Mobility model is a set of the nodes movement pattern models that mimics the behavior of the node movements throughout the network area over the simulation time. This movement pattern shows node locations, velocities, and accelerations at different instants of times over the network. Generally, mobility pattern scenario influences the networks connectivity graph which in turn influences on the protocols performance. Mobility models attempt to mimic the mobile users' movements in real

world. So, for simulation purposes, it is important to select a proper mobility model for a specific application of MANETs routing protocol evaluation.

Complexity is a measure of the computational resources needed to produce traces of the simulation scenario. Complicated mobility models take a significant fraction of the total simulation time. Stochastic models depend primarily on random movements without imposing any constraints on the movement of the nodes. Classical examples of mobility models include the random waypoint model and random direction mobility model.

Random Waypoint Mobility (RWP) has a wide range of usability in MANETs routing protocol environments. It is flexible, and it appears to create realistic mobility patterns for the way of people might move in, for example, a museum or conference setting. Moreover, there are objects of different moving speeds, like vehicles as fast movement objects, others (pedestrians) are moving slower. Hence, there are heterogeneous objects speeds based on the kind of the nodes used.

RWP model is a simple stochastic model which the nodes select destinations (waypoints) and move towards them. Changes in directions and speeds may occur in reasonable time intervals. There are neither dependencies nor any other restriction modeled [80], [81]. In our simulation scenarios, RWP model has been chosen to mimic the nodes movement patterns in disaster areas, where the nodes, like firefighters, may move within the disaster area. The firefighters move from one location to other and stay temporarily close to these locations. Some of them may walk around locations randomly and sometimes they are moving independent of each other.

RWP became a benchmark mobility model to evaluate the performance of different MANET routing protocols. This model was widely used for simulation purposes because of being simple, easy to implement and analyze [82]. Also, RWP is fairly easy to diversify the mobility model parameters (i.e. nodes speed, pause time, number of nodes. etc.) which could be changed in NS-2 simulators [83]. In order to generate nodes mobility pattern scenario using NS-2, the following executable command under the NS-2 platform directory $NS2_HOME/indep_utlis/cmu-scen-gen/setdest have to be used:

/setdest [-n No. of nodes] [-p pause time] [-s max. speed][-t sim. time] [-x max.x] [-y max.y] > [output file].

The command shows the node mobility parameters that the user can manipulate, such as number of nodes, maximum node speed, pause time, simulation period and dimensions of network terrain.

6.1.3 Traffic Pattern Generation

Sometimes emergency operation cases require a rapid wireless communicating between mobile nodes. These operations usually need a trusted and reliable communication, where a data packet losing may cause many troubles to the users. In order to avoid the effects of packet loses on the routing protocol performance studies; a connection oriented communication scheme could be used. FTP application traffic model running over the connection oriented agent of Transmission Control Protocol (TCP) is adopted in our simulation studies. Constant Bit Rate (CBR) and TCP scripts are available in NS-2 which can be used to generate random pairs of traffic flow within the simulation scenario. This script of traffic generation is created and setup under directory ~ ns/ indep-ultis/cmu-scen-gen. It can be used to generate TCP and

CBR traffic connection between mobile nodes. The following command is used to generate traffic files linked with the TCL script to run it:

ns cbrgen.tcl [-type cbr | tcp] [-nn nodes] [-seed seed] [-mc connections] [-rate rate]

6.1.4 Simulation Data Trace File

Open source network simulator, NS-2 provides data trace files that record every discrete event occurred over the simulation period. Trace files usually need more processes to extract the useful information. Network performances can be derived from the trace files using some of statistical tools. Determining the relevant information from the trace file will help the users to get and calculate the network performances. There are several categories of trace files formats exist depending on the nodes nature and the network operation environments (such as fixed or mobile nodes, different simulation networks, and different routing protocols). Generally, different packet actions, queuing and different layers of network can be traced. All traced discrete events can be documented in a trace files. For that reason, trace file size increases exponentially with the number of nodes used in the simulated scenario, which will consume computer resources effectively (processor cycles and memory space). So, it preferred to select minimum parameters to trace. Table 6.1, shows the fields of the wireless trace file format used in this work.

Table 6.1: Wireless trace format fields used

Event Type	**Time Stamp**	**Source ID**	**Agent**	**Packet ID**	**Packet Type**	**Packet Size**

Several documents available to explain NS-2 trace file format fields [79]. Understanding trace file fields is very important to extract the useful data for specific

performance metric calculations and by using additional statistical programs as AWK, Gun plot, Perl etc. [79].

6.1.5 AWK Programming Language

AWK programming language is designed and created by Alfred Aho, Peter Weinberger, and Briab Kamighan in 1977. AWK is, basically, formed to be a data driven script language to process textual data by extracting data from text files. It includes set of actions that calculates string of specific data from a texture file. AWK is a good filter tool for processing data in rows and columns forms. It searches a particular data string and manipulates it by its own commands and function. It is conventional with the C programming language [84]. AWK programming language is used to extract and report the important data from NS-2 trace file that was generated at the end of the simulation scenario. From studying trace files carefully, one can understand the wireless computer network operations. Appendix D shows AWK script program written to evaluate different performance metrics used in this study.

6.2 Modification of AODV Simulation Code

AODV folder in the NS-2 base directory consists of eight C++ of source code files (aodv.cc, aodv.h, aodv_rtable.cc, aodv_rtable.h, aodv_packet.h, aodv_logs.cc, aodv_rqueue.cc, and aodv_rqueue.h). Implementing and adding our fuzzy algorithm requires modifying some functions in related AODV files. The files that have been modified are aodv (cc, h), packet.h, and rtable.h. For example, in order to define and fetch the node parameter values (node speed and residual energy), in sendrequest function of aodv.cc file, code modification has been applied as shown in Figure 6.2 below:

Modify send request function

```
----------------------------------------------------------------------------------------------------
void
AODV::sendRequest(nsaddr_t dst) {

iNode = (MobileNode *) (Node::get_node_by_address(index));
rq->v = iNode->speed();
rq->iEnergy = iNode->energy_model()->energy();
iEnergy = iNode->energy_model()->energy();
node_speed = iNode->speed();
double Resenergy = iEnergy;
double Velocity = node_speed;
```

Figure 6.2: Modifications of send request function

Also the node parameters updated in the aodv.h file as shown in Figure 6.3. :

Modify header AODV request structure

```
struct hdr_aodv_request {
      u_int8_t      rq_type;          // Packet Type
      u_int8_t      reserved[2];
      u_int8_t      rq_hop_count;   // Hop Count
      u_int32_t     rq_bcast_id;    // Broadcast ID

      nsaddr_t      rq_dst;           // Destination IP Address
      u_int32_t     rq_dst_seqno;   // Destination Sequence Number
      nsaddr_t      rq_src;           // Source IP Address
      u_int32_t     rq_src_seqno;   // Source Sequence Number

      double        rq_timestamp;   // when REQUEST sent;

      double        v;
      double        iEnergy;

      double        stability;
 ----------------------------------------------------------------------------------------------------
```

Figure 6.3: Modifications of header AODV request structure

Appendix B explains the complete code modification of the AODV routing protocol folder to implement fuzzy AODV algorithm.

6.3 Simulation Parameters

Simulation study considers a default wireless channel model and IEEE 802.11 physical layer. The network area size of 900 m x 900 m and 50 wireless mobile nodes randomly distributed across the simulated area with a maximum speed of 20m/s for all scenarios studied. All performance metric results presented are an average of 20 different simulation runs. Three different network parameters scenarios are studied and tested with a 20 different traffic mobility patterns. Maximum node speeds, number of nodes and pause times varied to perform the network performance evaluations. In order to isolate effects of packet losses on the protocols comparison, FTP was used as a traffic pattern. Nodes communicating randomly choses source-destination pairs for packets transportation. Table 6.2 summarizes the network parameter values used to create the simulation scenarios.

Table 6.2: Simulation scenarios

Scenario	**Network parameter values**
node speeds (m/s)	2, 6, 10, 12 (Human speed) 8, 16, 24 (Vehicle speed)
pause times (sec)	0, 30, 60, 90
number of nodes	10, 30, 50, 70, 90

Other default network parameter values for all simulation scenarios are listed in Table 6.3.

Table 6.3: Default simulation parameters for all scenarios

Parameter	Value
Network simulator	NS-2.35
Routing protocols	AODV, Fuzzy AODV
Wireless Mac Layer protocol	IEEE 802.11
Simulation area	900m x 900m
Transmission range	250 meter
Mobility model	RWP
Simulation time	300 sec
Interface queue size	50
Size of packet	512 bytes/packet
Application type	FTP

Chapter 7

SIMULATION RESULTS AND DISCUSSION

Performance simulation results of classical AODV and modified Fuzzy AODV routing protocols implemented using NS- 2 simulator. Version NS-2.35 under Cygwin package platform that emulates Linux environment for windows operating system. Intel Core i7, 2.4 GHz, 64 bits processor was used to run our simulation programs. Average 20 random mobility traffic scenarios were used for all performance results.

7.1 Performance Metrics

Routing protocols performances are evaluated using number of quantitative metrics. In this study, popular performance evaluation metrics for MANET routing protocol simulation has been used.

7.1.1 Average Throughput

It can be expressed as the amount of data packets arrived successfully at the final destination per unit of the simulation period time.

7.1.2 Packet Delivery Ratio (PDR)

It can be expressed as the ratio of the packets successfully arrived at the final destination nodes to transmitted packets by the source nodes.

7.1.3 Average Routing Load

It can be expressed as the total number of all overhead routing control packets sent from all nodes within the entire MANET network over the simulation period.

7.1.4 Average End to End Delay

It can be expressed as the average time that the data packets elapsed to transfer from the source nodes to the destination while considering all delays of queuing, propagation and buffering.

7.2 Performance Simulation Results

The following routing protocol simulation study has four different usage scenarios that use default simulation parameters described in Table 6.3. Each of these scenarios was simulated under different node parameters. Performance of AODV and Fuzzy AODV routing protocols are also subjected to two different traffic flows to measure the effects of the traffic pattern on the routing protocol performance.

7.2.1 Varying Node Speeds

In our simulation scenarios, we categorized the node speeds into two types according to their relative speeds a) Low speed scenarios; b) High speed scenarios. In low speed scenarios as a human walking average speed of 2 (m/sec) (human walking speeds usually ranging from 1.4 (m/sec) to 2.5 (m/sec)). And human running speeds ranges approximately from 2.3 (m/sec) to as much as 12 (m/sec). While the vehicle speeds or high speed scenarios may ranges from 30 km/h (8.3 m/sec) to 90 km/h (25 m/sec) [85]. In both protocols the simulation scenarios simulated under node density of 50 nodes and pause time set to 20 sec. Table 7.1 describes the numerical average simulation results of the two studied routing protocols in varying low node speed scenarios.

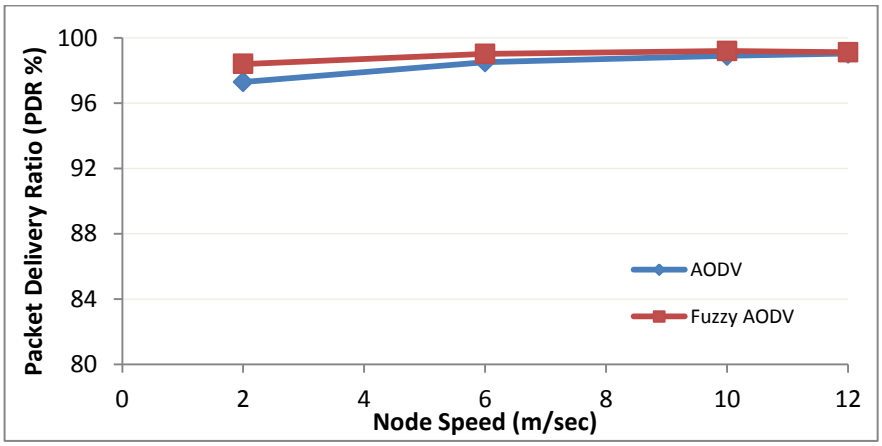
Table 7.1: Average results of AODV and Fuzzy AODV protocols for low node speeds (Human speed)

Metric	Protocol	Node speed (m/sec)			
		2	**6**	**10**	**12**
PDR (%)	AODV	97.295	98.51	98.9	9.04
	Fuzzy AODV	98.4	99.02	99.2	99.12
Average end to end delay (msec)	AODV	95.35	93.55	80.45	75.45
	Fuzzy AODV	87.825	86.88	75.445	72.69
Average routing load	AODV	9.62	13.56	13.85	13.9
	Fuzzy AODV	8	10.26	11.67	11.17
Average throughput (kbps)	AODV	212.8	189.8	207.6	229.1
	Fuzzy AODV	218.55	195.55	216	233.15

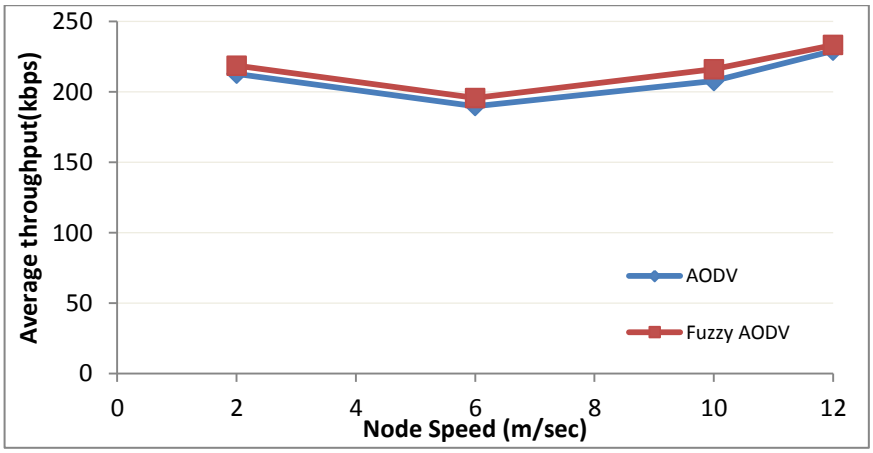
In low speed moving scenarios shown in Figures 7.1 (a), PDR metric increases slightly as the node speed increases in both AODV and Fuzzy AODV routing protocols studied. While the network throughput decreases as node speed increases up to the node speed approach to 6 m/sec. then the throughput increases gradually to maximum throughput value at node speed equal to 12m/sec as shown in Figure 7.1 (b). The increase of throughput does not necessarily indicate that the overall PDR percentage has improved. For that reason, the packet delivery ratio percentage calculated from the ratio of the number of data packets sent from sources to the number of data packets received by the destination nodes is an important performance metric, which defines the goodput of the network. The network throughput is expressed as the amount of data transferred over a period of time which

is related to the number of data sources send data irrespective of the transferred data loss through the network.

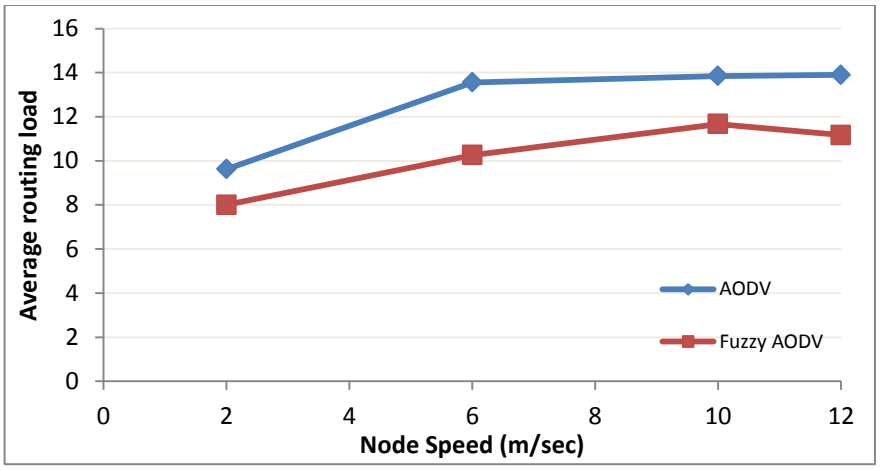
The simulation results reveal that Fuzzy AODV has the lowest routing load in both of node low speed and high speed cases compared with classic AODV routing protocol as shown in Figure 7.1 (c) and Figure 7.2 (c). Also the Fuzzy AODV performs better than classic AODV protocol in average end to end delay performance metrics in all varying node speeds as shown in Figure 7.1 (d) and Figure 7.2 (d). The probability of route failure of Fuzzy AODV is lower than the classical AODV with increasing the node speeds which indicates to the improving of fuzzy AODV scheme over classical AODV.



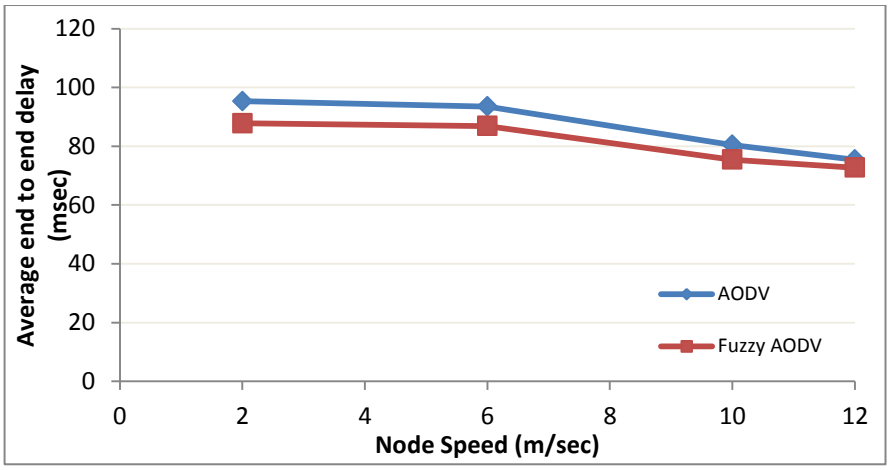
(a)



(b)



(c)



(d)

Figure 7.1: AODV and Fuzzy AODV performances vs. node speeds (Human speed) with 50 nodes and pause time of 20 sec

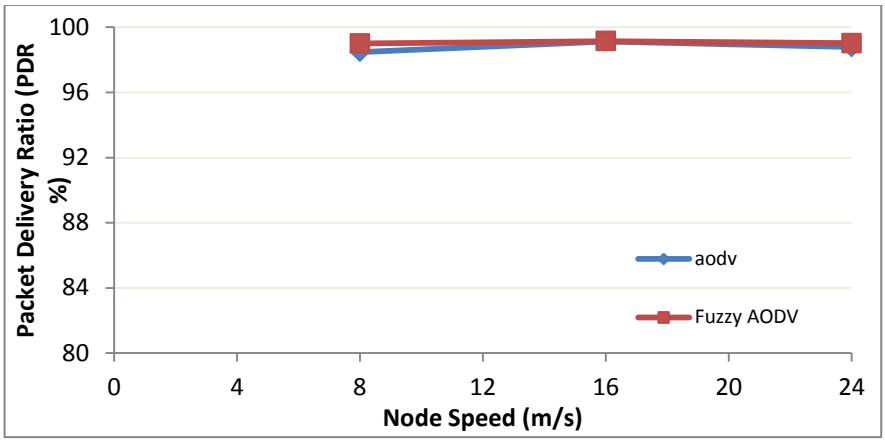
Table 7.2 describes the numerical average simulation results of the two studied routing protocols with high varying node speed scenario.

Table 7.2: Average results of AODV and Fuzzy AODV protocols for high node speed (Vehicles speed)

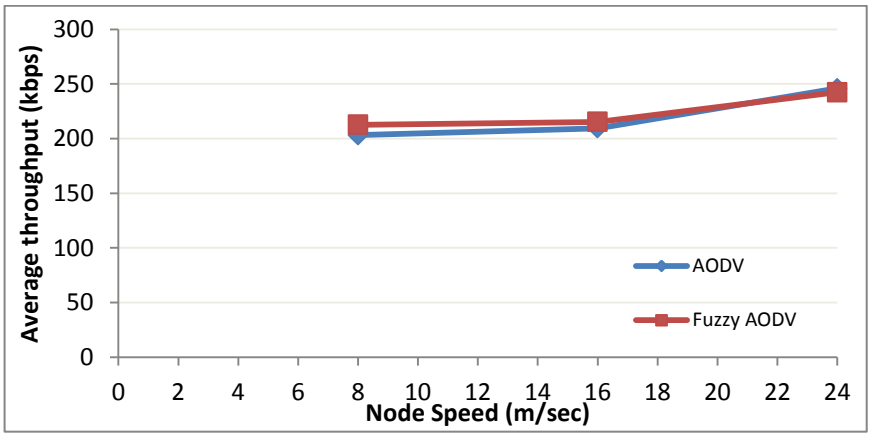
Metric	**Protocol**	**Node speed (m/sec)**		
		8	**16**	**24**
PDR (%)	AODV	98.48	99. 15	98.78
	Fuzzy AODV	99	99.15	99.02

Average end to end delay (msec)	AODV	87.05	78.35	71.3
	Fuzzy AODV	79.55	71.11	60.18
Average routing load	AODV	13.19	15.15	19.2
	Fuzzy AODV	10.66	12.67	16.7
Average throughput (kbps)	AODV	203.1	209.4	246
	Fuzzy AODV	212.7	215.3	242.4

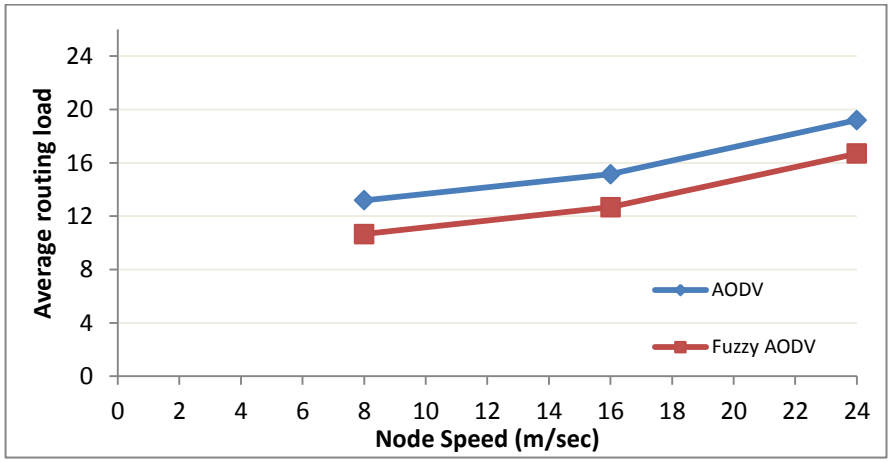
In high speed scenarios (vehicles speed), as shown in Figures 7.2, it is noted that, both routing protocols tested have (approximately) same effects on PDR and throughput performance metrics under various high speed values. PDR for both protocols keep their values reasonably unchanged with the node speed increasing as shown in Figures 7.2 (a). In contrast, the throughput of both protocols increases slightly against node speed increases especially at node speed of 24 (m/sec), as shown in Figure 7.2(b). Figures 7.2 (c) and (d) show the average routing load and average end to end delay performance values of Fuzzy AODV and classical AODV routing protocols. It's clear that the routing load packets in both protocols increase as long as the node speeds increases, that indicates to the increase of the route breaks with the increasing of node speeds, where the routing load packets are being in minimum values for the node speed equal to 8 m/sec. The number of routing load packets increase gradually with the node speed increasing. While, it is noted that in Figure 7.2 (d), the packets end to end delay decreases gradually along the high node speeds increases. The average of end to end delay for both routing protocols have minimum value at maximum simulated speed equals to 24 (m/sec)]. In both Figure 7.1 and Figure 7.2, we note that the modified FuzzyAODV protocol perform better than the classical AODV protocols.


Packet Delivery Ratio (PDR %)
100
96
92
88
84
80
0
4
8
12
16
20
24
Node Speed (m/s)
aodv
Fuzzy AODV

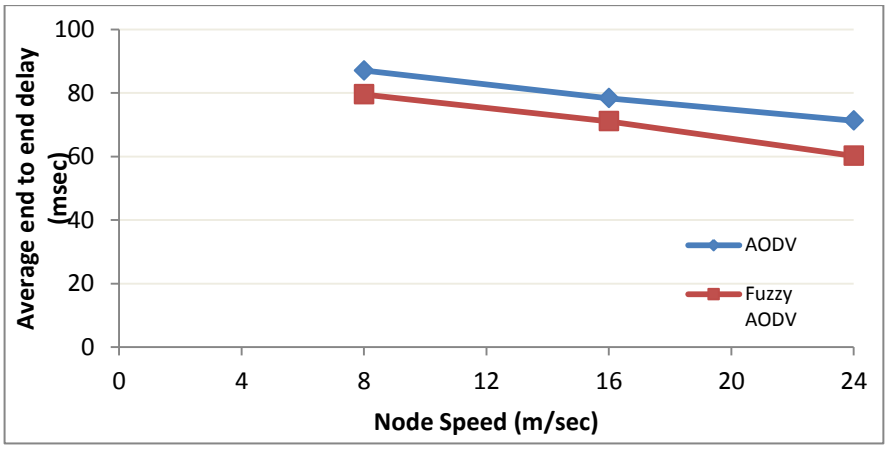
(a)


Average throughput (kbps)
300
250
200
150
100
50
0
0
2
4
6
8
10
12
14
16
18
20
22
24
Node Speed (m/sec)
AODV
Fuzzy AODV

(b)


Average routing load
24
20
16
12
8
4
0
0
2
4
6
8
10
12
14
16
18
20
22
24
Node Speed (m/sec)
AODV
Fuzzy AODV

(c)



(d)

Figure 7.2: AODV and Fuzzy AODV performances vs. node speeds (vehicle speed) with 50 nodes and pause time of 20 sec.

7.2.2 Varying Pause Times

Node mobility is inversely proportional with the node pause times. Node pause time deceases when node mobility increases and vice versa. The simulation scenario of varying node pause time parameter was simulated under node density of 50 nodes and maximum node speed of 20 m/sec. Table 7.3 describes the numerical comparison of the two studied routing protocols with various pause times varying scenarios.

Table 7.3: Average results of AODV and Fuzzy AODV for varying pause time

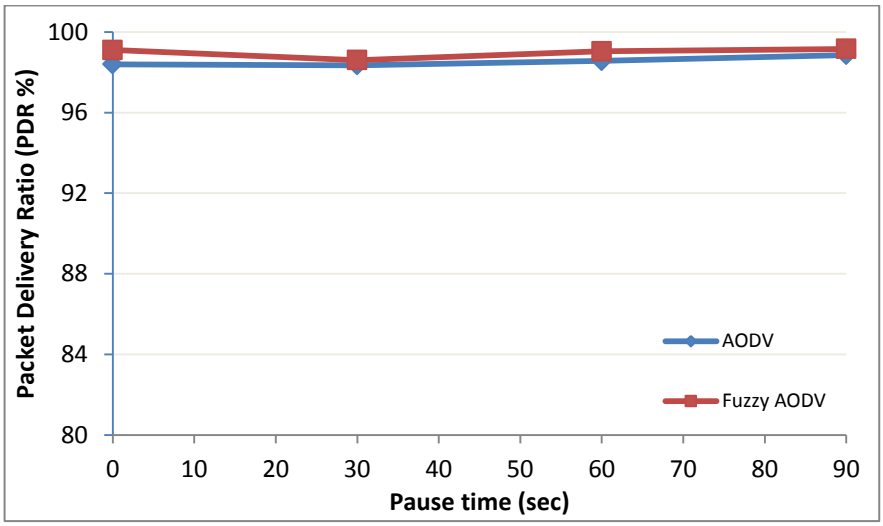
Metric	**Protocol**	**Pause time (sec)**			
		0	**30**	**60**	**90**
PDR (%)	AODV	98.4	98.3	98.56	98.86
	Fuzzy AODV	99.1	98.6	99.04	99.16
Average end to end delay (msec)	AODV	89.8	87.8	81.25	75.755
	Fuzzy AODV	75.5	71.1	70.75	70.34
Average routing load	AODV	18.8	18.2	17.77	13.18
	Fuzzy AODV	16.5	14.7	13.9	10.1

Average throughput (kbps)	AODV	185	200.8	207.9	212.2
	Fuzzy AODV	201	219.6	222.1	222.5

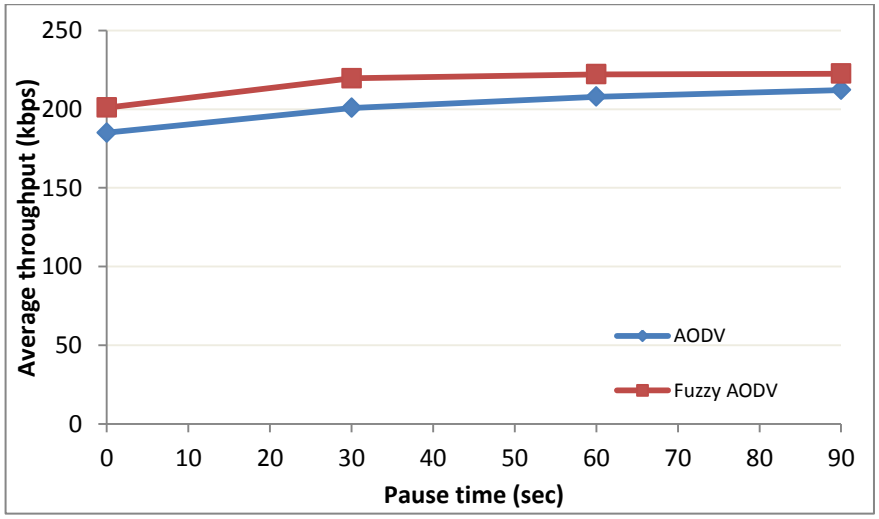
In Figures 7.3(a) and (b), it is noted that there is no sensible effects of nodes mobility on the PDR values over growing node pause times. Throughput of Fuzzy AODV exhibits better performance than classical AODV over different mobility cases. An average route load packets (overhead route control packets) for both protocols decreases significantly with increasing the node pause times especially beyond node pause time approach to 60 sec, as illustrated in Figure 7.3(c). And that is due to the fact of the expected number of route failure increases proportionally with the increasing of node mobility (low pause times). Hence more control packets needed to re-establish the route failure from source to destination nodes. It's clear from the figure; the amount of route overhead packets decreases as the node's pause time increases (low motility cases) and AODV has higher average control overhead packets compared to Fuzzy AODV. Fuzzy AODV protocol minimizes the number of control overhead packets broadcasted over the network as explained in Section 5.3.

Figure 7.3(d) shows the effects of increasing the node pause times on the average end to end delay. Fuzzy AODV has a lower average delay than classic AODV along the pause time values axis. This is due to the frequent route breaks occurring in classical AODV at high mobility scenarios. Frequent route breaking usually increases the packet end to end delays. Also, it shows that the packet delay slightly decreases with the increase of pause times especially after the pause time equal to 60 sec. However,

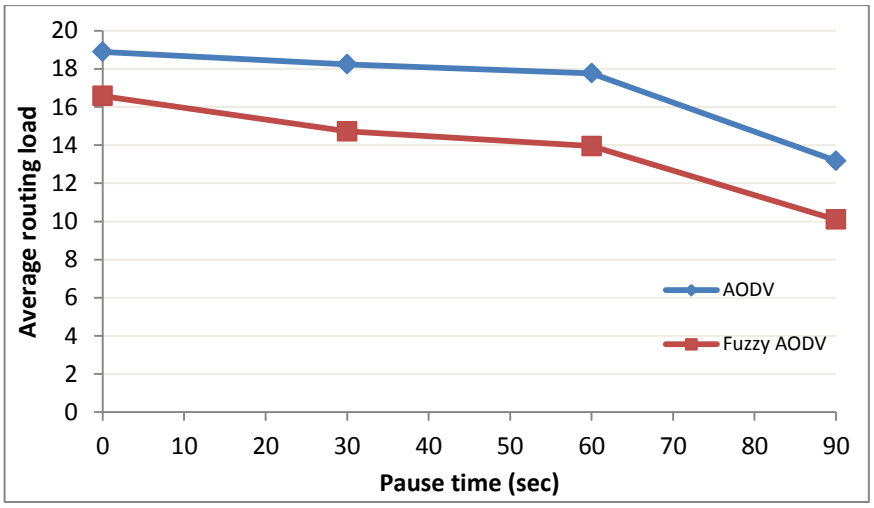
the average delay of Fuzzy AODV gives a better end to end delay performance than the classical AODV.



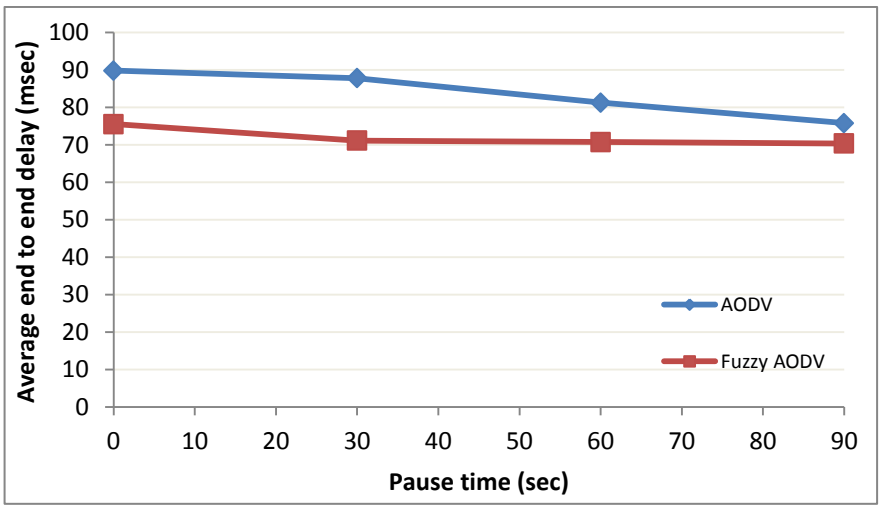
(a)



(b)



(c)



(d)

Figure 7.3: AODV and Fuzzy AODV performances vs. node pause times with 50 nodes and node speed of 20 m/sec

7.2.3 Varying Number of Nodes

Generally, network density is defined as the number of nodes deployed within the square area units. The simulation scenarios of varying number of nodes were simulated under node pause time of 20 sec and maximum node speed of 20 m/sec. Table 7.4 describes the numerical comparison of the two studied routing protocols with different number of nodes varying scenario.

Table 7.4: Average results of AODV and Fuzzy AODV for varying number of nodes

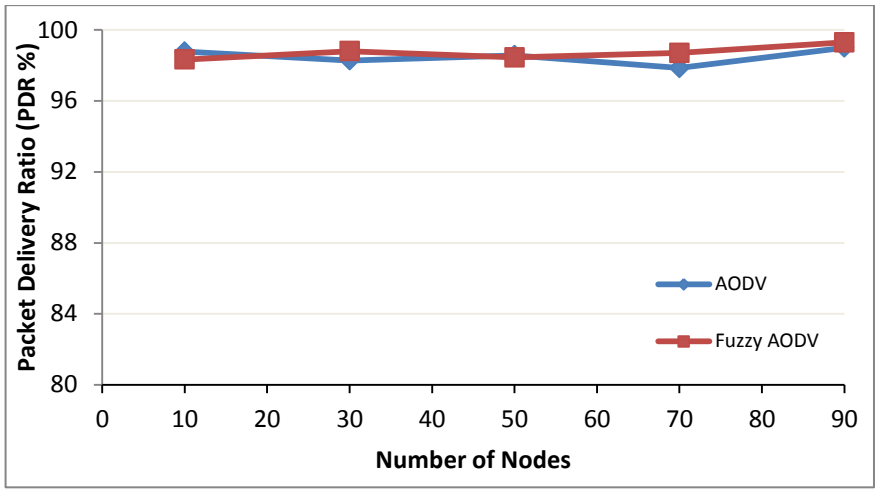
Metric	Protocol	Number of nodes				
		10	**30**	**50**	**70**	**90**
PDR (%)	AODV	98.775	98.28	98.57	97.86	99
	Fuzzy AODV	98.33	98.8	98.45	98.7	99.3
Average end to end delay (msec)	AODV	58.544	80.825	86.79	89.7	76.5
	Fuzzy AODV	58.13	77.2	81	79.5	70.4
Average routing load	AODV	0.97	10.85	18.6	21.7	22.87
	Fuzzy AODV	0.85	8.46	16.12	17.7	19.27
Average throughput (kbps)	AODV	143.7	202.3	200.15	210	224
	Fuzzy AODV	165.75	205.1	205.7	215	229

Fuzzy AODV and classical AODV have same behaviors in both PDR and throughput performance metrics as shown in Figure 7.4 (a) and (b). The packet delivery ratio of both protocols are somewhat have unchanged values along the variation of the network densities. However, It can be clearly seen that the amount of packets received by destination nodes significantly increases with the increase of node density (from 10 nodes to 30 nodes) and the throughput slightly increases at higher node densities (between 30 to 90 nodes), as shown in Figure 7.4 (b).

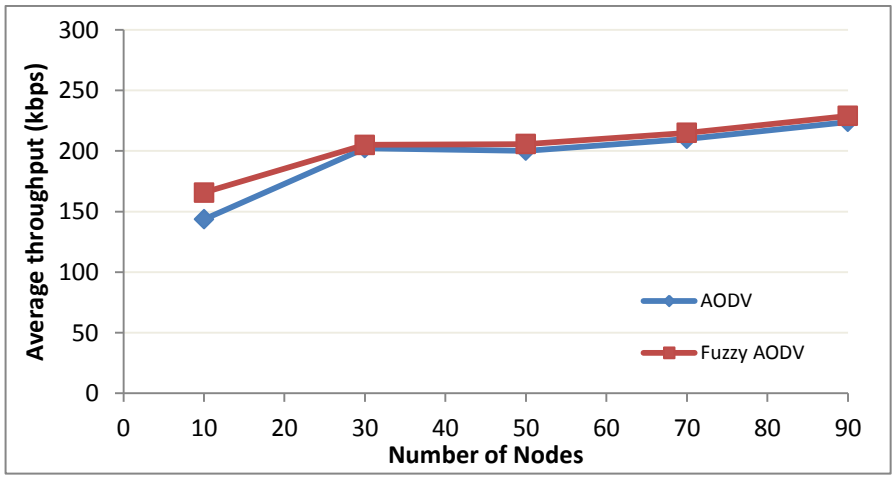
In the two routing protocols, the average routing load increases gradually as the number of nodes increase, and that is due to the fact of the increasing the number of node will contribute in increasing the control packets exchanges between the nodes during route establishing at route discovery stages. However, as shown in Figure 7.4 (c) the classic AODV's routing load has, in general, higher values than Fuzzy AODV

protocol for all network densities. In comparison, Fuzzy AODV has a lower routing load compared with classic AODV routing protocol as shown in Figure 7.4(c) because the proposed Fuzzy AODV set a high stable route with trusted nodes comparing with classical AODV. As a result, proposed Fuzzy AODV decreases the probability of route failures during the route life time and consequently it decreases the control overhead packets flooded throughout the network each time a route failure occurs.

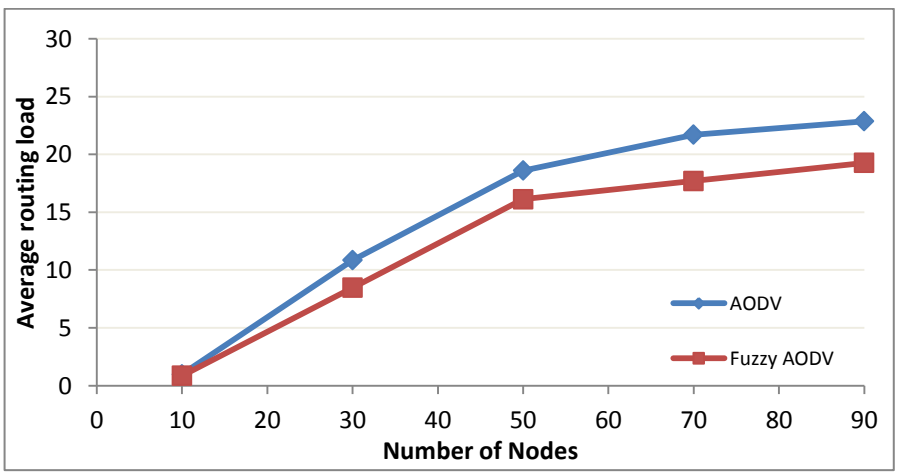
Figure 7.4 (d) shows, end to end delay of both routing protocols have lowest delay at node density of 10 nodes. Delay values increases gradually specifically at node density of 70 nodes. Then, it is noted that the delay decreases slightly at node density of 90 nodes. Comparing with classical AODV, the Fuzzy AODV has lower average end to end delay along the increasing the number of nodes. When the node density increases, the number of control routing packets sources also increases, causing increasing of packets collisions which leads to increasing of average end-to-end delay in both routing protocols. It is important to note that, both routing protocols have lowest end-to-end delay at the minimum number of nodes that is due to the facts of the minimum number of nodes generates minimum amount of overhead control packets so it has loweet routing loads as illustrated in Figure 7.4 (c).


Packet Delivery Ratio (PDR %)
100
96
92
88
84
80
0
10
20
30
40
50
60
70
80
90
Number of Nodes
AODV
Fuzzy AODV

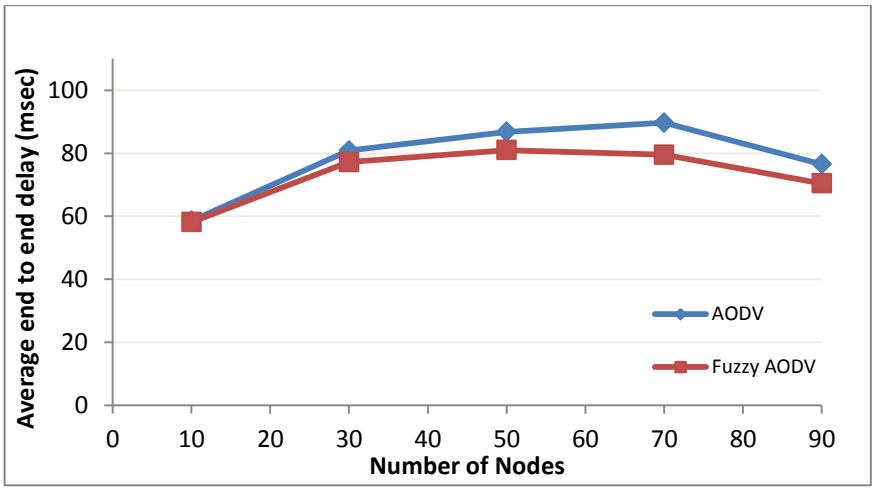
(a)


Average throughput (kbps)
300
250
200
150
100
50
0
0
10
20
30
40
50
60
70
80
90
Number of Nodes
AODV
Fuzzy AODV

(b)


Average routing load
30
25
20
15
10
5
0
0
10
20
30
40
50
60
70
80
90
Number of Nodes
AODV
Fuzzy AODV

(c)



(d)

Figure 7.4: AODV and Fuzzy AODV performances vs. number of nodes with node speed of 20 m/sec and pause time of 20 sec

7.2.4 Traffic Pattern Comparison Results

Voice and video communication applications are mainly constant data rate datagram applications. To emulate similar loads, constant bit rate (CBR) traffic was used as application traffic model running over User Datagram Protocol (UDP) transport layer [86]. Figures 7.5 (a) and (b) are snapshots showing a simple NS-2 simulation script that create TCP and UDP agent, also create FTP and CBR as new NS applications [87].

```
#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
```

(a)

```
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false
```

(b)

Figure 7.5: Snapshot of NS-2 agents and application layer generating commands [87]

The simulation scenario for TCP and CBR traffic flow patterns for classic AODV and Fuzzy AODV routing protocols was simulated under different node densities of 10, 30, 60, and 90 nodes deployed randomly across simulation area of 900 x 900 square meters, the maximum node speed of 2 m/sec and node pause time equal to zero was used. Table 7.5 describes the numerical comparison of average results of the two studied routing protocols of different traffic flow patterns for both routing protocols scenarios.

Table 7.5: Average results of AODV and Fuzzy AODV for FTP and CBR traffic flows

Metric	**Traffic pattern**	**Protocol**	**Number of nodes**			
			10	**30**	**60**	**90**
PDR (%)	TCP traffic (FTP)	AODV	94.7	98.8	99.6	99.4
		Fuzzy AODV	93.3	99.2	99.4	99.2
	UDP traffic (CBR)	AODV	43.1	94.6	99.1	99.4
		Fuzzy AODV	49.13	96.2	99.3	99.8
Average end to end delay (msec)	TCP traffic (FTP)	AODV	71.6	90.44	102.3	98.6
		Fuzzy AODV	68	78.289	86.7	90.21
	UDP traffic (CBR)	AODV	204.86	104.24	62.51	36.2
		Fuzzy AODV	192.43	98.555	52.09	22.9
Average routing load	TCP traffic (FTP)	AODV	2.97	5.92	16.768	19.438
		Fuzzy AODV	1.09	5.21	14.8	16.846
	UDP traffic (CBR)	AODV	0.06	0.325	1.2	1.139
		Fuzzy AODV	0.48	0.72	1.16	1.192
	TCP	AODV	46.7	95	149	158.06

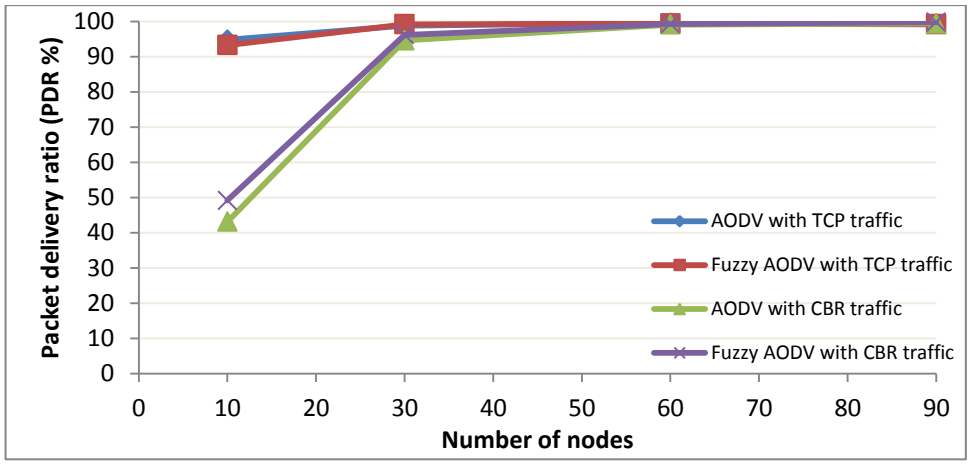
Average throughput (kbps)	traffic (FTP)	Fuzzy AODV	47.37	110	154	162.86
	UDP traffic (CBR)	AODV	7.36	8.16	8.1	8.2
		Fuzzy AODV	14.37	15.9	16.2	16.3

Figure 7.6 (a) shows the effects of using different traffic flow models (UDP and TCP) on the AODV and Fuzzy AODV performance. Packet delivery ratio was used as a measure of the network throughput when the traffic flows are increased, there will be an obvious increase in the throughput, but that does not necessarily indicate that the overall delivery percentage has been improved. Figure 7.6 (a) and (b) show packet delivery ratio and the throughput the routing protocols against the increasing of number of nodes with different traffic flows. It is noted that from Figure 7.6 (a), the connection oriented scheme (TCP) traffic flow exhibits a maximum constant values close to the maximum rate of packet delivery ratio at various node densities compared with the connectionless scheme (CBR). CBR throughput values increase gradually for both routing protocols and approach to the maximum value of PDR at network density equal to 30 nodes where the Fuzzy AODV with CBR traffic was improved in 4.17% over the classical AODV. CBR traffic flow presents consistent behavior and lesser magnitudes compared with the throughput of TCP traffic flows as shown in Figure 7.6 (b). In both cases of traffic flows, the Fuzzy AODV using TCP and CBR traffic models perform better than the classical AODV and improving the average throughput percentage values about 5% and 90% respectively.

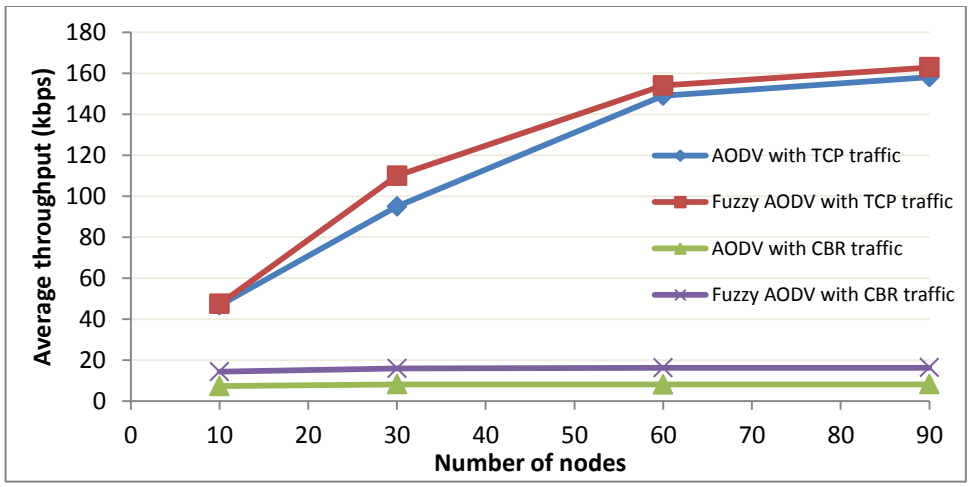
Figure 7.6 (c) shows the routing protocols under TCP flow flooded more overhead control packets which increases proportionally with the increase of the node density

especially beyond number of nodes equal to 30. That is due to the fact that the connection oriented scheme uses more control packets to maintain the communication between terminals compared to connectionless scheme. AODV and Fuzzy AODV protocols using CBR traffic flows have lower and approximately fixed routing load values compared to protocols uses TCP traffic flows as shown in Figure 7.6 (c). It is noted that the FuzzyAODV with TCP traffic protocol decreases the amount of routing control packets broadcasted with 24% over the classical AODV, this improvement, the minimization of the routing control packets, also affects the network scalability in good manner.

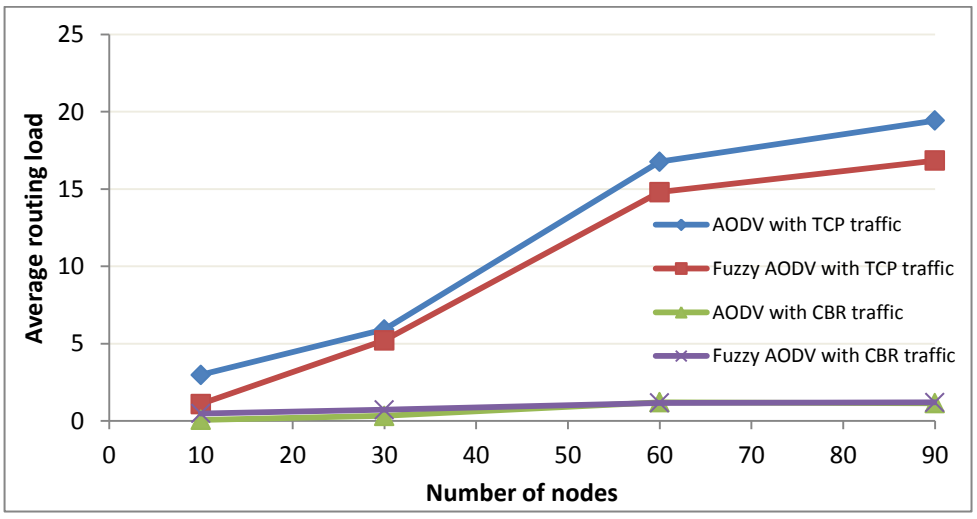
As the node density increases, both routing protocols under the TCP traffic have higher end to end delays, especially beyond the node density of 30 nodes, when compared to CBR traffic. Before the node density of 30 nodes, the magnitude of end to end delay of routing protocols under CBR traffic have smaller delay values than the routing protocols under TCP traffic as shown in Figure 7.6 (d). The value of end to end delay of routing protocols under CBR traffics decreases gradually with the increase of node density and reaches to minimum value at the highest node density. Whereas, the value of end to end delay of routing protocols which uses TCP traffic increases slightly along the node density variations? In both scenarios of traffic models used in Fuzzy and classical AODV routing protocol, it is noted that the average end to end delay was improved as 10.5% for Fuzzy AODV under TCP and 16% for Fuzzy AODV under CBR traffic when compared to the classical AODV under TCP and CBR traffics respectively.


Packet delivery ratio (PDR %)
100
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
90
Number of nodes
AODV with TCP traffic
Fuzzy AODV with TCP traffic
AODV with CBR traffic
Fuzzy AODV with CBR traffic

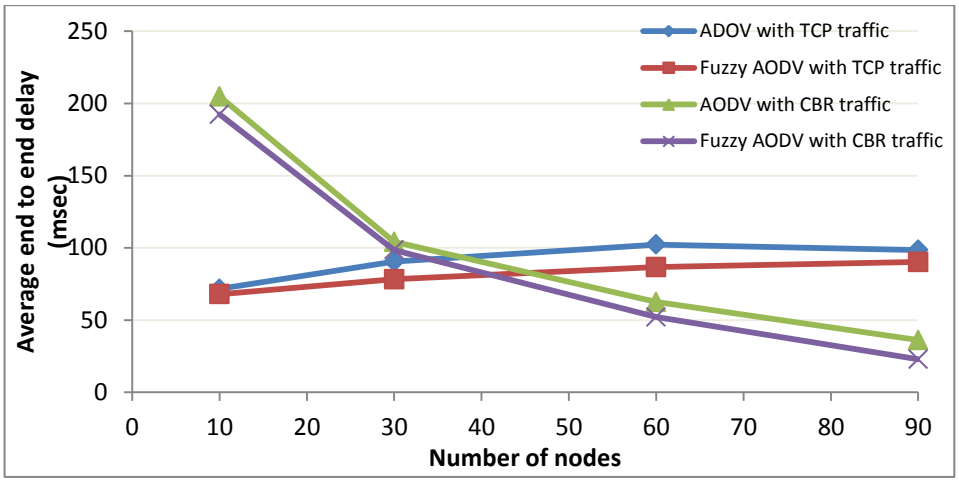
(a)


Average throughput (kbps)
180
160
140
120
100
80
60
40
20
0
0
10
20
30
40
50
60
70
80
90
Number of nodes
AODV with TCP traffic
Fuzzy AODV with TCP traffic
AODV with CBR traffic
Fuzzy AODV with CBR traffic

(b)


Average routing load
25
20
15
10
5
0
0
10
20
30
40
50
60
70
80
90
Number of nodes
AODV with TCP traffic
Fuzzy AODV with TCP traffic
AODV with CBR traffic
Fuzzy AODV with CBR traffic

(c)



(d)

Figure 7.6: AODV and Fuzzy AODV performances vs. number of nodes for different traffic flows (node speed=2 m/sec and pause time = 0 sec)

7.2.5 Confidence Interval Computations

The confidence interval (CI) is calculated for simulation scenario of 50 nodes with maximum node speed of 20 m/sec and 30 sec pause time. Initially, twenty different random mobility scenario files generated. Each mobility file used to run AODV and Fuzzy AODV routing protocols to produce simulation performance outputs as shown in Table 7.6. Then, the sample mean value and sample standard deviation for confidence intervals for confidence levels of 90%, 95%, and 99% respectively calculated by determining the values of $t_{\alpha/2,\ n-1}$ using *t* distribution table [88]. For example, for confidence 90%, we can find the CI as following:

Confidence level = 100 * (1-α) %

90% = 100 *(1-α) %

0.9 = 1-α

α = 0.1

α/2 =0.05

From *t* distribution table in reference [89], the value of $t_{0.05,n-1}$ found, where *n* is the number of replications (Table 7.7, shows the values of $t_{\alpha/2,n-1}$ for different confidence

levels and different number of replications). Then, the confidence intervals using equation (1), in Appendix A, calculated. Details are explained in Appendix A. Tables 7.8 and 7.9 show confidence intervals values for AODV and Fuzzy AODV routing protocols under different confidence levels.

Table 7.6: AODV and Fuzzy AODV performance values for different mobility files

Performance metrics / Mobility files	AODV				Fuzzy AODV			
	PDR (%)	Average end to end delay (msec)	Average routing load	Average throughput (Kbps)	PDR (%)	Average end to end delay (msec)	Average routing load	Average throughput (Kbps)
Mob.file1	97.6	104.6	22.5	121	98.1	94.2	16.2	136
Mob.file2	98.2	91.2	17.3	179	99.2	66.6	17.01	194
Mob.file3	96.1	101.1	28.8	169	96.8	71.9	26.6	195
Mob.file4	98.5	71.2	18.5	241	98.7	55.8	16.9	264
Mob.file5	98.7	113.5	19.89	106	98.9	96.3	15.8	117
Mob.file6	99.1	88.3	9.96	198	99.7	56.3	9.02	226
Mob.file7	99.1	85	21.5	180	99.4	86.9	18.2	199
Mob.file8	98.9	56.2	16.4	215	99.1	76.1	14.8	250
Mob.file9	96.8	87.3	20.3	259	97.6	71.9	18.9	289
Mob.file10	99.6	59.7	10.5	271	99.7	65.2	7.82	297
Mob.file11	98.2	106.4	18.2	160	98.9	96.3	16.9	184
Mob.file12	98.4	93.1	10.79	183	98.6	56.3	8.9	212
Mob.file13	98.9	105	28.7	143	99.2	81.6	14.9	151
Mob.file14	98.2	73.8	20.1	204	98.9	52.1	15.2	225
Mob.file15	98.1	85.8	24.9	226	98.3	67.4	21	246
Mob.file16	98	80.5	12.9	235	98.4	44.2	6.94	238
Mob.file17	98.5	95.1	16.8	260	98.7	79.7	10.9	261
Mob.file18	97.8	79.3	17.7	194	97.8	61.3	13.5	210
Mob.file19	98.1	89.9	17.9	209	98.3	75.6	14.1	218
Mob.file20	99.4	88.9	11.7	264	99.6	67	10.8	281

.

Table 7.7: The values of *t* parameter for different CL and different number of replications [88]

Confidence level (C.L.) / Replications (n)	90%	95%	99%
5	2.132	2.776	4.604
10	1.833	2.262	3.250
15	1.761	2.145	2.977
20	1.729	2.093	2.861

Table 7.8: AODV confidence intervals for simulation scenario of number of nodes= 50, maximum node speed = 20 m/sec, and pause time =30 sec

Parameters / Performance metric	Samples mean (Q)	Sample standard deviation (s)	Standard error (s/√n)	C.I. of 90% confidence	C.I. of 95% confidence	C.I. of 99% confidence
PDR (%)	98.312	0.832	0.186	± 0.321	± 0.389	± 0.532
Average end to end delay (msec)	87.795	14.97	3.347	± 5.786	± 7.005	± 9.575
Average routing load	18.267	5.460	1.22	± 2.109	± 2.553	± 3.490
Average throughput (kbps)	200.85	47.108	10.355	± 17.903	± 21.673	± 29.625

Table 7.9: Fuzzy AODV confidence intervals for simulation scenario of number of nodes= 50, maximum node speed = 20 m/sec, and pause time =30 sec

Parameters / Performance metric	Samples mean (Q)	Sample standard deviation (s)	Standard error (s/√n)	C.I. of 90% confidence	C.I. of 95% confidence	C.I. of 99% confidence
PDR (%)	98.695	0.743	0.166	± 0.287	± 0.347	± 0.479
Average end to end delay (msec)	71.135	14.907	3.333	± 5.762	± 6.975	± 9.527

Average routing load	14.719	4.784	1.055	± 1.824	± 2.208	± 3.018
Average throughput (kbps)	219.65	48.99	10.95	± 18.932	± 22.918	± 31.327

Chapter 8

CONCLUSION

This study provided a detailed investigation of operation and performance of classic AODV and a modified AODV protocol using network simulation. Fuzzy Inference logic scheme is used to improve the classic AODV routing protocol. The Fuzzy modified protocol is called Fuzzy based AODV protocol (Fuzzy AODV). Fuzzy logic appears to be an efficient approach for constructing robust routes and avoiding some of the shortcomings of the simple single-metric classical AODV reactive routing protocol. The Fuzzy AODV protocol has adaptive properties that improve the performance of classical AODV routing protocol. Good routing performance indicators summarized as high throughput and highest PDR, low average end to end delay, as well as low control route load packets. Our simulation study results investigate that the Fuzzy proposed protocol modifications, under different network environments, behaves better than the classical AODV, in the sense of control overhead packets and end to end delays, as shown in various simulation results. Also the network throughput of the proposed algorithm is slightly improved than the classical AODVs' protocol.

In the future, the simulation study can be extended to study more factors and metrics that may be considered in the fuzzy inference engine to enhance the route selection efficiency to construct more stable route between source to destination. Also, different MANETs routing protocols such as DSR and DSDV routing protocols can

be improved by modifying their route selection mechanisms to investigate the effect of Fuzzy Inference modification in their performance. Also, the simulation of improved Fuzzy AODV will be generalized to study the routing protocol under various environment parameters and different network loads.

REFERENCES

[1] Tanaka, T., Inayoshi, M., & Mizuhara, N. (1998). Overview of Communication Network Evolution. *Hitachi Review*. Vol. 4, (pp.45-50).

[2] Bansal, R. K., Gupta, V., & Malhotra, R. (2010). Performance Analysis of Wired and Wireless LAN Using Soft Computing Techniques- A Review. *Global Journal of Computer Science and Technology*, Vol. 10 Issue 8, (pp. 67-71).

[3] Kaur, S., kaur, S. & Sharma, C. (2013). An Overview of Mobile Ad Hoc Network: Application, Challenges and Comparison of Routing Protocol. *Journal of Computer Engineering*, Volume 11, Issue 5, (pp.07-11).

[4] Legendre, F., Hossmann, T., Sutton, F., & Plattne, B. (2011). 30 Years of Wireless Ad Hoc Networking Research: What about Humanitarian and Disaster Relief Solutions? What are we still missing?. *ACWR*, Amritapuri, Kollam, Kerala, India, ACM. (pp.03-11)

[5] Kapadia, V. V., Patel, S. N., & Jhaveri, R. H. (Mar. 2010). Comparative Study of Hidden Node Problem and Solution Using Different Techniques and Protocols. *Journal of Computing*, Vol. 2, Issue 3, (pp.65-67)

[6] Sarkar, S. K., Basavaraju, T., & C. Puttamadappa (2008). Ad Hoc Mobile Wireless Networks: Principles, Protocols, and Applications, Auerbach

Publications, *Taylor & Francis Group*.

[7] Monks, J. P. (2001). Transmission Power Control for Enhancing the Performance of Wireless Packet Data Networks. *University of Illinois at Urbana-Champaign*. (pp. 1-21)

[8] Sharma, P., Karkhanawala, Y., & Kotecha, k. (2011). Bandwidth Constrained Routing of Multimedia Traffic over Hybrid MANETs Using Ant Colony Optimization. *International Journal of Machine Learning and Computing*, Vol. 1, No. 3. (pp. 242-246).

[9] Ishibashi, B., & Boutaba, R. (2005). Topology and Mobility Considerations in Mobile Ad Hoc Networks. Elsevier, *Ad Hoc Networks*, (pp.762–776).

[10] Aarti, & Tyagi, S. S. (2013). Study of MANET: Characteristics, Challenges, Application and Security Attacks. *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 3, Issue 5, (pp. 252-257).

[11] Chitkara, M., & Ahmad, M. W. (2014). Review on MANET: Characteristics, Challenges, Imperatives and Routing Protocols. *International Journal of Computer Science and Mobile Computing*, Vol. 3, Issue. 2, (pp. 432-437).

[12] Goyal, P., Parmar,V., & Rishi, R, (2011). MANET: Vulnerabilities, Attacks, Application. *International Journal of Computational Engineering &*

Management, Vol. 11, (pp.32-37).

[13] Stuedi, P. (2008). Fundamental Properties and Services of Mobile Ad Hoc Networks. *Swiss Federal Institute of Technology Zurich*, Zurich, (pp.17-28)

[14] Dhawan, S., Saroha, V., & Dhawan, R. (2013). Review on Performance Issues of Routing Protocols of Mobile Ad Hoc Networks. *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 3, Issue 6, (pp.1024-1029).

[15] Hinds, A., Ngulube, M., Zhu, S,. & Al-Aqrabi, H. (2013). A Review of Routing Protocols for Mobile Ad Hoc Networks. *International Journal of Information and Education Technology*, Vol. 3, No. 1, (pp. 1–5).

[16] Jacob, J., & V. Seethalakshmi. (2012). Performance Analysis and Enhancement of Routing Protocol in MANET. *International Journal of Modern Engineering Research*, Vol.2, (pp-323-328).

[17] Halvardsson, M. & Lindberg, P. (2004). Reliable Group Communication in A Military Mobile Ad Hoc Network. A Report from School of Mathematics and Systems Engineering. *Vaxjo University*, Sweden, (pp. 15-20)

[18] Perkins, C., Royer, C. E. & Das, S. (2003). Ad Hoc On-Demand Distance Vector (AODV) Routing - Internet Draft. RFC 3561, *IETF Network Working Group*, (pp.1-37).

[19] Nishat, H., Krishna, F., Rao, D., & Ahmed, s. (2011). Performance Evaluation of On Demand Routing Protocols AODV and Modified AODV (R-AODV) in MANETS. *International Journal of Distributed and Parallel Systems.* Vol.2, No.1, (pp. 94-101).

[20] Al-Jarrah, O., & Megdadi, O. (2008). Enhanced AODV routing protocol for Bluetooth scatternet. *Elsevie*r, Computers and Electrical Engineering, (pp.1-12).

[21] Das,S., R., Perkins, C. E., & Royer, E. M. (2000). Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. *IEEE INFOCOM*. (pp. 3-12)

[22] Ortiz, A. M., & Olivares, T. (2012). Fuzzy Logic Applied to Decision Making in Wireless Sensor Networks. *Fuzzy Logic-Emerging Technologies and Applications*, (pp. 221–240).

[23] Sarma, N., & Nandi, S. (2010). Route Stability Based QoS Routing in Mobile Ad Hoc Networks. *Wireless Personal Multimedia Communications*, Vol.54, Issue 1, (pp. 203-224).

[24] Thenmozhi, D.S., Rajaram, M. (2011). Contention Aware Multi-hop Stable Routing to Provide Quality of Service Based on Multiple Constraints in Mobile Ad Hoc Networks. *European Journal of Scientific Research*, Vol.48, No.4, (pp.567-579).

[25] Hunda, G. S., Gupta, S. K., & Bedi A. (2014). Adaptive Approach to Find A Stable Path Between Nodes in MANET. *International Journal of Current Engineering and Technology*. Vol.4, No.4, (pp.2898-2091)

[26] Toh, C. K. (1997). Associativity-Based Routing for Ad Hoc Mobile Networks. *International Journal on Wireless Personal Communications*, Vol. 4, Issue 2, (pp. 103-139).

[27] Sridhar, K. N., & Chan, M. C. (2005).Stability and Hop-Count Based Approach for Route Computation in MANET. *IEEE*, (pp. 25-31).

[28] Yang, P., & Huang, B., (2008). QoS Routing Protocol Based on Link Stability with Dynamic Delay Prediction in MANET. IEEE, *Pacific-Asia Workshop on Computational Intelligence and Industrial Applicatio*n, Vol.1, (pp. 515-518).

[29] Sarma, N., & Nandi, S. (2010). Route Stability Based QoS Routing in Mobile Ad Hoc Networks. *Wireless Personal Multimedia Communications*, Vol.54, Issue 1, (pp. 203-224).

[30] Bingkun X. & Yanping L. (2014). A Novel Link Stability and Energy Aware Routing with Tradeoff Strategy in Mobile Ad Hoc Networks. *Journal of Communications.* Vol. 9, Issue 9, {pp. 706-713).

[31] Toh, C.K. (2001). Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks. *IEEE Communications*

Magazine, (pp. 2-11).

[32] Korkmaz, T. & Krunz, M. (2003). Bandwidth-Delay Constrained Path Selection Under Inaccurate Slate Information. *IEEE/ACM,* Vol. II, Issue 3, (pp. 384 – 398).

[33] Khanpara, P. (2014). A Review on Fuzzy Logic Based Routing in Ad Hoc Networks. *International journal of advanced research in engineering and technology*. Vol. 5, Issue 5, (pp. 75-81).

[34] Santhi, G., & Nachiappan, A. (2012). Fuzzy-Cost Based Multiconstrained Qos Routing With Mobility Prediction in MANETs. *Egyptian Informatics Journal*. Vol. 13, Issue 1, (pp. 19-25).

[35] Pi, S., & Sun, B. (2012). Fuzzy Controllers Based Multipath Routing Algorithm in MANET. *International Conference on Applied Physics and Industrial Engineering.* Elsevier Physics Procedia 24, (pp.1178 – 1185).

[36] Ali, M., Stewart, B. G., Shahrabi, A., & Vallavaraj, A. (2015). Fuzzy Based Load and Energy Aware Multipath Routing for Mobile Ad Hoc Network. *International Journal of Computer Applications*. Vol. 114, Issue 16, (pp. 25-32).

[37] Dahiya, R., Dureja, A. (2014). Fuzzy Based Efficient Routing Protocol for Route Recovery in MANET. I*nternational Journal of Engineering and*

Computer Science, Volume 3 Issue 6, (pp. 6681-6687).

[38] Singhal, A., & Daniel, A.K. (2014). Fuzzy Logic based Stable On-demand Multipath Routing Protocol for Mobile Ad Hoc Network. IEEE, *Fourth International Conference on Advanced Computing & Communication Technologies*, (pp. 421-426).

[39] Gad, W., & Abdelkader, T. (2014). A Fuzzy-based Routing Protocol for Metropolitan-Area Mobile Ad Hoc Networks. *IEEE*. (pp.133-138).

[40] Geetha, k., Thangaraj, P. (2015). An Enhanced Associativity Based Routing with Fuzzy Based Trust to Mitigate Network Attacks. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, Vol. 9, No. 8, (pp. 1926-1934).

[41] Dana, A., & Babaei, M. H. (2011). A Fuzzy Based Stable Routing Algorithm for MANET. *International Journal of Computer Science Issues*, Vol. 8, Issue 1, (pp. 367- 371).

[42] Jain, B., & Shrivastava, L. (2015). Improvement in Conventional AODV Routing Protocol Using Fuzzy Logic Based Controller. *IJCSN* Vol.5, Issue 2, (pp. 44-51).

[43] Ghalavand, G., Dana, A., Ghalavand, A., & Rezahosieni, M. (2010). Reliable Routing Algorithm Based on Fuzzy Logic For Mobile Ad Hoc Network. *3rd International Conference on Advanced Computer Theory and Engineering*

IEEE. (pp.606-609)

[44] Srivastava,S., & Daniel, A.K. (2013). An Efficient Routing Protocol Under Noisy Environment for Mobile Ad Hoc Networks Using Fuzzy Logic. *International Journal of Advanced Research in Artificial Intelligence*, Vol. 2, No. 6, (pp. 34-39)

[45] Chaythanya, P., & Ramya, M. M. (2014). Fuzzy Logic Based Approach for Dynamic Routing in MANE. *International Journal of Engineering Research & Technology*, Vol. 3 Issue 6, p(1437-1441).

[46] Sataraddi, M. J., & Budyal, V R. (2014). Bandwidth and Delay Guaranteed Unicast Routing in Mobile Ad Hoc Networks using Fuzzy Logic. *International Journal of Advances in Computer Networks and Its Security*, Volume 4, Issue 2, (pp.51-55).

[47] Deshpande, K. G. (2012). Agent Approach QoS Routing in MANET Based on Fuzzy Priority Scheduler. *International Journal of Electronics Signals and Systems*, Vol.1, Issue 2, (pp. 76-81).

[48] Kalpana, G., & Punithavalli, M. (2013). Fuzzy Logic Technique for Gossip Based Reliable Broadcasting in Mobile Ad Hoc Networks. *Journal of Theoretical and Applied Information Technology*, Vol. 51, Issue 3, (pp. 498-505).

[49] Legendre, F., Hossmann, T., Sutton, F., & Plattner, B. (2011). 30 Years of

Wireless Ad Hoc Networking Research: What about Humanitarian and Disaster Relief Solutions? What are we still missing?. *ACWR* , Dec 18-21, Amritapuri, Kollam, Kerala, India, ACM.

[50] Verma, S., Nayak, P., & Agarwa, R., (2012). Energy Efficient Routing in Mobile Ad Hoc Networks based on AODV Protocol. *International Journal of Computer Science Issues*, Vol. 9, Issue 6, No 2, (pp. 344-349).

[51] Li, P., Guo, S., Yu, S., & Vasilakos, A. A. (2014). Reliable Multicast with Pipelined Network Coding Using Opportunistic Feeding and Routing. *IEEE Transactions on Parallel and Distributed Systems*, VOL. 25, No. 12, (pp. 3264-3273).

[52] Legendre, F., Hossmann, T., Sutton, F., & Plattner, B. (2011). 30 Years of Wireless Ad Hoc Networking Research: What about Humanitarian and Disaster Relief Solutions? What are we still missing?. *ACWR '11*, Amritapuri, Kollam, Kerala, India, ACM.

[53] Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control 8*, (pp. 338-353).

[54] Dernoncourt, F. (2013). Introduction to fuzzy logic. *MIT*.(pp.1-21)

[55] Marwaha, S., Srinivasan, D., Tham, C. K., & Vasilakos, A. (2004). Evolutionary Fuzzy Multi-Objective Routing For Wireless Mobile Ad Hoc Networks. *CEC2004. Congress on*, Vol. 2, (pp. 1964-1971).

[56] Abbas, N. I., Ilkan, M., & Ozen, E..(2015). Fuzzy Approach to Improving Route Stability of the AODV Routing Protocol. *EURASIP Journal on Wireless Communications and Networking.* (pp.1-11).

[57] Ban, X., Gao, X. Z., Huang, X., & Yin, H. (2006). Stability Analysis of the Simplest Takagi-Sugeno Fuzzy Control System Using Circle Criterion. IEEE *International Conference on Fuzzy Systems*, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada. (pp. 16-21).

[58] Vasilakos, A., Saltouros, M. P., Atlassis, A. F., & Pedrycz, W. (2003). Optimizing QoS Routing in Hierarchical ATM Networks Using Computational Intelligence Techniques. *IEEE Transaction on Systems, Man, and Cybernetics—Part C: Applications and Reviews,* Vol. 33, No. 3, (pp. 297-312).

[59] El-Hajj, W., Kountanis, D., Al-Fuqaha, A., & Guizani, M. (2006). A Fuzzy-Based Hierarchical Energy Efficient Routing Protocol for Large Scale Mobile Ad Hoc Networks (FEER). *IEEE Communications Society,* IEEE. (pp.3585-3590).

[60] Hua, H. W., Ling,F. K., & Zhenge, Z. (2010). Sufficient Condition of Universal Approximation of Fuzzy Controllers with Generalized Linear Membership Function. *IEEE, Proceedings of the 29th Chinese Control Conference* July 29-31, Beijing, China.(pp.2483-2487).

[61] Barua, A., Mudunuri, L. S. & Kosheleva, O. (2014). Why Trapezoidal and

Triangular Membership Functions Work So Well: Towards a Theoretical Explanation. *Journal of Uncertain Systems*. Vol.8, Issue 3, (pp. 164-168).

[62] Pedrycz, W. (1994). Why Triangular Membership Functions?. *Fuzzy Sets and Systems*, Vol. 64, Issue 1, (PP. 21–30).

[63] Marwaha, S.,Srinivasan, D., Tham, C. k., & Vasilakos, A. (2004). Evolutionary Fuzzy Multi-Objective Routing For Wireless Mobile Ad Hoc Network. Evolutionary Computation, *CEC2004 Congress 2, IEEE*, (pp.1964–1971).

[64] Youssef, M., Ibrahim, M., Abdelatif, M., Chen, L., & Vasilakos, A. (2014). Routing Metrics of Cognitive Radio Networks: A Survey. *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 1, (pp. 92-109).

[65] Nurcahyo, G. W., Shamsuddin, S. M., Alias, R. A., Sap, N. M. (2003). Selection of Defuzzification Method to Obtain Crisp Value for Representing Uncertain Data in a Modified Sweep Algorithm. *JCS&T*. Vol. 3 No. 2 (pp. 22-28).

[66] Ross, T. J. (2010). Fuzzy Logic with Engineering Applications. Third Edition, *John Wiley & Sons*, Ltd. ISBN: 978-0-470-74376-8.

[67] Chaythanya, B. P., & Ramya, M. M. (2014). Fuzzy Logic Based Approach for Dynamic Routing in MANET. *International Journal of Engineering*

Research & Technology, Vol. 3 Issue 6, (pp.1347-1441).

[68] Naaz, S., Alam, A., & Biswas, R. (2011). Effect of Different Defuzzification Methods in A Fuzzy Based Load Balancing Application, *International Journal of Computer Science Issues*, Vol. 8, Issue 5, No 1, (pp. 261-267).

[69] Lee, C. (1990). Fuzzy logic in control systems: fuzzy logic controller, Parts I and II, *IEEE Trans. Syst., Man, Cybern*. Vol. 20, (pp.404–435).

[70] Kalpana, G., & Punithavalli, M. (2013). Fuzzy Logic Technique for Gossip Based Reliable Broadcasting in Mobile Ad Hoc Networks. *Journal of Theoretical and Applied Information Technology,* Vol. 51 No.3, (pp.498-505).

[71] Gad, W., & Abdelkader, T. (2014). A Fuzzy-based Routing Protocol for Metropolitan-Area Mobile Ad Hoc Networks. *IEEE,* (pp. 133-138).

[72] Sataraddi, M. J., & Budyal, V. R. (2014). Bandwidth and Delay Guaranteed Unicast Routing in Mobile Ad Hoc Networks Using Fuzzy Logic. *International Journal of Advances in Computer Networks and Its Security*. Vol. 4: Issue 2, (pp. 51-55).

[73] Deshpande, K. J. (2012). Agent Approach QoS Routing in MANET Based on Fuzzy Priority Scheduler. *International Journal of Electronics Signals and Systems.* Vol.1, Issue-2, (pp.76-81).

[74] Mhemed, R., Aslam, N., Phillips, W., & Comeau, F. (2012). An Energy Efficient Fuzzy Logic Cluster Formation Protocol in Wireless Sensor Networks. *The 3rd International Conference on Ambient Systems, Networks and Technologies*, (pp. 255 – 262).

[75] Yuste, A. J., Triviño, A., Trujillo, f. D., Casilari, E. (2010). Using Fuzzy Logic in Hybrid Multihop Wireless Networks. *International Journal of Wireless & Mobile Networks*. Vol.2, No.3, (PP. 96-108).

[76] Sharma, k. & Grag, D. (2009). Complexity Analysis Involving Heterogeneous System. *Computer and Information Science*, Vol. 2, No. 1, (pp. 48-52).

[77] Ki, Y. H., Ahn, S. C., & Kwon, W. H. (2000). Computational Complexity of General Fuzzy Logic Control and Its Simplification for A Loop Controller. *Fuzzy Sets and Systems, Elsevier Science*. (pp.215-224).

[78] Network Simulator, NS-2, and http://www.isi.edu/nsnam/-ns/.

[79] Issariyakul, T., & Hossain, E. (2009). Introduction to Network Simulator NS2. Springer Science, University of Manitoba. *Springer* New York Dordrecht Heidelberg London. DOI 10.1007/978-1-4614-1406-3.

[80] Bai, F., Sadagopan, N., & Helmy, A. (2003). IMPORTANT: A Framework to Systematically Analyze the Impact of Mobility on Performance of Routing

Protocols For Ad Hoc Networks. *IEEE INFOCOM*, (pp775-780)..

[81] Camp, T., Boleng, J., & Davies, V. (2002). A Survey of Mobility Models for Ad Hoc Network Research. *Wireless Communication & Mobile Computing, Trends and Applications*, Vol. 2, No. 5, (pp.483-502).

[82] Kurkowski, S., Camp, T., & Navidi, W. Minimal Standards for Rigorous MANET Routing Protocol Evaluation. Colorado School of Mines.

[83] Aschenbruck, Padilla, N. E., & Martini. P. (2008). A Survey on Mobility Models for Performance Analysis in Tactical Mobile Networks. *Journal of Telecommunications and Information technology*, (pp.54-61).

[84] Moses, G., Kumar, D., Varma, S., & Supriya, N. (2012). A Simulation Based Study of AODV, DSR, DSDV Routing Protocols in MANET Using NS-2. *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol.2, Issue 3, (pp.43-51).

[85] https://en.wikipedia.org/wiki/Preferred_walking_speed.

[86] P. Annamalai, P. (2005). Comparative Performance Study of Standardized Ad Hoc Routing Protocols and OSPF-MCDS. Blacsburg, Virginia.(pp.1-26)

[87] http://nile.wpi.edu/NS/simple_ns.html.

[88] Dougherty, C. (2001). Introduction to Econometrics. Third Edition, Oxford

University Press, Oxford.

[89] Christine, C., & Cheng, R. (2013). A Practical Introduction to Analysis of Simulation Output Data. University of Southampton, High field, Southampton, SO11BJ, UK.IEEE, (pp. 328-341).

[90] Kaur, B. P., & Aggrarwal, H. (2013). An Optimization of a Planning Information System Using Fuzzy Inference System and Adaptive Neuro-Fuzzy Inference System. *WSEAS Transactions On Information Science and Applications*, Vol. 10, Issue 8, (pp. 249-260).

[91] Taher, A. A. (2010). Adaptive Neuro-Fuzzy Systems. Electrical Communication & Electronics Systems Engineering department, *Modern Science and Arts* University, Egypt, (pp. 85-110).

APPENDICES

Appendix A: Statistical Consideration and Confidence Intervals

MANET is considered as a complex system which is very difficult to estimate its behaviors precisely at different operation environments. Multi-hop communication, asymmetrical time varying propagation channels, hidden and exposed terminals, lack of fixed infrastructure and central controller increase the MANET system complexity. Generally, many MANET parameters are mutually correlated each other's at the working conditions. Hence, statistical Analysis will be an important tool to analyze and calculate the simulation performance metrics of random MANET properties.

The confidence interval (CI) was used to calculate different simulation performance metrics. First, 20 random waypoint mobility scenario files generated for the calculations. The following formulas were used to calculate the confidence intervals:

$$\theta - t_{\alpha/2,n-1}\frac{S}{\sqrt{n}} \le \theta \le \theta + t_{\alpha/2,n-1}\frac{S}{\sqrt{n}} \quad (1)$$

Where Θ is the mean sample set, S is the sample standard deviation, n is the sample size. The mean sample set Θ is formulated by:

$$\boldsymbol{\theta} = \frac{1}{n}\sum_{i=1}^{n} Y_i \quad (2)$$

Where Y_i is the i^{th} sample from sample set.

The sample standard deviation S is expressed as:

$$S = \sqrt[2]{\sum_{i=1}^{n}\frac{(Yi-\theta)^2}{(n-1)}} \quad (3)$$

The term ($t_{\alpha/2\ ,\ n-1}\ *(\frac{S}{\sqrt{n}})$) represents the margin of error determining the upper and the lower bounds of the 100(1-α)% of the confidence intervals for the mean sample

value Θ. $t_{\alpha/2,n-1}$ denotes the upper α/2 percentage point of the t distribution. Then, any sample value Yi may lie between the confidence interval's (CI) ranges for each particular simulation run to satisfy the specific confidence level of 100(1-α)% [88], [89].

Appendix B: Modification of the Classical AODV Routing Protocol

The main aodv package files to satisfy the required improvement of route selection which have been modified are aodv.cc, aodv.h, aodv.packet.h, and aodv.rtable.h.

aodv.cc file modification

```
/*
Constructor
*/

/*******Add**************/
stability =0.0;
MobileNode *iNode;
iEnergy =0.0;
node_speed =0;
/**************************/

/*
  Packet Transmission Routines
*/

void
AODV::forward(aodv_rt_entry *rt, Packet *p, double delay) {
struct hdr_cmn *ch = HDR_CMN(p);
struct hdr_ip *ih = HDR_IP(p);

/******** Add***************/
iNode = (MobileNode *) (Node::get_node_by_address(index));
iEnergy = iNode->energy_model()->energy();
node_speed = iNode->speed();
//printf("at time(%.6f),position of %d is X:%.4f Y:%.4f
\n",CURRENT_TIME,index,xpos,ypos);
printf(" at time (%.6f):updatedd energy for node %d is energy %.4f
\n",CURRENT_TIME, index,iEnergy);
printf("velocity of %d, speed= %d \n", index ,node_speed);
/****************************/

/*
 * Cache the broadcast ID
 */

  if ((rq->rq_src_seqno > rt0->rt_seqno) ||
       ((rq->rq_src_seqno == rt0->rt_seqno) &&
        ((rq->rq_hop_count < rt0->rt_hops)||(rq->stability > rt0->stability)))) {
```

```
   // If we have a fresher seq no. or lesser #hops for the
   // same seq no., update the rt entry. Else don't bother.
rt_update(rt0, rq->rq_src_seqno, rq->rq_hop_count, ih->saddr(),rq->stability,
/*
  * Can't reply. So forward the Route Request
  */
/*******Add******************/

 forward((aodv_rt_entry*) 0, p, MY_TIMER_VALUE);

/*****************************/


/*
   Packet Transmission Routines
*/

/********** Add*****************/
iNode = (MobileNode *) (Node::get_node_by_address(index));
iEnergy = iNode->energy_model()->energy();
node_speed = iNode->speed();
//printf("at time(%.6f),position of %d is X:%.4f Y:%.4f
\n",CURRENT_TIME,index,xpos,ypos);
printf(" at time (%.6f):updatedd energy for node %d is energy %.4f
\n",CURRENT_TIME, index,iEnergy);
printf("velocity of %d, speed= %d \n", index ,node_speed);
/*******************************/


AODV::sendRequest(nsaddr_t dst) {
// Allocate a RREQ packet
/****************Add**************/
#include <aodv/fuzzylogic.c>


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

rq->v = iNode->speed();
rq->iEnergy = iNode->energy_model()->energy();
iEnergy = iNode->energy_model()->energy();
node_speed = iNode->speed();
double Resenergy = iEnergy;
double Velocity = node_speed;
double Hopcount = rq->rq_hop_count;


// Create function blocks
FunctionBlock_tipper tipper;
// Parse input
tipper.Resenergy = iEnergy;
```

```
tipper.Velocity = node_speed;
tipper.Hopcount = rq->rq_hop_count ;

// Calculate
stability =tipper.calc();
// Show results
tipper.print();

/*************************/
```

Aodv.h header file modifications

```
/*******Add**************/
#include <mobilenode.h>
#include <timer-handler.h>
/*************************/


/*******Add******************/
#define MY_TIMER_VALUE    (0.1 * 2 * NODE_TRAVERSAL_TIME *
NETWORK_DIAMETER)
/*****************************/

/*
       * Route Table Management
 */
/**********  add stability in rt-update********/
      void          rt_update(aodv_rt_entry *rt, u_int32_t seqnum,
                                u_int16_t metric, nsaddr_t nexthop, double
stability,
                                double expire_time);
/************************************/

/*
     * History management
 */

  /**********Add*****************/
      double stability;
      double iEnergy;
      int node_speed;
      MobileNode *iNode;
      /*********************************/
```

Packet.h header file modification

```
struct hdr_aodv_request {
```

```
/**************Add*********/
      double       v;
      double       iEnergy;
      double       stability;
/************************/
```

rtable.h header file modification

```
class aodv_rt_entry {

/**************Add**************/
      double       stability ;
/*******************************/
```

Appendix C: AWK Code for Evaluation of MANET Performance

```
BEGIN {
seqno = -1;
Rx_pkts = 0;
drop_Pkts = 0;
received_packets = 0;
 count = 0;
sent_packets = 0;
 ctrl_pkts = 0;
 PDR = 0;
sim_time = 300;
}

{
Time = $2
Event = $1
Pkt_size = $8
Level = $4
Node_id = $3
If ($4 == "AGT" &&$1=="s" && seqno < $6) {
Seqno = $6;
sent_packets++;
}
  else if(($4 == "AGT") && ($1 == "r")) {
 received_Packets++;
  }
     else if ($1=="d" && ($7=="tcp" || $7=="cbr")  &&  $8 > 512){
    drop_Pkts ++;
    }
      else if ($4=="RTR" && ($1=="s" || $1 == "f") && ($7 == "DSR" || $7 ==
"AODV" || $7 == "message")) {
    ctrl_pkts ++;
       }
if ($4 == "AGT"  &&  $1 == "r" && $8 >= 512) {
# packet header rip off
hdr_size = Pkt_size  %  512;
Pkt_size - = hdr_size;
# keep Rec. packet  size
Rxpkts += Pkt_size;
}
#Compute End-to-End delays
```

```
if($4 == "AGT"  &&  $1 == "s") {
start_time [$6] = $2;
}
else if (($1 == "r" &&($7 == "tcp"|| $7== "cbr") )) {
end_time [$6] = $2;
}
else if ( ($7 == "tcp" || $7 == "cbr") &&$1 == "d") {
end_time [$6] = -1;
}
}
END {
for(i=0; i<= sent_packets; i++) {
if(end_time[i] > 0) {
delay[i] = end_time [i] -  start_time [i];
count++;
}
else{
delay[i] = -1 ;
}
}
for(i=0; I <= seqno; i++) {
if (delay[i] > 0) {
n-to-n-delay = n-to-n-delay + delay [i];
} }
n-to-n-delay = n-to-n-delay / count;
print "Gen.Pkts = " seqno+1;
print "sent pkts = " sent_packets; print "Received Pkts = " received_packets;
PDR = received_packets/(sent_packets+1)*100
print "Pckt delivery ratio = " PDR" %";
print "Avag. end to  end delay = " n-to-n-delay*1000 " ms " ;
print" Total_Drop_pkts = " drop_Pkts;
print "Total Control pkts = " ctrl_pkts;
print "Avag. Routing Load = " ctrl_pkts /sim_time;
print "Average throughput (kbps) = "(Rxpkts * 8.0)/sim_time/1000;
}
```

Appendix D: Fuzzy Logic Inference Code for Node Trust Calculation

```
#include <stdlib.h>
#include <stdio.h>

double rAM_max(double def_Value, double
valueToAggregate)
{ return ( def_Value > valueToAggregate ? def_Value :
valueToAggregate ); }
double rAM_min(double degofsupport, double memship)
{ return (degofsupport < memship ? degofsupport : memship); }
double rCM_and(double antecedent1, double antecedent2)
{ return (antecedent1 < antecedent2 ? antecedent1 : antecedent2); }

//ruleAccumulationMethod_max = rAM_max
//def_Value =def_Value
//degofsupport =degofsupport
//memship =memship
//rAM_min =rAM_min
//rCM_and =rCM_and

class FunctionBlock_tipper {
public:
// VAR_INPUT
double Resenergy;
double Velocity;
double Hopcount;

// VAR_OUTPUT
double Stability;

private:
// FUZZIFY Res.energy
double Resenergy_low;
double Resenergy_med;
double Resenergy_high;

// FUZZIFY velocity
double Velocity_low;
double Velocity_med;
double Velocity_high;

// FUZZIFY Hopcount
```

```
double Hopcount_low;
double Hopcount_med;
double Hopcount_high;

// DEFUZZIFY Stability
double defuzzify_Stability[1000];

public:
FunctionBlock_tipper();
void calc();
void print();
private:
void defuzzify();
void fuzzify();
void reset();
double memship_Resenergy_low(double X);
double memship_Resenergy_med(double X);
double memship_Resenergy_high(double X);
double memship_Velocity_low(double X);
double memship_Velocity_med(double X);
double memship_Velocity_high(double X);
double memship_Hopcount_low(double X);
double memship_Hopcount_med(double X);
double memship_Hopcount_high(double X);
double memship_Stability_verylow(double X);
double memship_Stability_low(double X);
double memship_Stability_med(double X);
double memship_Stability_high(double X);
double memship_Stability_veryhigh(double X);


void calc_No1();
};

// Constructor
FunctionBlock_tipper::FunctionBlock_tipper() {
Stability = 0.0;
}

// Calculate function block
void FunctionBlock_tipper::calc() {
reset();
fuzzify();
calc_No1();
```

```
defuzzify();
}

// RULEBLOCK No1
void FunctionBlock_tipper::calc_No1() {
// RULE 1 : IF (((Resenergy is Low) AND (velocity is Low)) AND(Hopcount is Low)) THEN Stability IS Med;
double degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_low) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 2 : IF (((Resenergy is Low) AND (velocity is low)) AND(Hopcount is med)) THEN Stability IS Med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_low) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 3 : IF (((Resenergy is Low) AND (velocity is low)) AND(Hopcount is high)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_low) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
```

```
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 4 : IF (((Resenergy is Low) AND (velocity is med)) AND(Hopcount is low)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_med) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 5 : IF (((Resenergy is Low) AND (velocity is med)) AND(Hopcount is med)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_med) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 6 : IF (((Resenergy is Low) AND (velocity is med)) AND(Hopcount is high)) THEN Stability IS verylow;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_med) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
```

```
double memship = memship_Stability_verylow(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 7 :  IF (((Resenergy is Low) AND (velocity is high)) AND(Hopcount is low)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_high) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 8 : IF (((Resenergy is Low) AND (velocity is high)) AND(Hopcount is med)) THEN Stability IS verylow;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_high) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_verylow(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 9 : IF (((Resenergy is Low) AND (velocity is high)) AND(Hopcount is high)) THEN Stability IS verylow;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_low
 , Velocity_high) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
```

```
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_verylow(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 10 : iF (((Resenergy is med) AND (velocity is low)) AND(Hopcount is low)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_low) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 11 : IF (((Resenergy is med) AND (velocity is low)) AND(Hopcount is med)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_low) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 12 :  IF (((Resenergy is med) AND (velocity is low)) AND(Hopcount is high)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_low) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
```

```
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_maX(
defuzzify_Stability[M], y );
}
}
// RULE 13 : IF  (((Resenergy is med) AND (velocity is med)) AND(Hopcount is low)) THEN Stability IS med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_med) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 14 : IF  (((Resenergy is med) AND (velocity is med)) AND(Hopcount is med)) THEN Stability IS med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_med) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 15 :IF  (((Resenergy is med) AND (velocity is med)) AND(Hopcount is high)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_med) ,Hopcount_high) );
```

```
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[i] += rAM_max(
defuzzify_Stability[i], y );
}
}
// RULE 16 :IF  (((Resenergy is med) AND (velocity is high)) AND(Hopcount is low)) THEN Stability IS med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_high) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 17 : IF  (((Resenergy is med) AND (velocity is high)) AND(Hopcount is med)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
 , Velocity_high) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 18 : IF  (((Resenergy is med) AND (velocity is high)) AND(Hopcount is high)) THEN Stability IS low;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_med
```

```
 , Velocity_high) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_low(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 19 : IF  (((Resenergy is high) AND (velocity is low)) AND(Hopcount is low)) THEN Stability IS veryhigh;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_low) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_veryhigh(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 20 : IF  (((Resenergy is high) AND (velocity is low)) AND(Hopcount is med)) THEN Stability IS veryhigh;;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_low) ,Hopcount_med) );;
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_veryhigh(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 21 : IF  (((Resenergy is high) AND (velocity is low)) AND(Hopcount is high)) THEN Stability IS veryhigh;
degofsupport_1 = 1.0 * (
```

```
rCM_and(rCM_and(Resenergy_high
 , Velocity_low) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_veryhigh(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 22 : IF  (((Resenergy is high) AND (velocity is med)) AND(Hopcount is low)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_med) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 23 : IF  (((Resenergy is high) AND (velocity is med)) AND(Hopcount is med)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_med) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 24 : IF  (((Resenergy is high) AND (velocity is med)) AND(Hopcount is high)) THEN Stability IS high;
```

```
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_med) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 25 : IF  (((Resenergy is high) AND (velocity is high)) AND(Hopcount is
low)) THEN Stability IS high;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_high) ,Hopcount_low) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_high(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
// RULE 26 : IF  (((Resenergy is high) AND (velocity is high)) AND(Hopcount is
med)) THEN Stability IS med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_high) ,Hopcount_med) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
```

```
// RULE 24 : IF  (((Resenergy is high) AND (velocity is high)) AND(Hopcount is high)) THEN Stability IS med;
degofsupport_1 = 1.0 * (
rCM_and(rCM_and(Resenergy_high
 , Velocity_high) ,Hopcount_high) );
if( degofsupport_1 > 0 ) {
for (int M = 0 ; M < 1000 ; M++ ) {
        double X = 0.0 + M * 0.1;
double memship = memship_Stability_med(X);
double y = rAM_min( degofsupport_1
, memship );
defuzzify_Stability[M] += rAM_max(
defuzzify_Stability[M], y );
}
}
}
// Defuzzify and centroid calculation y=(SUMMMION of (stability value * memship values) / SUM of (memship values))

void FunctionBlock_tipper::defuzzify() {
double sum_Stability = 0.0;
double wsum_Stability = 0.0;
for (int M = 0; M < 1000 ; M++ ) {
double X = 0.0 + M * 0.1;
sum_Stability += defuzzify_Stability[M];
wsum_Stability += X * defuzzify_Stability[M];
}
Stability = wsum_Stability / sum_Stability;
}

// Fuzzify all variables
void FunctionBlock_tipper::fuzzify() {
Resenergy_low = memship_Resenergy_low(Resenergy);
Resenergy_med = memship_Resenergy_med(Resenergy);
Resenergy_high = memship_Resenergy_high(Resenergy);
Velocity_low = memship_Velocity_low(Velocity);
Velocity_med = memship_Velocity_med(Velocity);
Velocity_high = memship_Velocity_high(Velocity);
Hopcount_low = memship_Hopcount_low(Hopcount);
Hopcount_med = memship_Hopcount_med(Hopcount);
Hopcount_high = memship_Hopcount_high(Hopcount);
}
```

```
// memship input variables to fuzzy system functions
double FunctionBlock_tipper::memship_Resenergy_low(double X) {
if ( X <= 25.0 ) return 1.0;
if ( X > 50.0 ) return 0.0;
if ( X <= 50.0 ) return  ( ( 50.0 - X ) / (50.0 - 25.0 ) );
}

double FunctionBlock_tipper::memship_Resenergy_med(double X) {
if ( X <= 25.0 ) return 0.0;
if ( X > 75.0 ) return 0.0;
if ( X <= 50.0 ) return  ( ( X - 25.0 ) / (50.0 - 25.0 ) );
if ( X <= 75.0 ) return  ( ( 75.0 - X ) / (75.0 - 50.0 ) );
}
double FunctionBlock_tipper::memship_Resenergy_high(double X) {
if ( X <= 50.0 ) return 0.0;
if ( X > 75.0 ) return 1.0;
if ( X <= 75.0 ) return  ( ( X - 50.0 ) / (75.0 - 50.0 ) );

}


double FunctionBlock_tipper::memship_Velocity_low(double X) {
if ( X <= 1.0 ) return 1.0;
if ( X > 5.0 ) return 0.0;
if ( X <= 5.0 ) return  ( ( 5.0 - X ) / (5.0 - 1.0 ) );
}

double FunctionBlock_tipper::memship_Velocity_med(double X) {
if ( X <= 1.0 ) return 0.0;
if ( X > 20.0 ) return 0.0;
if ( X <= 5.0 ) return  ( ( X - 1.0 ) / ( 5.0 -1.0 ) );
if ( X <= 15.0 ) return 1.0;
if ( X <= 20.0 ) return  ( ( 20.0 - X ) / ( 20.0 -15.0 ) );
}

double FunctionBlock_tipper::memship_Velocity_high(double X) {
if ( X <= 15.0 ) return 0.0;
if ( X > 20.0 ) return 1.0;
if ( X <= 20.0 ) return  ( ( X - 15.0 ) / ( 20.0 -15.0 ) );
}


double FunctionBlock_tipper::memship_Hopcount_low(double X) {
if ( X <= 0.0 ) return 1.0;
```

```
if ( X > 4.0 ) return 0.0;
if ( X <= 4.0 ) return  ( ( 4.0 - X ) / (4.0 - 0.0 ) );
}

double FunctionBlock_tipper::memship_Hopcount_med(double X) {
if ( X <= 2.0 ) return 0.0;
if ( X > 8.0 ) return 0.0;
if ( X <= 4.0 ) return  ( ( X - 2.0 ) / (4.0 - 2.0 ) );
if ( X <= 6.0 ) return 1.0;
if ( X <= 8.0 ) return  ( ( 8.0 - X ) / (8.0 - 6.0 ) );
}
double FunctionBlock_tipper::memship_Hopcount_high(double X) {
if ( X <= 6.0 ) return 0.0;
if ( X > 8.0 ) return 1.0;
if ( X <= 8.0 ) return  ( ( X - 6.0 ) / (8.0 - 6.0 ) );
}

// memhership of output ( stability of node)

double FunctionBlock_tipper::memship_Stability_verylow(double X) {
if ( X <= 0.0 ) return 0.0;
if ( X > 25.0 ) return 0.0;
if ( X <= 12.5 )  ( ( X - 0.0 ) / (12.0 - 0.0 ) );
if ( X <= 25.0 ) return  ( ( 25.0 - X ) / (25.0 - 12.5 ) );
}
double FunctionBlock_tipper::memship_Stability_low(double X) {
if ( X <= 20.0 ) return 0.0;
if ( X > 40.0 ) return 0.0;
if ( X <= 30.0 ) return  ( ( X - 20.0 ) / (30.0 - 20.0 ) );
if ( X <= 40.0 ) return  ( ( 40.0 - X ) / (40.0 - 30.0 ) );
}
double FunctionBlock_tipper::memship_Stability_med(double X) {
if ( X <= 30.0 ) return 0.0;
if ( X > 60.0 ) return 0.0;
if ( X <= 45.0 ) return  ( ( X - 30.0 ) / ( 45.0 - 30.0 ) );
if ( X <= 60.0 ) return  ( ( 60.0 - X ) / (60.0 - 45.0 ) );
}
double FunctionBlock_tipper::memship_Stability_high(double X) {
if ( X <= 50.0 ) return 0.0;
if ( X > 80.0 ) return 0.0;
if ( X <= 65.0 ) return ( ( X - 50.0 ) / (65.0 - 50.0 ) );
if ( X <= 80.0 ) return  ( ( 80.0 - X ) / (80.0 - 65.0 ) );
}
double FunctionBlock_tipper::memship_Stability_veryhigh(double X) {
```

```
if ( X <= 70.0 ) return 0.0;
if ( X > 100.0 ) return 0.0;
if ( X <= 85.0 ) return  ( ( X - 70.0 ) / (85.0 - 70.0 ) );
if ( X <= 100.0 ) return  ( ( 100.0 - X ) / (100.0 - 85.0 ) );
}

// Print

void FunctionBlock_tipper::print() {

printf("Function block tipper:\n");
printf(" Input %10s : %f\n", "Resenergy" , Resenergy);
printf(" %20s : %f\n", "Resenergy_low" , Resenergy_low);
printf(" %20s : %f\n", "Resenergy_med" , Resenergy_med);
printf(" %20s : %f\n\n\n", "Resenergy_high" , Resenergy_high);
printf(" input %10s : %f\n", "Velocity" , Velocity);
printf(" %20s : %f\n", "Velocity_low" , Velocity_low);
printf(" %20s : %f\n", "Velocity_med" , Velocity_med);
printf(" %20s : %f\n\n\n", "Velocity_high" ,Velocity_high);
printf(" input  %10s : %f\n", "Hopcount" , Hopcount);
printf(" %20s : %f\n", "Hopcount_low" , Hopcount_low);
printf(" %20s : %f\n", "Hopcount_med" , Hopcount_med);
printf(" %20s : %f\n\n\n", "Hopcount_high" , Hopcount_high);


printf(" Output %20s : %f\n", "Stability" , Stability);
}
// Reset output
void FunctionBlock_tipper::reset() {
for( int M=0 ; M < 1000 ; M++ ) { defuzzify_Stability[M] = 0.0; }
}
int main() {
// Create function blocks
FunctionBlock_tipper tipper;
// Parse input

tipper.Resenergy = 80;
tipper.Velocity =1;
tipper.Hopcount =3;
// Calculate
tipper.calc();
// Show results
tipper.print();
}
```

Appendix E: TCL Code

```
#=====================================================
# Define simulation scenario parameters
#=============================================================
set val(chan)      Channel/WirelessChannel
set val(nn)        50
set val(ll)        LL
set val(mac)       Mac/802_11
set val(ant)       Antenna/OmniAntenna
set val(ifqlen)    50
set val(prop)      Propagation/TwoRayGround
set val(ifq)       Queue/DropTail/PriQueue
#set val(ifq)      CMUPriQueue  //used with DSR protocol only instead of line 8
set val(netif)     Phy/WirelessPhy
set val(rp)        AODV
set val(y)         900
set val(x)         900
set val(stop)      300

# Simulation instance creation
set ns_ [new Simulator]

# define the objects of ns and animator
set tracefd      [open out50-s30-p20-t300-900.tr w]
# set namtrace     [open outaodv.nam w]

$ns_ trace-all $tracefd
#$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)


# set up topography object
set topo      [new Topography]
# determine network topology
$topo load_flatgrid $val(x) $val(y)


# generating the GOD
#create-god $val(nn)
set god_ [create-god $val(nn)]

set chan_1_ [new $val(chan)]
```

```
set chan_2_ [new $val(chan)]

# Configure nodes
      $ns_ node-config -adhocRouting $val(rp) \
                  -ifqType $val(ifq) \
                  -llType $val(ll) \
                  -macType $val(mac) \
                  -ifqLen $val(ifqlen) \
                  -phyType $val(netif) \
                  -antType $val(ant) \
                 -propType $val(prop) \
                 -channelType $val(chan) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
                 -macTrace OFF \
                 -movementTrace OFF
 # Model of energy
    $ns_ node-config  -energyModel EnergyModel \
                    -initialEnergy 100 \
                    -txPower 35.2e-2 \
                    -rxPower 31.32e-2\
                    -idlePower 712e-5 \
                    -sensePower 144e-9


for {set i 0} {$i < $val(nn) } { incr i } {
          set node_($i) [$ns_ node]
         # $node_($i) random-motion 0            ;   # random moving disable
    }

# pattern of mobility loading command
puts "Loading nodes mobility scinaerio..."
source mob50-s30-p20-t300-900

# connect node_(1) and node_(5)
set tcp1 [new Agent/TCP]
$tcp1 set class_ 2
$tcp1 set window_ 8
$tcp1 set packetSize_ 512
set sink1 [new Agent/TCPSink]
$ns_ attach-agent $node_(5) $sink1
$ns_ attach-agent $node_(1) $tcp1
$ns_ connect $tcp1 $sink1
```

```
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns_ at 0.0 "$ftp1 start"
$ns_ at 300.0 "$ftp1 stop"


# determine the nodes starting location in namitor
# determine nodes size in animator, it should defined after definition of mobility model
# puts "Processing node $i"

for {set i 0} {$i < $val(nn)} {incr i} {
$ns_ initial_node_pos $node_($i) 30
}

#### provide Labels for nodes

 $ns_ at 0.0 "$node_(1) label Source1"
     $ns_ at 0.0 "$node_(5) label Destination1"
     #Setting Color For Server
     $node_(1) color orange
     $ns_ at 0.0 "$node_(1) color orange"
     $node_(5) color green
     $ns_ at 0.0 "$node_(5) color green"


# inform the nodes that the simulation time is over
for {set i 0} {$i < $val(nn) } {incr i} {
$ns_ at $val(stop).0 "$node_($i) reset";
}

$ns_ at $val(stop).0 "stop"
$ns_ at $val(stop).0 "puts \" Exit simulating...\" ; $ns_ halt"

proc stop {} {
   global tracefd  ns_  ;
# namtrace
   $ns_ flush-trace
   close $tracefd

   # run nam outaodv.nam &
}
```

```
puts "Starting Simulation..."
puts $tracefd "M 0.0  rp $val(rp)  y $val(y) x $val(x)  nn $val(nn)  stop $val(stop)"
$ns_ run
```

Appendix F: Node Trust Value Comparison for Different Number of Membership Functions.

In order to justify the effects of different number of membership functions for each Fuzzy Inference System, Two, Three, and Four triangular-trapezoid membership functions are selected for each crisp input variables of our Fuzzy Inference System. Figures F1 and F2 show the two and four membership functions used in this justification study, where the three membership functions are appeared in Figure 5.6.

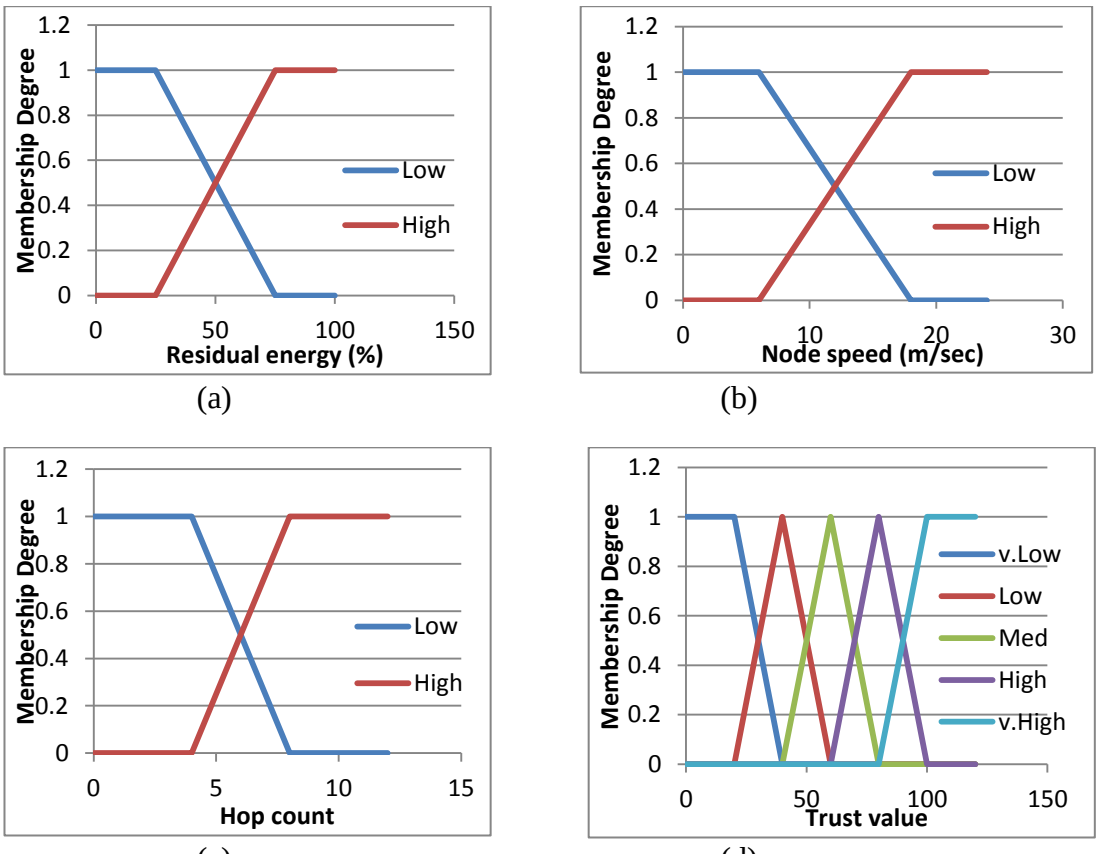


Figure F1: Two membership functions for each crisp input variable

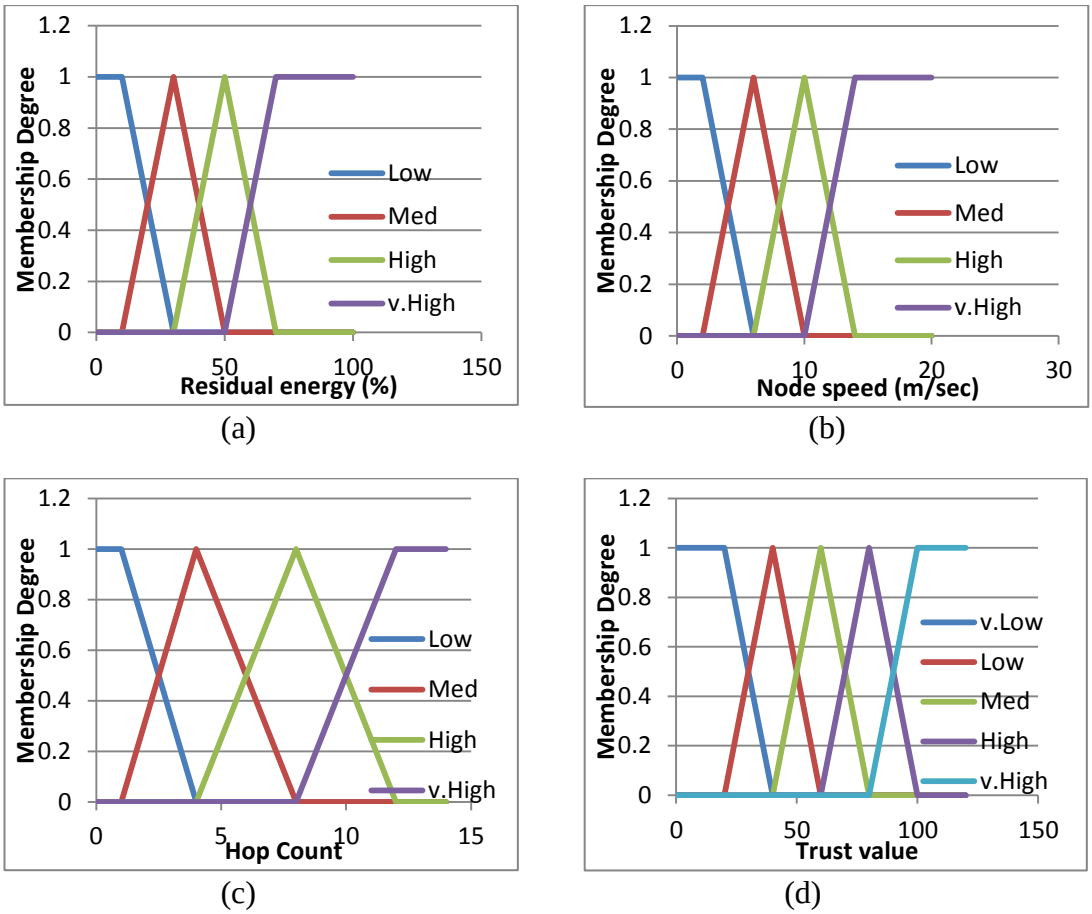


Figure F2: Four membership functions for each crisp input variable

The IF-THEN based rules used for inference engine for Two, Three, and Four membership functions are demonstrated in the following tables.

Table F1: TF-THEN rule base for two membership function for each crisp input

Inputs			**Output**
Residual energy	**Node speed**	**Hop count**	**Trust value**
Low	Low	Low	**Med**
Low	Low	High	**Low**
Low	High	Low	**Low**
Low	High	High	**v. Low**
High	Low	Low	**v. High**
High	Low	High	**High**
High	High	Low	**Med**
High	High	High	**Low**

Table F2: TF-THEN rule base for three membership function for each crisp input

Inputs			Output	Inputs			Output	Inputs			Output
Residual energy	**Node speed**	**Hop count**	**Trust value**	**Residual energy**	**Node speed**	**Hop count**	**Trust value**	**Residual energy**	**Node speed**	**Hop count**	**Trust value**
Low	Low	short	**Med**	Med	Low	short	**High**	High	Low	short	**v. High**
Low	Low	Med	**Med**	Med	Low	Med	**High**	High	Low	Med	**v. High**
Low	Low	Long	**Low**	Med	Low	Long	**High**	High	Low	Long	**v. High**
Low	Med	short	**Low**	Med	Med	short	**Med.**	High	Med	short	**High**
Low	Med	Med	**Low**	Med	Med	Med	**Med.**	High	Med	Med	**High**
Low	Med	Long	**v. Low**	Med	Med	Long	**Low**	High	Med	Long	**High**
Low	High	short	**Low**	Med	High	short	**Med.**	High	High	short	**High**
Low	High	Med	**v. Low**	Med	High	Med	**Low**	High	High	Med	**Med**
Low	High	Long	**v. Low**	Med	High	Long	**Low**	High	High	Long	**Med**

Table F3: TF-THEN rule base for four membership function for each crisp input

Inputs			Output	Inputs			Output	Inputs			Output
Residual energy	**Node speed**	**Hop count**	**Trust value**	**Residual energy**	**Node speed**	**Hop count**	**Trust value**	**Residual Energy**	**Node peed**	**Hop count**	**Trust value**
Low	Low	Low	**Med**	Med	Med	High	**Med**	High	v.High	Low	**Med**
Low	Low	Med	**Med**	Med	Med	v.High	**Low**	High	v.High	Med	**Med**
Low	Low	High	**Low**	Med	High	Low	**Med**	High	v.High	High	**Low**
Low	Low	v.High	**v.Low**	Med	High	Med	**Low**	High	v.High	v.High	**Low**
Low	Med	Low	**Low**	Med	High	High	**Low**	v.High	Low	Low	**V.High**
Low	Med	Med	**Low**	Med	High	v.High	**Low**	v.High	Low	Med	**V.high**
Low	Med	High	**v. Low**	Med	v.High	Low	**Low**	v.High	Low	High	**v. high**
Low	Med	v.High	**v.Low**	Med	v.Hgih	Med	**Low**	v. High	Low	v.High	**High**
Low	High	Low	**Low**	Med	v.Hgih	High	**v. Low**	v.High	Med	Low	**High**
Low	High	Med	**Low**	Med	v.High	v.High	**v. Low**	v.High	Med	Med	**V.high**
Low	High	High	**v. Low**	High	Low	Low	**High**	v.High	Med	High	**v. high**
Low	High	v.High	**v.Low**	High	Low	Med	**High**	v. High	Med	v.High	**High**
Low	v.High	Low	**v. Low**	High	Low	High	**Med**	v.High	High	Low	**High**
Low	v.High	Med	**V.low**	High	Low	v.High	**Med**	v.High	High	Med	**High**
Low	v.High	High	**v.Low**	High	Med	Low	**Med**	v.High	High	High	**Med**
Low	v.High	v.High	**v.Low**	High	Med	Med	**Med**	v. High	High	v.High	**low**

Med	Low	Low	**Med**	High	Med	High	**Low**	v.High	v. High	Low	**med**
Med	Low	Med	**Med**	High	Med	v.High	**Low**	v.High	v.High	Med	**Meh**
Med	Low	High	**Med**	High	High	Low	**Med**	v.High	v.High	High	**Low**
Med	Low	v.High	**Low**	High	High	Med	**Med**	v. High	v.High	v.High	**Low**
Med	Med	Low	**Med**	High	High	High	**Low**				
Med	Med	Med	**Med**	High	High	v.High	**Low**				

MATLAB 7.6.0 (version R2008a) under Window 8 operating system and Intel Core i7, 2.4 GHz, 64 bits processor was used to run our trust value calculations. Table F4 shows node trust value for different number of trapezoid-triangular membership functions applied.

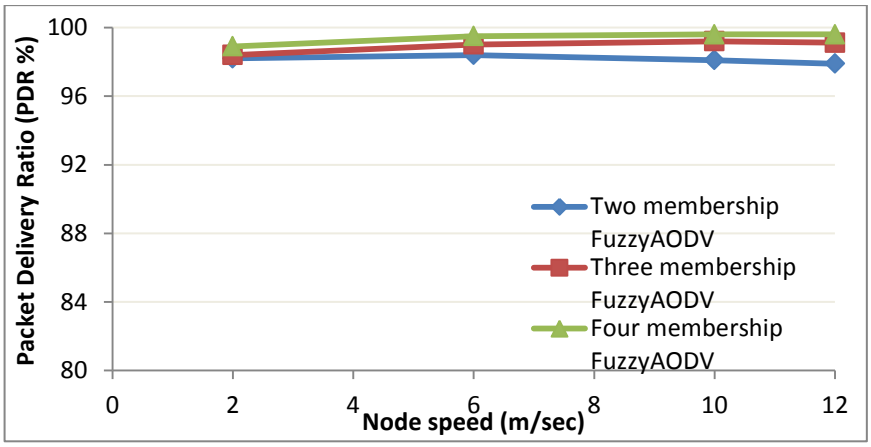
It is noted that the three membership functions for each crisp input variables have a good resolution compared to the other two sets of two and four membership functions. The three membership function set has lower complexity computations when compared with the four membership function set. So, three membership functions have been used in our simulation model of Fuzzy Inference System.

Table F4: Node trust value for different number of membership functions used

Inputs			**outputs**		
Residual Energy (%)	**Node Speed (m/sec)**	**Hop Count**	**Trust value of three inputs with two membership functions**	**Trust value of three inputs with three membership functions**	**Trust value of three inputs with four membership functions**
2	2	1	60	54.2	60
8	2	2	60	53.1	60
10	4	2	60	48.9	50.9
6	12	6	37.3	40	27.3
12	20	3	40	23.9	18.8
15	3	12	40	27.3	21.6
11	7	10	37.5	15.2	22
18	5	6	50	40	36.1
22	2	6	50	54.2	51
26	14	3	49	41.5	39.7

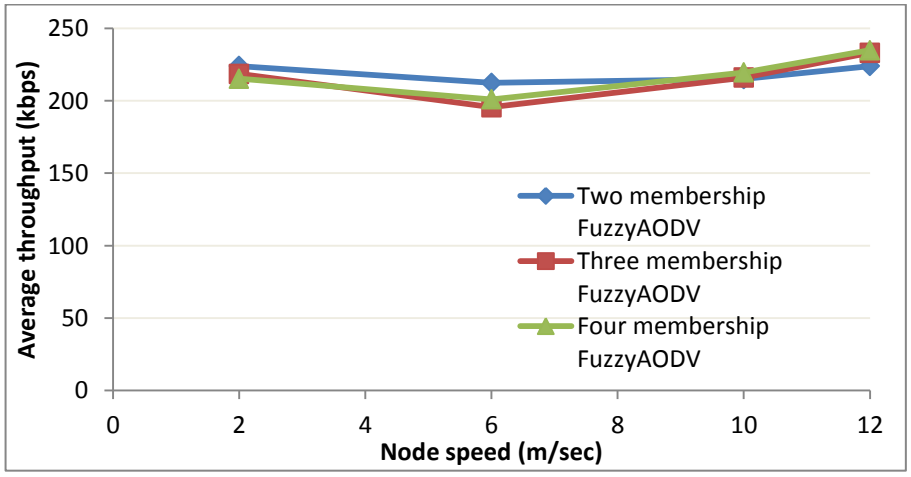
34	2	2	70.9	61.9	65.2
37	8	3	69.1	49.1	52
42	18	10	23.6	31.5	29
44	1	1	78.9	74.4	72.9
47	6	8	58.4	36.5	44.3
51	12	4	72.2	61.1	61.5
57	3	7	70.9	67.7	72.8
62	15	3	67.5	69.8	60
66	4	2	95.3	79.2	87.6
72	2	4	99.9	95	99.9
74	12	8	59.3	77.1	53.3
78	5	3	99	80	96.9
82	3	6	92.7	92	99
85	2	6	92	98.5	99.9
88	6	3	89.9	80	97.8
89	16	5	64 8	74.2	54.2
90	4	2	98	88.1	93.8
92	12	6	72.7	80	62.8
98	5	7	86 8	96.1	99.9
100	2	4	98.5	98.5	99.9
99	8	6	82.2	80	85.2
67	13	3	75.8	72	69.9
88	5	8	80	80	99
54	6	6	74.4	64	62.1
43	4	3	78.2	59.4	70.9
32	6	4	69	46.3	61.3
28	9	2	58.5	44	50.6

Figure F3 below shows the effects of using different number of membership functions on the Fuzzy AODV routing protocol. The simulation scenarios were simulated under node density of 50 nodes and pause time set to 20 sec.



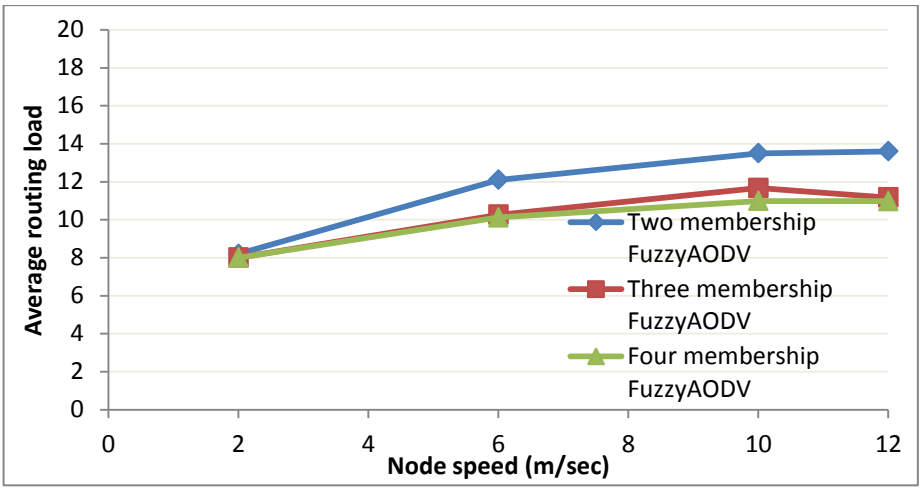
(a)

As illustrated in figure F3(a) when node speed increases packet delivery ratio of Fuzzy AODV with four membership functions and three membership functions converges to each other and shows an increasing percentage where the packet delivery ratio of Fuzzy AODV with two membership functions decreases.



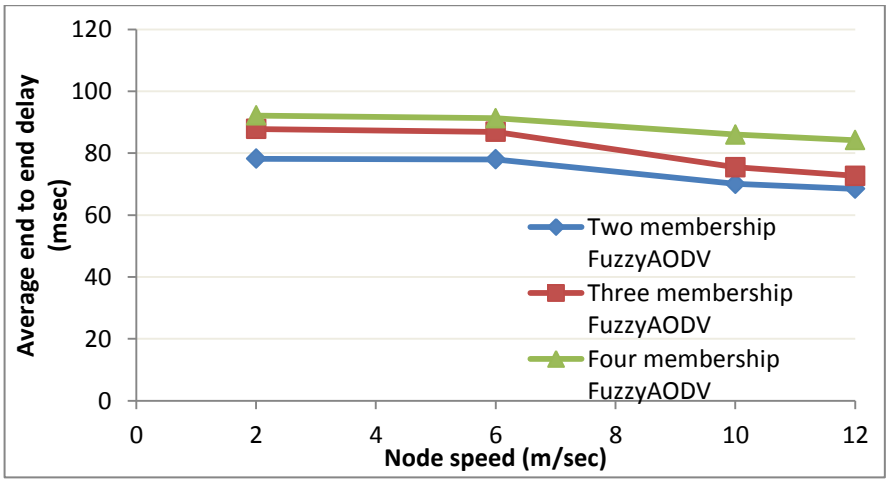
(b)

Figure F3 (b) presents the throughput of Fuzzy AODV under various node speeds. With three membership functions and four membership functions of Fuzzy AODV shows similar and increasing behavior while Fuzzy AODV with two membership functions has lower throughput under increasing node speeds.



(c)

Figure F3(c) demonstrates the average routing load under different node speeds. Fuzzy AODV with two membership function has the worst routing load, while three and four membership Fuzzy AODV shows almost the same routing load.



(d)

Figure F3: Fuzzy AODV with different number of membership performances vs. node speeds (Human speed) with 50 nodes and pause time of 20 sec

Four membership function Fuzzy AODV has highest end to end delay when compared to Fuzzy AODV with two and three membership functions as presented in figure F3 (d).

Appendix G: Train and Test Phases of the Fuzzy Logic System

After determining the inputs and output variables of our Fuzzy Inference System train and test dataset tables of 50 rows for each has been constructed as shown in Table G1. Three input variables of Fuzzy Inference System are residual energy, node speed and hop count and one output of trust value with five membership functions was used. Each value as train and test values were selected randomly. The designed interference mechanism uses min. (AND) logic operator to combine the input variables to construct the rule base of inference system.

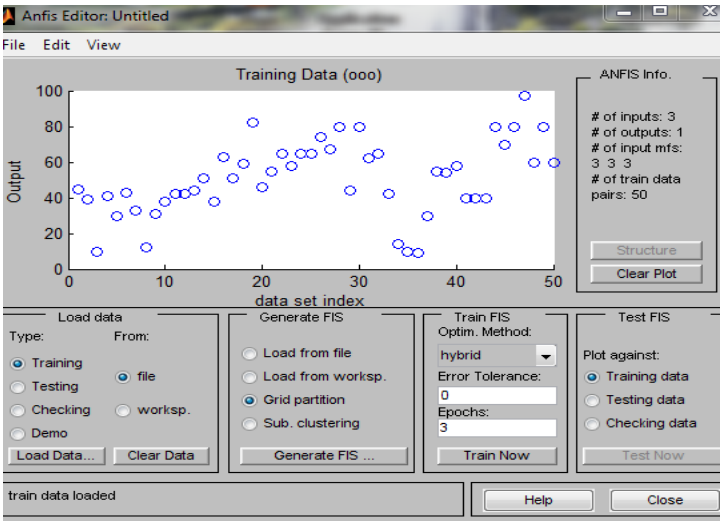
Table G1: Training and testing datasets table for ANFIS

Training Dataset				**Testing Dataset**			
Input1 **Residual energy (%)**	**Input2** **Node speed (m/s)**	**Input3** **Hop count**	**Output** **Trust value (%)**	**Input1** **Residual energy (%)**	**Input2** **Node speed (m/s)**	**Input3** **Hop count**	**Output** **Trust value (%)**
8	1	1	**45**	97	30	12	**60**
5	4	5	**39**	98	2	1	**98**
12	8	12	**10**	78	11	5	**80**
21	3	4	**41**	43	12	3	**54**
23	20	2	**30**	20	4	1	**40**
32	3	1	**43**	20	12	4	**42**
36	10	3	**33**	22	14	9	**16**
35	16	8	**12**	20	20	2	**40**
41	3	14	**31**	23	24	6	**16**
46	3	14	**38**	12	2	1	**55**
48	7	4	**42**	13	4	5	**45**
51	6	7	**42**	8	1	8	**39**
56	18	3	**44**	22	20	6	**15**
62	10	5	**51**	24	18	10	**14**
66	20	12	**38**	34	1	2	**66**
70	3	3	**63**	36	2	6	**66**
62	13	6	**51**	40	3	9	**54**
68	4	8	**59**	45	6	2	**52**
77	2	3	**82**	67	2	5	**82**
70	20	3	**46**	73	2	8	**98**
69	6	6	**55**	76	12	3	**78**
79	14	12	**65**	79	14	5	**69**
72	10	4	**58**	81	14	10	**81**
82	6	9	**65**	67	2	6	**84**
86	16	4	**65**	87	3	3	**97**
88	4	8	**74**	90	22	6	**65**
91	13	6	**67**	95	26	11	**60**
96	2	2	**80**	97	3	2	**99**
62	8	10	**44**	89	3	2	**97**
99	2	12	**80**	12	3	2	**50**
89	16	4	**62**	11	5	6	**47**

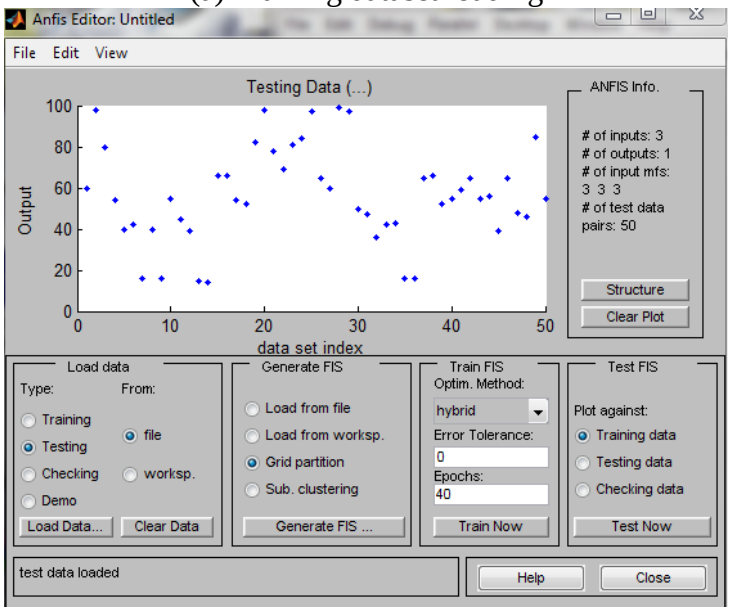
82	12	10	**65**	8	1	6	**36**
55	18	4	**42**	20	6	2	**42**
35	8	24	**14**	22	10	5	**43**
20	20	16	**10**	23	12	10	**16**
8	7	14	**9**	22	18	12	**16**
26	14	6	**30**	32	2	2	**65**
40	2	9	**55**	40	2	5	**66**
44	5	1	**54**	42	2	6	**52**
49	11	5	**58**	45	6	2	**55**
8	0	8	**40**	51	12	6	**59**
18	5	1	**40**	55	20	2	**65**
20	12	4	**40**	44	6	2	**55**
77	16	1	**80**	52	10	4	**56**
79	17	6	**70**	54	6	7	**39**
82	15	10	**80**	54	18	1	**65**
88	2	4	**97**	59	19	6	**48**
92	24	6	**60**	60	6	8	**46**
77	16	1	**80**	61	3	3	**85**
92	24	6	**60**	45	6	2	**55**

G1: Training of Adaptive Neuro-Fuzzy Inference System (ANFIS)

ANFIS is a multilayer adaptive network based on Fuzzy Inference System which learns and tune parameters of the FIS. Random dataset was used for training the ANFIS model. ANFIS model have four stages of loading data, generating FIS, Training FIS, and Testing FIS. At first, the dataset loaded to MATLAB 7.6.0 (version R2008a) ANFIS loading stage as a training dataset. These dataset includes three columns (residual energy, node speed, and hop count) and 50 rows and it have an output response as shown in Figure G1(a). And the testing dataset is shown in Figure G1(b).



(a) Training dataset loading



(b) Testing dataset loading

Figure G1: ANFIS loading stage

G2: Generating and training FIS stages

To generate FIS for the dataset loaded to an ANFIS, grid partition was after determining the number of membership functions and their types (Three membership functions have been selected with triangular types and one output selected with constant membership function type). The ANFIS model generated will be as shown in Figure G2 that has been developed by ANFIS grid partitioning method.

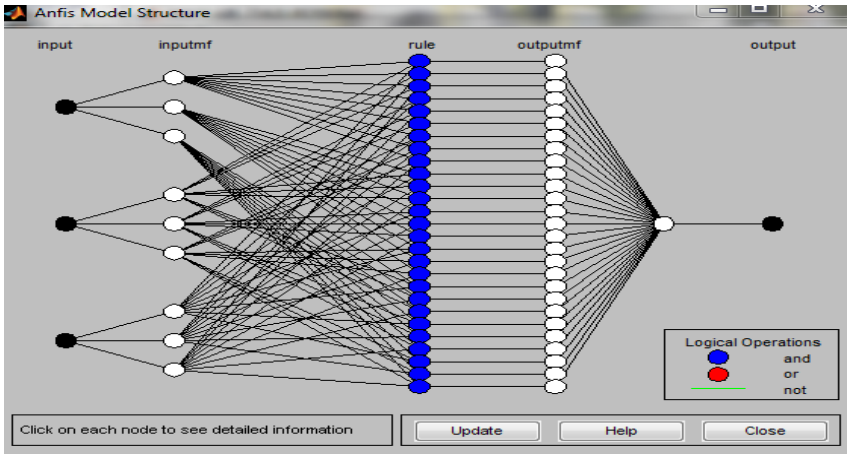


Figure G2: Structure of ANFIS designing model

The rules are generated by ANFIS is shown in Figure G3.

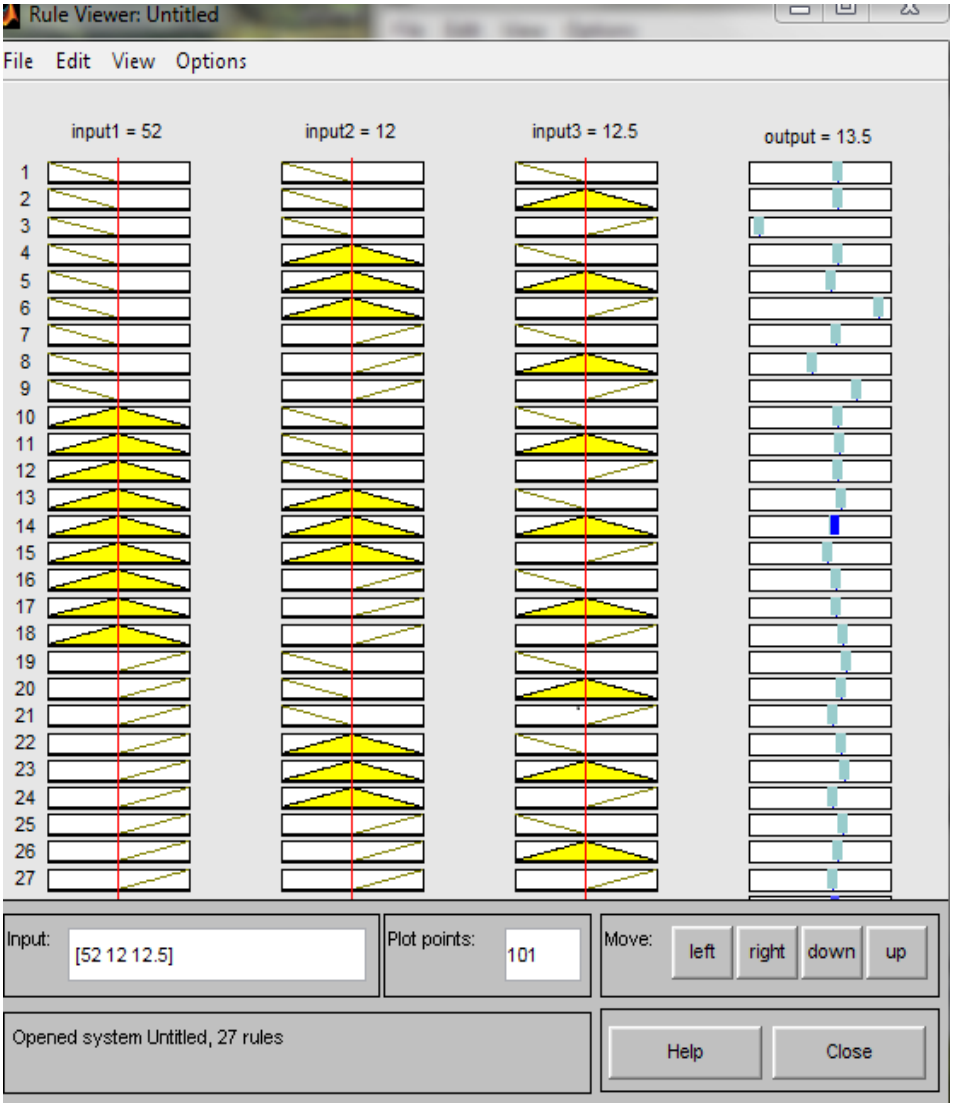


Figure G3: Snapshot of rule viewer of FIS generated by ANFIS

The ANFIS model uses hybrid optimization method to train the membership parameters following the training data as shown in Figure G4. For this model, the number of training epochs is set to 40 and the training error tolerance is set to zero.

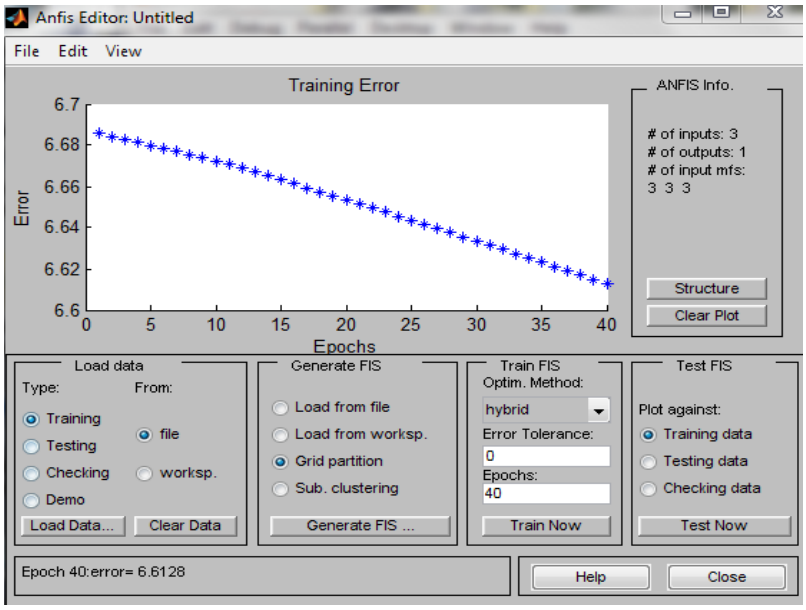


Figure G4: Snapshot of training error

After FIS training, the ANFIS model validated via testing data. Average testing errors of training, and testing data in the ANFIS model are presented in Figures G5 and G6 respectively [90] [91].

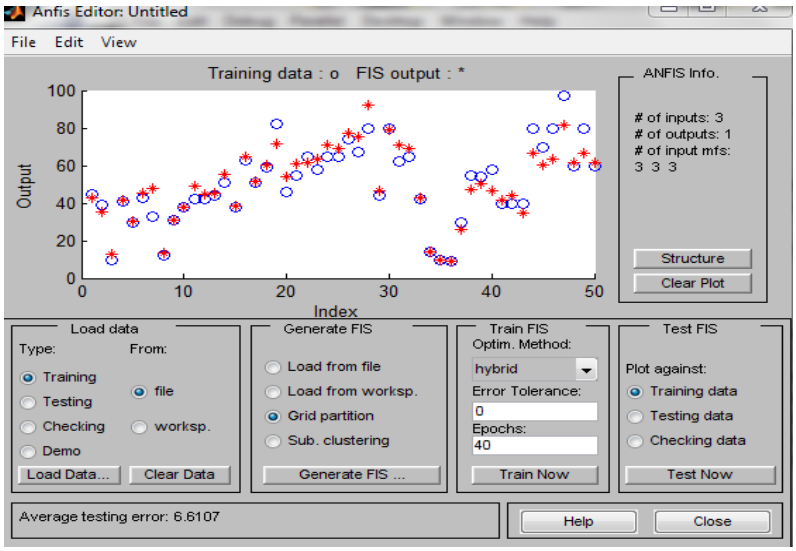


Figure G5: Trust node output of ANFIS with training data

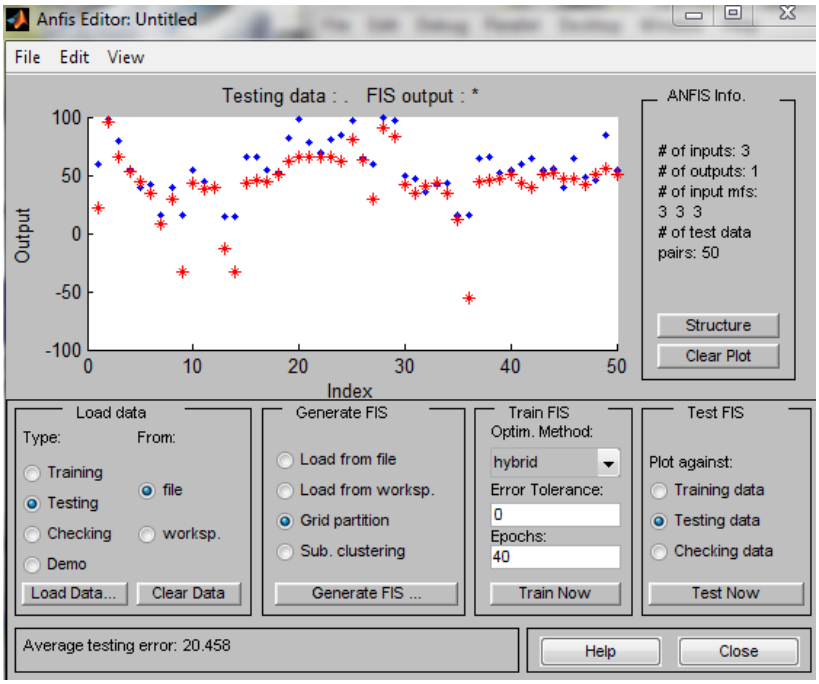


Figure G6: Trust node output of ANFIS with testing data