FRESNO STATE
Discovery. Diversity. Distinction.

California State University, Fresno

Electrical and Computer Engineering

ECE 298

Final Project (Spring 2019)

Instructor Dr. Nan Wang

Project Report on

DESIGN AND IMPLEMENTATION OF CONGESTION CONTROL IN DSR ROUTING PROTOCOL FOR MANET

By HARSHDEEP JHAJJ (109890371)

A project submitted for ECE 298 course work in fulfillment of the requirements for the degree of Masters of Science in Computer Engineering

California State University, Fresno

Spring 2019

ACKNOWLEDGEMENT

I would like to express my gratitude to my advisor Dr. Nan Wang for his support, expert guidance, understanding and encouragement throughout my study. I sincerely thank my committee members, Dr. Reza Raeisi and Dr. Hayssam El-Razouk for giving their valuable time on my project course work.

I want to thanks whole ECE department, who help in gaining the knowledge so that I can complete my studies. I would also like to thank my family members to support me in every phase of life.

ABSTRACT

Mobile Ad-hoc Network is a self-configured network, in which communication takes place wirelessly. It is a collection of nodes which establish the communication without any centralized access point. Each node act as source and destination. These nodes help to establish the communication over wireless network.

Dynamic Source Routing is one of the reactive protocols. In this protocol communication is established whenever source wants to communicate with the destination. It is on-demand source routing. While sending the Route Request Packet (RREQ), the intermediate nodes are too busy to transmit the date, so the load increased on these nodes.

There are other nodes available that are not utilized properly. The bandwidth in the available network in not fully utilized. To overcome this problem, a congestion counter is added in the RREQ packet and Route Reply Packet (RREP) to check the congestion on the current node.

All the simulations are carried out in Network Simulator 3 (NS3). NS3 is simulation tool for the network engineer. All the wired and wireless network can be designed using the NS3. A wireless model can be designed in the NS3. It is easier for the user to establish the network and make the desired changes. Using this simulator, performance is evaluated based on- packet loss, packet delivery ratio, throughput and end-end delay. Overall performance of DSR is increased in all four aspects.

Table of content

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1. Background

Mobile Ad-hoc Networks are a type of multi-hop network. In this network information is exchanged by using wireless transmission between the nodes. As these nodes are freely to move, the network coverage can be extended. These networks can work without any centralized point and network is fully distributed. These nodes have less cost and more flexibility as compared to the wired networks. The major application of MANETs are rescue operations like maintain an information between the soldiers and head base. It is also used in personal area network, commercial sector, wearable devices.

Each node in the MANETs can act as transfer node and receiver node. A node can be source and destination at same time. The nodes are mobile and can move freely so topology can be varied based on mobility of nodes.

All the nodes in the Ad-hoc network are mobile and use wireless medium for communication. All the packets are sent and receive through wireless channel. Any link failure and topology changed is fully adaptive to the system. The protocols are used for the communication is also provide the energy consumption, bandwidth and power consumption of the network. [1]

Node mobility is the one of the major challenges faced in the MANETs. As the mobility of nodes increase, it results to more link breaks, energy consumption and density also increased. The increased nodes density affects the throughput of the network. Nodes also consumed energy while sending, receiving and discarding the packet data. Energy consumption further increased, when more packets are used for route discovery and topology organization.

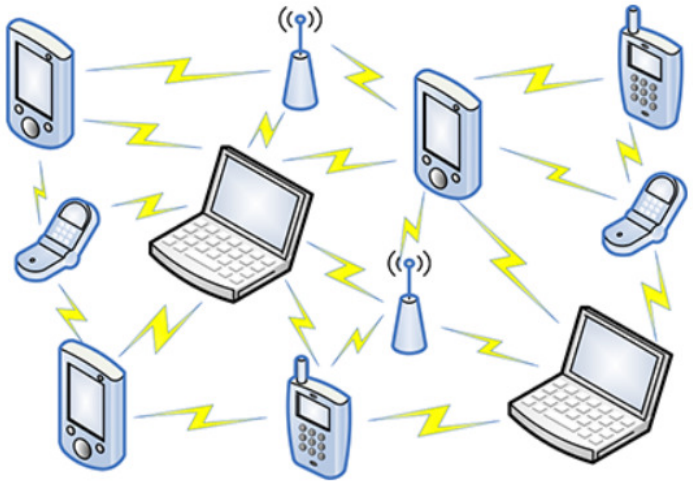


Fig.1.1 Mobile AD-Hoc Network

1.2 Characteristics of MANETS

1. Distributed operation- Each node in the networks act as source and destination. It is not controlled by any single centralized network. All these nodes establish a communication network between each other.
2. Multi-hop – Each node in MANETs have different range and intermediate node. Each node has small range to send the data packets. Node "A" wants to send data to Node "E", but don't have direct link between two nodes. So, communicate with the help of intermediate nodes.
3. Dynamic- In MANETs all the nodes are freely to move with different speed. So, the network topology changes with time.
4. Autonomous- Mobile nodes are not dependent and can act as both source and destination.

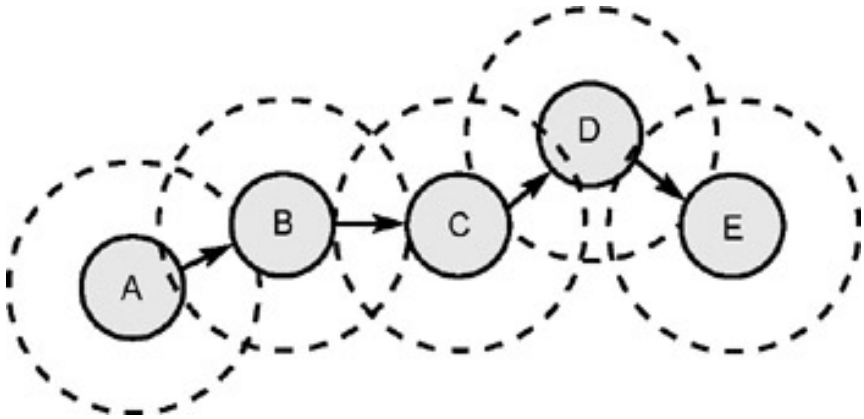


Fig 1.2 Network Connection

The above figure shows the network connection of Mobile Ad-hoc network these nodes communicate with each other wirelessly. Node "A" wants to communicate with node "E". Each node has small transmission range, So node cannot send data packets directly. The intermediate nodes "B", "C" and "D" will be used to transfer the data packets from node "A" to node "E".

1.3 Applications of MANET

1. Defense- MANET can be use in military applications. Military personnel can use this technology to maintain information between the soldiers and the vehicles.
2. Rescue Operation- Mobile Ad-hoc network can be used in disaster areas for the rescue operations. For example- Earth Quake, fire etc
3. Personal Area Network- In short range network such as personal area network, AD-hoc network can be used.

Mobile Ad-hoc network is used in various applications like military, commercial, emergency and safety. As the name suggests, it is the multi-hop network. All the nodes act as source and destination. On the other hand, single-hop network make connection between base station and mobile station which are conventional cellular network.

In single-hop network, it uses the fixed base network. On the other hand, in Mobile Ad-Hoc Network, no node is fixed. In MANET the throughput and efficiency increased as compare to the single-hop network. Since the transmission is carried out on the short links, so large amount of energy is consumed. The major feature of multi-hop network is that node can communicate directly to each other without any fixed structure. So, MANET can be used, where a fixed structure is not

available. For example, in disaster area, where all the infrastructure destroyed, MANET can be implemented in those areas.

1.4 Routing Protocols

The process of sending the data packets or information from source to a destination is called routing. Routing is used to find the optimal routing path from source to destination and transfer the data packets through the internetwork.

There are several methods are used by the routing protocol to send the data packet from source to destination. Routing algorithms are used to find the route from source to destination. These algorithms used the optimal path for the route of the packet. Routing algorithms maintain the information of all the nodes and its neighbor in the routing table. Different routing algorithms used different method to reach out the destination.

Routing table contain variety of information including the IP-Address prefix, next hop, destination sequence number etc. Different routing protocol also use more fields in the communication.

For MANETs, routing protocols are defined into three types

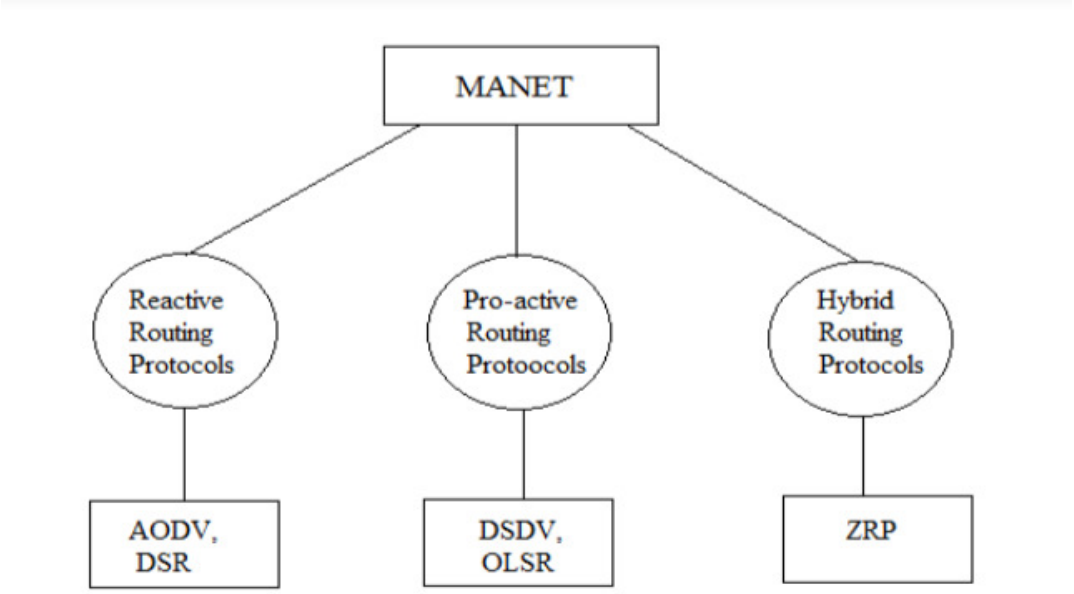


Fig 1.3. Routing Protocol

1.4.1 Reactive Protocol

In reactive protocol, route discovery process is initiated when source wants to communicate with the destination. This is also called on-demand routing protocol. In this protocol source sends the route request packet in order to find the destination. Nodes don't make any routing information if there is no communication. A route mechanism will find a route by searching for the route in order to establish a connection to transmit.

Each node has list of other nodes which it has direct link. Nodes also record the time of neighbor discovery in the neighbor list. This time of discovery is not changed even during the re-discovery of the existing neighbors. Time of discovery is recorded only once for the new neighbor is discovered. Each node sends "Hello" packet to its neighboring nodes in order to maintain neighbor list. Nodes the receives "Hello" packets are required to respond back quickly.

Dynamic Source Routing (DSR) and Ad-hoc On-Demand Distance Vector routing (AODV) are examples of reactive routing protocols.

1.4.2 Proactive Routing

In this protocol, every node of a network has a routing table to store the routing information. If the is change in network due to any reason, this change will be updated in the table. The size of network is increased as the no of nodes increased. This causes the overhead in network and proactive protocol makes it easy to detect the overhead in the network. The amount of delay experienced while sending data packets reduced by finding the destination path immediately.

It increases the amount of topology information stored at each node which avoids loops and speeds up the protocol. It dynamically varies the size of route updates. In proactive protocols bandwidth consumption and delay in increased.

Destination Sequenced Distance Vector (DSDV) routing and Fisheye State Routing (FSR)

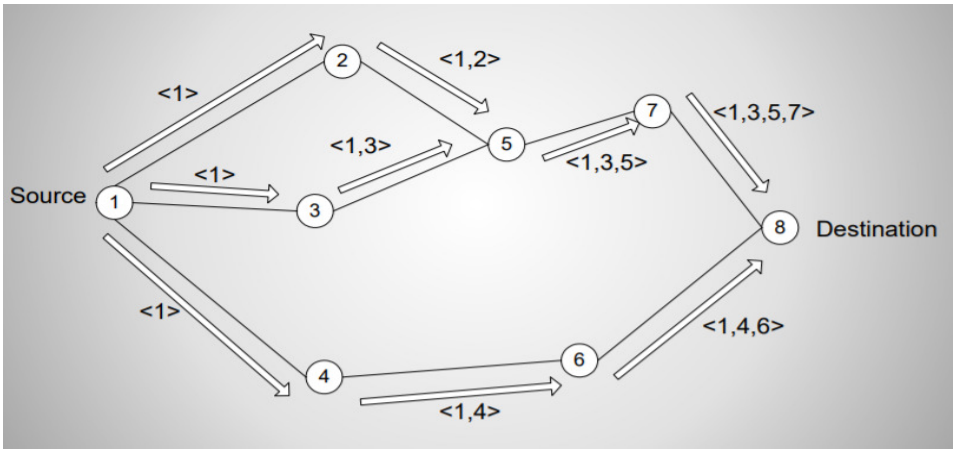
1.4.3 Hybrid Routing

Hybrid routing is combination of both proactive and reactive protocol. In hybrid, the node with proximity work together in order to reduce the route discovery overhead. This procedure maintains the scalability of the network. The nearby nodes use proactive technique to maintain routes and reactive technique for the far away nodes using route discovery strategy. Hybrid protocols are zone-based protocols which means network is considered as number of zones by each node. Zone routing protocol, Temporarily Ordered Routing Algorithm (TORA) are example of hybrid routing protocol.

2. Related Work

2.1 Dynamic Source Routing

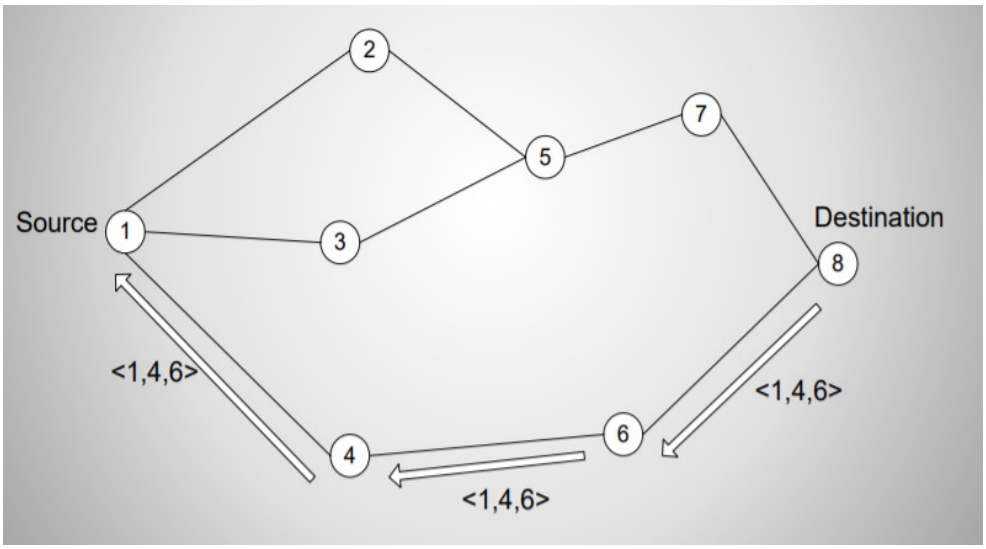
In Dynamic Source Routing, data packets contain a list of nodes that represent the routes from source to destination. This makes the protocol on demand routing protocol. Protocol consist of route discovery and route maintenance. Nodes in DSR uses the cache to reach out to the destination. It may have multiple routes to a destination but may use any of the routes for packet being sent.



Source-Google

Fig 2.1. Propagation of RREQ packet

As you can see in the above figure, source sends the Route Request Packet to its neighbor to reach out the destination. If the destination node is not available in the intermediate nodes, it further sends the RREQ packet along with the information of nodes through it pass the RREQ packet.



Source-Google

Fig 2.2. Propagation of Route Reply Packet

When the Destination node receives the RREQ packet, it replies with the Route Reply Packet (RREP) back to source node. RREP act as an acknowledgment sent back to source node. It also sends the information of all nodes through which the RREP packet passed. Source used the optimal path to reach out the destination.

The source first checks if the required route is available before sending data packets. It checks through the route cache. If the route is available to use, source sends the routing information inside the packet along with the data packet. If the route is not available, source node broadcast a route request (RREQ) packet. A node that receives the RREQ packet checks its route cache and replies from its cache if it has route to destination. If that node is not destination node, it appends its own address to the route record field of the RREQ packet and broadcast the RREQ locally to its neighbors. When the destination node receives the RREQ packet, it generates a route reply packet (RREP) that has the list of addresses received in RREQ and send it back to along this path to the source and update the routing information. [2]

Each node on the route confirm that the packet has been received by the next node in the route and retransmit the packet if necessary. Even after limited number of retransmissions, packet is still not getting delivered, it is considered the link from this node to next hop is broken. The route error

packet (RERR) is to the source node indicating that this link is broken. Source must use alternate route from its route cache or discover a new route to the destination.

2.2 AD-hoc On-Demand Distance Vector Routing

AODV is very simple and effective outing protocol for Mobile Ad-hoc network. Every node in network act as a router and find its routes to the destination. Every node maintains a routing table with routing information of its neighboring nodes, which contains a node sequence number and broadcast-id. Whenever source wants to communicate with destination node, it increments its broadcast-id and initiates a path discovery by broadcasting a RREQ to its neighbors. The RREQ have different fields as- Source Address, Source Sequence number, Address of Destination, Destination sequence number, number of hops the packet can take.

The source address and broadcast id pair are used to identify the RREQ uniquely.

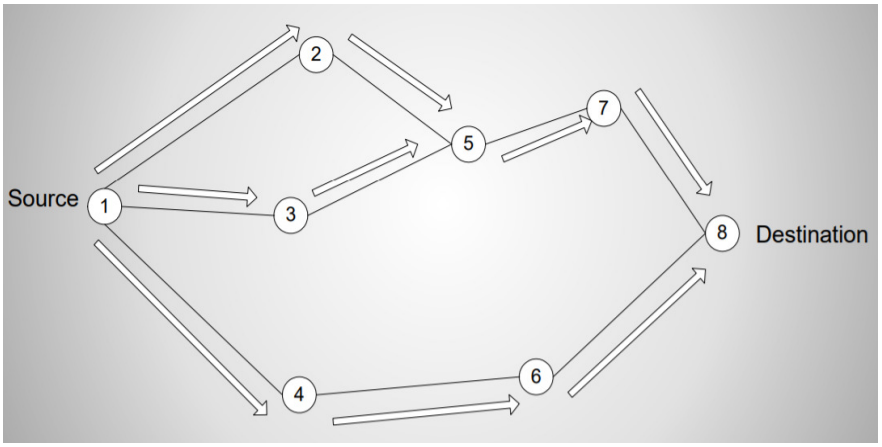
Each time routing information is added to the routing table, when the node receives the RREQ packets. Router compare the sequence number and hop count with he existing information in table. IF the router has already the information about new entry, it updates the routing table. A node can match the RREQ packets with the same ID as the last received packet. For this routing table has some fixed entries 1. Destination IP address 2. sequence number 3. destination sequence number flag 4. hop count 5. network interface 6. next hop 7. Live time

A node checks that if the received packet is with the same ID packets, then it should update the table or not. For this, it checks if the sequence number is greater than the entry in table and hop count is less than the previous one. Based on this information is updated. The information about the RREP will be loaded in the routing table.

Whenever there is a link failed between the nodes, a router error (RERR) packet is generated in the network and sent to the source node. After receiving the RERR packet, source sends the RREQ packet to the entire network to establish a new connection. Entire network is flooded with RREQ packets. As the intermediate nodes are closer to source nodes, all the data is loaded on these nodes. These nodes get busier and some nodes are not utilized properly. As a result, network cannot use

bandwidth properly and some nodes have low traffic during the transmission. It affects the overall performance of network and cause the congestion.

In AODV, the route is discovered, when source generated the RREQ packets and send them to its neighboring nodes. When any intermediate node receives the RREQ packet, it adds the routing information to the node's routing table. This routing table helps to determine, which node is destination. If the node itself is not a destination node, it further checks the RREQ packet with same ID have been received before or not. Node will discard the current received RREQ packet, if it has already received the RREQ packet from same ID. If the RREQ packet 9is not from same id, it will further broadcast the packet to its neighboring nodes. All the routing tables will be updated.

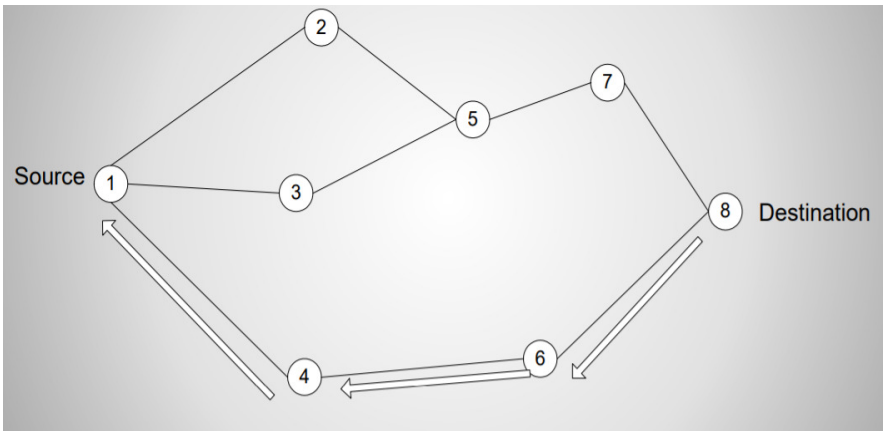


Source-Google

Fig 2.3. Route of RREQ Packet

As you can see in the above figure, source node "1" wants to communicate with the Destination "8". Node "1" sends the RREQ packets to its neighboring nodes, if these nodes are not destination nodes, further sends the RREQ packets till reach the destination. This make the network multi-hop network.

When the destination node receives the RREQ packet, it generates the RREP packet. These RREP packet act as acknowledgment to the source node. The information is updated on the routing table and intermediate nodes sends these routing packets to the source node based on the routing table. As the source node multiple RREP packet, so the path taken by the RREP packet selected by the source node. Source selects the optimal path for the RREP packet and discards all the path.



Source-Google

Fig 2.4. Route Reply Packet

When the destination received the RREQ packet, It reply with route request packet (RREP). Source gets the multiple route reply packet for single route request packet. This causes the overhead in the network. Source chooses the shortest path to reach the destination.

In AODV, while discovering the route source start sending the Route Request Packet (RREQ) to all the nodes in the network. The RREQ packets broadcast in the entire network. Each node receives the RREQ packet. Nodes check its routing table for the available route from source to destination to establish the communication. If there is route available from source to destination, destination node generates the RREP packet to its neighbor. After getting RREP packet, source node start sending the packets to the destination node using the same path by the RREP packet.

There could be chance that some node rebroadcast the RREQ in AODV. The entire network gets blindly flooded with the RREQ packet, so while discovery the route, congestion is caused in the network. Also, whenever there is link failure between the nodes, A RERR packet is generated. The RERR packet is sent to source node by a node that loss the connection. After receiving the RERR packets, the source node starts sending RREQ packet in whole network. It blindly floods the whole network with the RREQ packets. It affects the performance of AODV as the packet delivery ratio decrease. During my graduation period, I also worked to Modify the AODV (EMAODV) which overcome the problem of congestion. [3]

2.3 Destination Sequenced Distance Vector (DSDV) Protocol

DSDV is proactive routing protocol, which has information about its node and route to neighboring nodes, even before establishing the communication. In DSDV, sequence number of every node is added in the routing table. Each node has information in its routing table, based on this route pf packet is decided and transmit in the network.

The node forms a list of all the available destinations and no of hops require to reach the destination. All this information is available in the routing table. Each node updates the routing table for transmission pf packets and to maintain the connection between nodes in network. A sequence number is also added in the routing table, from the destination node.

IN DSDV, every mobile node broadcast its routing table to its neighbor, so that the routing information will be keep updated all the time. When the nodes move in the network, the information about the packets broadcasted in network to keep the routing information. As each node contain the sequence number, so in new node contain the information about sequence number, it also contains more information regarding destination address, how many no. of hops required to reach the destination, a new sequence number.

The routing table contain the sequence number, which generated by the source node. In order to update the table, new sequence number is required from the destination. The receiving node increment the metric and transmit the information. The incoming packet could have to travel more to reach out to destination, if the metric is not incremented before the transmission. [4]

2.4 Fisheye State Routing (FSR) Protocol:

The Fisheye State Routing Protocol is a hierarchical routing protocol. It works on the 'fisheye' technique. In the fish eye technique, the eye of a fish captures with high detail of pixels near the focal point. As the detail of pixels decrease when the distance from focal point start increasing. This technique is proposed by Kleinrock and Stevens. In FSR, the fish eye approach translates to maintain accurate distance and the path quality information about the nodes in the neighborhood.

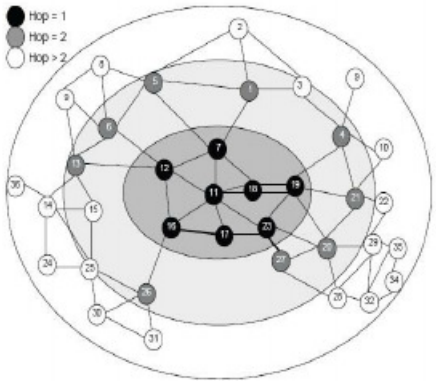


Fig 2.5 FSR routing protocol

The Global State Routing (GSR) is also the same concept as FSR. In Global State Routing, there is only one fisheye scope level. So, the entire topology table is updated with the neighbors. As the network size increases, the consumption of the bandwidth in network is also increased. In the FSR, link state packets are exchanged to start the communication. This is not used the event driven technology.

The scope distance describes the updating link state information. This information related with the different frequencies. The FSR scales the wall to large network and keep the computation accuracy near the destination.

In Fish State Routing (FSR), the movement of mobile nodes in network cause the broken link. This broken link is detected and described as infinity. Whenever there is route break in network, the infinity metric is assigned to the metric by updating the hop and sequence number. The sequence number that are generated by the mobile hosts called to be even number and the sequence number generated by the infinity metrics called the odd numbers

The sequence number compared whenever the information packet is received from another node. The sequence number is compared with the available sequence number entry in the table. If the sequence number is larger, the entry in the table will be updated. The information will be updated with the new sequence number. If the sequence number is same as in the table, it will check the metric entry. If the no of hops in metric entry is less than the previous entry, the new information will be updated in the system. When the node information is updated, the metric is increased by 1 and the sequence number is increased by 2. When the node enters in the network, it announces in the network, so that the routing information will be updated. This routing information will add the new entry for the new node.

3. Published EMAODV Protocol

This paper has been published in 9th IEEE Annual Computing and Communication Workshop and Conference (IEEE CCWC 2019), Las Vegas, NV, USA, January 2019, pp. 895-899. The paper is represented by Harshdeep S Jhajj in the Conference.

3.1 Problem Statement

In AODV, while discovering the route source start sending the Route Request Packet (RREQ) to all the nodes in the network. The RREQ packets broadcast in the entire network. Each node receives the RREQ packet. Nodes check its routing table for the available route from source to destination to establish the communication. If there is route available from source to destination, destination node generates the RREP packet to its neighbor. After getting RREP packet, source node start sending the packets to the destination node using the same path by the RREP packet.

There could be chance that some node rebroadcast the RREQ in AODV. The entire network gets blindly flooded with the RREQ packet, so while discovery the route, congestion is caused in the network. Also, whenever there is link failure between the nodes, A RERR packet is generated. The RERR packet is sent to source node by a node that loss the connection. After receiving the RERR packets, the source node starts sending RREQ packet in whole network. It blindly floods the whole network with the RREQ packets. It affects the performance of AODV as the packet delivery ratio decrease. During my graduation period, I also worked to Modify the AODV (EMAODV) which overcome the problem of congestion.

3.2 Proposed Method

The EMAODV use the time to live (TTL) factor to find the route from source to destination. Along with the overhead, it also reduces the flooding of entire network with RREQ packet. This protocol finds the path from source to destination and information regarding the all the other nodes. TTL is factor that set the hop range until which the RREQ packet can propagate. In the starting the TTL value is set to zero, so that all the nodes can take participate in the route discovery. The RREQ packet can propagate with in the hop range which is equal to TTL value.

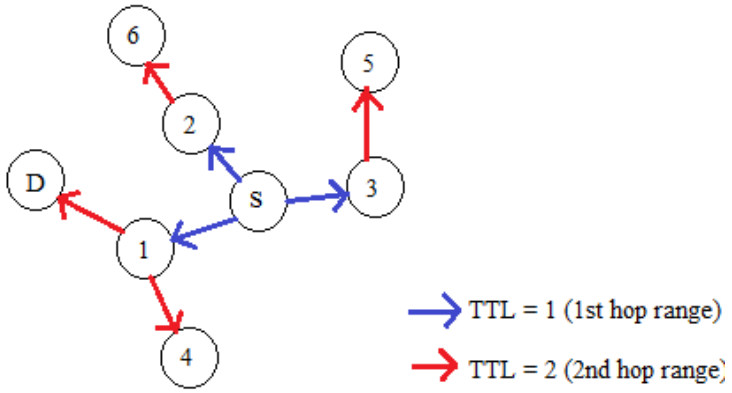


Fig 3.1. Time to Live Factor

If the route reply packet is not received by the source in first hop range, hop range is increased so that the destination will be in next hop. As in above figure, the source "S" wants to send data to destination "D". In the EEMAODV, it will use the "TTL" factor reach out the destination. In the starting of communication, TTL value will be set as "1", will increment the TTL value till find the destination. When the value of TTL is 1, the route discovery will happen only in its first hop. This means the RREQ packets will be sent only to its one hop neighbors. If there is no information about the destination in the first hop range, the source node increases the value of TTL to two and send the RREQ packets. As the nodes that are in reach of the second hop, will receive the RREQ packet from source node. If the destination will be in 2^{nd} hop range, it will reply with the RREP packet as an acknowledgment. Even if the destination is not in 2^{nd} hop, source will increase the TTL value and look for destination in 3^{rd} hop. This will go on and increase the TTL value until it reaches at the destination.

In EEMAODV, we used the initial and relay value, based on these nodes could be silent or relaying. If the node is silent means that node doesn't take part in the route discovery process. If the node is relaying means that node can take part in route discovery process. In the starting of communication, all the relay and forward value set to "1", so that all the nodes can take part in route discovery. The relay vale is updated based on the Predecessor-field and forward value depend on the Time to Live value of RREQ. RREQ packet contain the various fields including source address, destination address, hop count, source sequence number, destination sequence number and the last address. The last address is the address of node from which the current node receives the RREQ packet. [4] [5]

In EEMAODV, one more filed is added to the RREQ packet format, is Predecessor-address filed, which contain the address of node from which last node receives its RREQ packet.

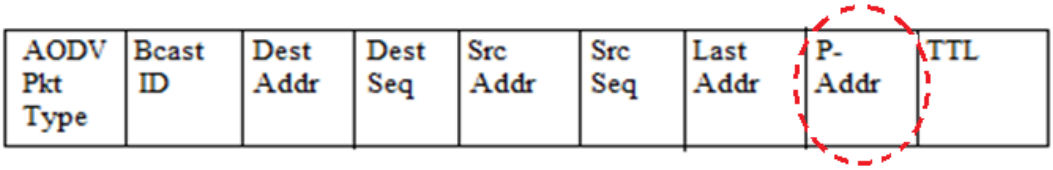


Fig 3.2. Predecessor-address

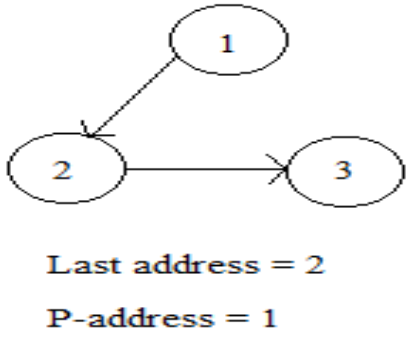


Fig 3.3. Last and P-Address

As in figure 3.3, The node sends the data packet to node 2, so the last address at node 2 will be 1. Node two further send the packet to node 3, at node 3 the last address will be 2 and the P-adder will be 1. The last address of the previous node will be the P-adder of current node which receive the RREQ packet.

The P-Addr field in the RREQ packet is checked when the RREQ packet is received. It is used to verify whether the packet has been processed before or not. When the received bode address is same as the P-Addr of the RREQ, the Relay value will be set to 1. When the Relay value is set to 1, nodes can take part in route discovery. IF the relay value is set to zero, node can't take part in route discovery. Each time node receives the RREQ, it checks the TTL value in the node. If the TTL value is zero, the RREQ will be drop by checking the P-Addr of the node. If the TTL value

is greater than zero, TTL value, current node set its own address as last node and forward the route request packet.

Architecture

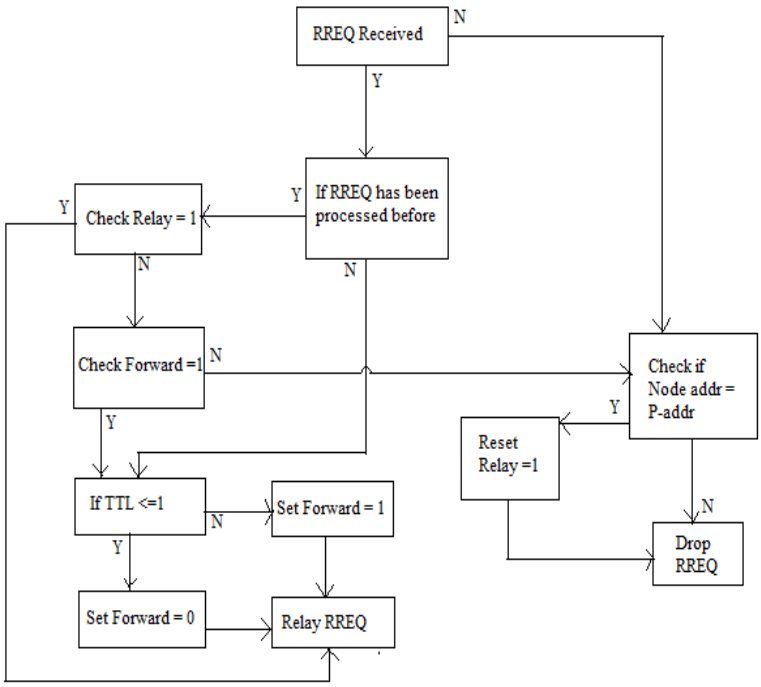


Fig 3.4. EMAODV Architecture

While discovery the route, source flooded the entire network with the RREQ packet. When there is link failure between the nodes, source sends the RREQ packet again. Some nodes receive the duplicate RREQ packet. When a node receives the RREQ packet with the same broadcast id, it will check the Predecessor-address of the RREQ packet. If the node has same Predecessor-address value in the RREQ packet, it will set the relay value as 1, so that node can take part in the route discovery process. If it doesn't have same value as Predecessor-address, it will not take part in route discovery. This will also set the relay value as zero, this means that node already processed the RREQ packet once. The TTL value also played the important role in the processing of RREQ packet. Whenever the node receives an RREQ packet, it checks the TTL value, if TTL value is zero, it will drop the RREQ packet by checking the Predecessor-address. If the TTL value is grater

than zero, the last address of RREQ packet will be update as Predecessor-address. It will forward the RREQ packet after adding its own address as last address.

In completion, the work of EMAODV, I have worked on the implementation part of the routing protocol.

3.3 Simulation Setup

All the simulations of EMAODV carried out in network simulator 2(NS2). For the simulation, we must set some parameters.

Parameter	Value
Operating System	Ubuntu 14.04
Simulator	NS-2(ns-2.35
Channel Type	Wireless Channel
Number of Nodes	10,60,100
Speed(m/s)	10
Data Type	UDP
Simulation Time	100
MAC Protocol	802.11
Data Packet Size	512
Simulation Area	1200*1200
Radio Propagation Model	Two Ray Ground
Routing Protocol	EMAODV, AODV, DSR

Table 3.1 Parameters

All these parameters are same for different number of nodes. We use 10, 60, and 100 no nodes for the simulation.

Performance of all the MANET protocols are check on different metrics. These metrics are packet loss, packet delivery ratio, end-to end delay and throughput.

1. Packet Loss- Packet loss in any routing protocols can be determined by the number of packets transmitted minus the number of packets received We can get total no packets lost during the transmission.

   Packet loss= Packets Transmitted – Packets Received

2. Packet Delivery Ratio- It is the ratio of number of packets received to the total number of packets transmitted.
   Packet Delivery Ratio= No of packets received/ no of packets transmitted.

3. End-to-End Delay- It the time taken by the routing protocols for the packets delivered from a one node to destination node.

4. Throughput- Throughput describes the efficiency of any routing protocols. It is the ration of data packets received to the time taken by the simulation.

3.4 Simulation Results

Based on the above performance metrics, performance of all the routing protocols have been calculated.

1) Packet Loss- The packet loss in EMAODV is very less as compared to the AODV. We received more data packets at the receiver end in EMAODV than AODV. But the packet loss is less than the DSR, because the DSR has the best performance in the Packet loss.

Number of Nodes	DSR	AODV	EMAODV
10	5935	6160	5960
60	5126	5557	5473
100	5391	7159	6912

Table 3.2 Packet loss

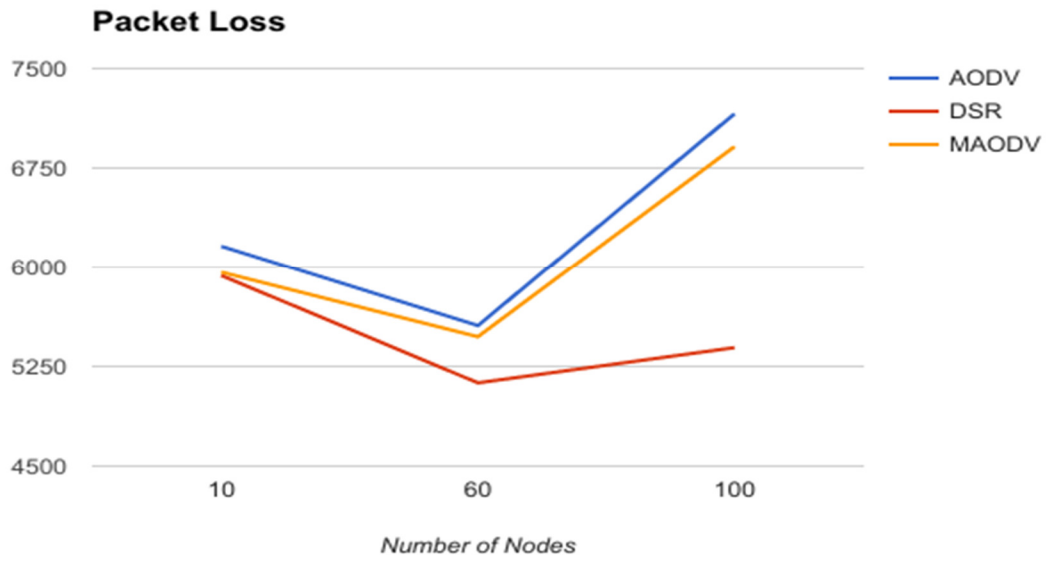


Fig 3.5. Packet Loss Comparison

2) Packet Delivery Ratio- The EMAODV has better delivery ratio when compared to the AODV. The main purpose of this protocol is get improvement than AODV, which is achieved.

Number of Nodes	DSR	AODV	EMAODV
10	79.936	79.584	79.746
60	81.189	79.412	79.781
100	83.481	78.663	78.937

Table 3.3 Packet Delivery ratio

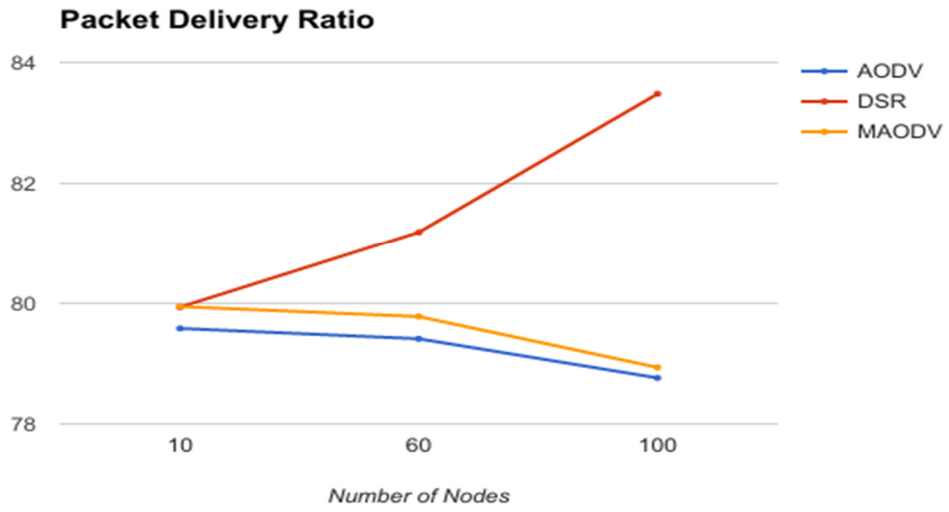


Fig 3.6. Packet Delivery Ratio

3) END-END Delay- The modifeid AODV (EMAODV) has better end to end delay as compared to both DSR and AODV.

Number of Nodes	DSR	AODV	EMAODV
10	0.1471	0.1468	0.1453
60	0.1483	0.1472	0.1464
100	0.1512	0.1494	0.1492

Table 3.4 END-END Delay

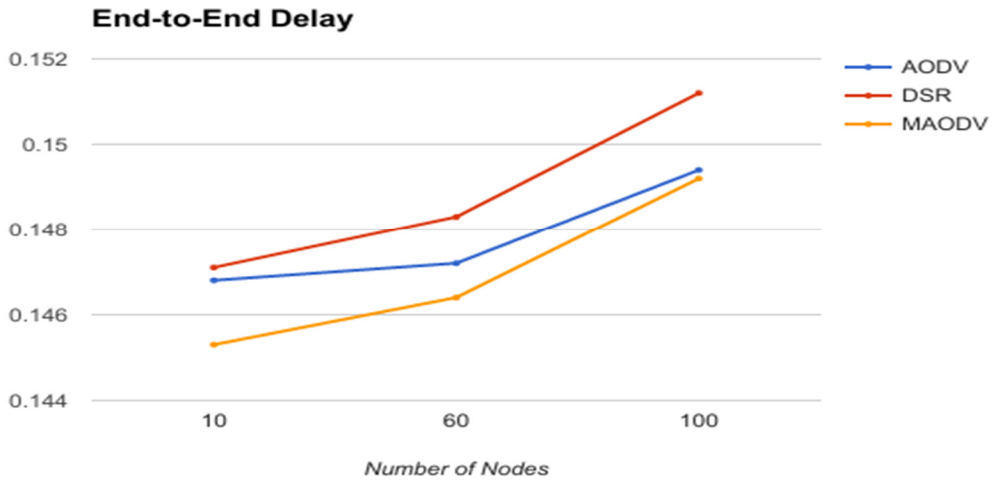


Fig 3.7. End to End Delay Comparison

End-To-End delay decides the time taken by the packets to reach at the destination. Less end-end delay mens, the smilulation is faster. So, EMAODV gives the better end-to-end delay

4) Throughput- EMAODV has better throughput when compared to the other reactive protocols like AODV and DSR.

Number of Nodes	DSR	AODV	EMAODV
10	369.468	403.112	403.963
60	389.113	401.828	403.161
100	395.371	400.061	401.981

Table 3.5 Throughput

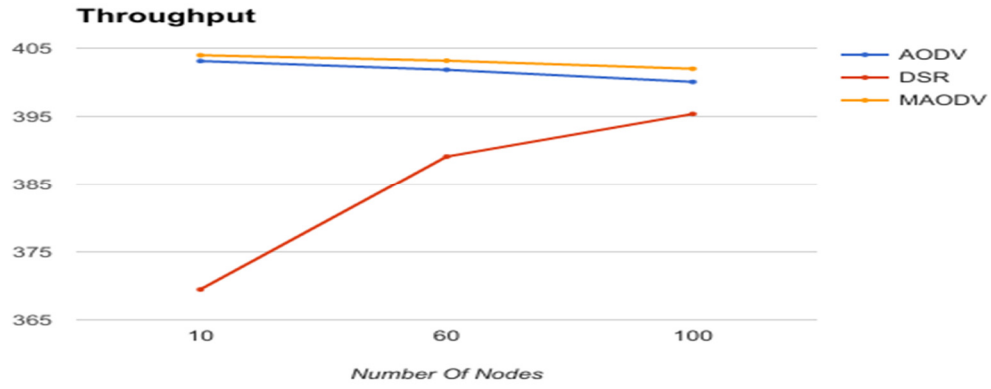


Fig 3.8. Throughput

3.5 Results Justification

Based on the throughput values of the DSR, AODV and EMAODV, the precision values are calculated. We got higher values of precision in EEMAODV, this means protocols has higher efficiency.

Protocol	Mean	Variance	Precision
DSR	384.648	182.675	0.00547
AODV	401.667	2.346	0.00547
EMAODV	403.701	1.659	0.6027

Table 3.6 Performance Results

The modification in existing AODV has been done to remove the congestion due to link failures. In EMAODV, we have created a path for route discovery in the protocols. The EEMAODV is more efficient as compared to AODV and DSR based on the precision value.

My contribution- In this paper, I worked on the Implementation of the Proposed routing algorithm.

5. Published NCMDSDV Protocol

This paper has been accepted to the 11th IEEE International Conference on Communication Software and Networks (ICCSN 2019), Chongqing, China, 06/2019.

4.1 Problem Statement

In DSDV, nodes send the "Hello" packet to its neighbor to maintain the route to destination. Every-time also broadcast its routing table to the neighbor nodes to keep the information of network. A node which receives the "Hello" packet will update its routing table and add the sender node as its neighbor node. Now every node will have full information about its neighbors, it makes the nodes capable to make the routing table updated all the time. Each node will have the valid path from source to destination node.

Whenever the link between two nodes is broken, DSDV uses the stale path, because it is a single path routing protocol. This reduces the packet delivery ratio. To overcome this issue multipath routing can be used.

4.2 Proposed Protocol

A new protocol is designed, which overcome the problem of DSDV, called Neighbor Coverage Multipath DSDV. In MANET all the nodes are independent and mobile in nature. So, the network topology changed according to the nature of nodes.

There are many non-linked paths, that can be used to send the data packets faster. These non-linked paths don't have any common nodes between the source and destination nodes. So in case of broken links, instead of dropping the RREQ packets, these non-linked paths can be used. This also make sure the delivery of data packet more quickly. So, the overall performance of the routing protocol increased by maintaining the multiple paths in Mobile Ad-Hoc Network. The use of non-link path is less expensive and help to utilize the bandwidth properly.

In order to maintain the non-linked paths, two new fields are added in the routing table. These fields are generated by the destination- "second hop" and "link-id"

Routing table contains the information of all the nodes and its neighboring nodes. Every time there is change in network, it will be updated in the routing table. A new filed "Second hop" is added in the routing table. This contain the information about the other nodes that can reach the destination.

Field	Description
Destination Node	Address of the destination node
Next Hop	First hop to destination
Second Hop	Second hop to destination
No. of hops	Number of hops to destination
Link-Id	An ID generated by new node for the new routers
Sequence Number	A number that distinguishes between the stale and fresh routers
Time	The time when the path was discovered.

Table-4.1 DSDV

In DSDV, neighbor table is used that contain all the information about the neighbor of nodes and find the status of a node. Whenever the neighbor table is empty, node generate the "Hello" message to get the updated list of its neighbors. When any node wants to establish the communication, neighbor table is used. Every time any node leaves the network or new node enters in network; neighbor table will be updated.

Field	Description
Neighbor ID	Address of neighbor node
Link-ID	Link number between new node and neighbor

Table 4.2 Link-Id

In NCMDSDV, whenever the path between two nodes is broken, a message will be generated by the node which detect the failure and send it to all the neighbors in the network. As you can in figure below, there is link failure between the nodes 4 and 6. Node 4 will generate the failure message and will send to its neighbors.

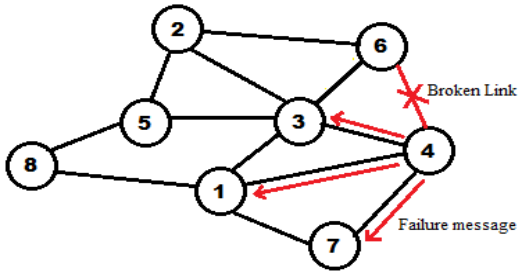


Fig 4.2. Failure message

Field	Description
Sender	Address that sends the failure message
Destination	Address of the Destination
Link-Id	Link of the broken path

Table 4.3 NCMDSDV

This message will have the link-id of the broke path. When the node sends the failure message of broken link, routing table will delete all the entries related to that link-id. Also, the nodes that receive the failure message, also check its table and delete all the path related to broken link-id. So whole network will update its routing table and remove the broken link.

If any node is network, still uses the broken path for the communication. It will use the alternate path for the communication, will generate the error message and sent to the previous nodes for deletion of the broken path. This error message will contain the information about the link-id of broken path and information about the alternate path that can be used. This message will be broadcasted until it reaches at destination.

4.3 Simulation- The simulation of pro-active routing protocols DSDV and FSR have been carried out using the network simulator 2. For the simulations of the protocols performance metrics has been set.

Parameter	Value
Operating System	Ubuntu 14.04
Simulator	NS-2(ns-2.35
Channel Type	Wireless Channel
Number of Nodes	10,60,100
Speed(m/s)	10
Data Type	UDP
Simulation Time	100
MAC Protocol	802.11
Data Packet Size	512
Simulation Area	1200*1200
Radio Propagation Model	Two Ray Ground
Routing Protocol	DSDV, FSR, NCMDADV

Table 4.4 Simulation Parameters

1. Packet Loss- Packet loss in any routing protocols can be determined by the number of packets transmitted minus the number of packets received We can get total no packets lost during the transmission.

Packet loss= Packets Transmitted – Packets Received

2. Packet Delivery Ratio- It is the ratio of number of packets received to the total number of packets transmitted.

Packet Delivery Ratio= No of packets received/ no of packets transmitted.

3. End-to-End Delay- It the time taken by the routing protocols for the packets delivered from a one node to destination node.

End-to-end delay = Time at which packets received - time at which packet has been sent

4. Throughput- Throughput describes the efficiency of any routing protocols. It is the ratio of data packets received to the time taken by the simulation. [7]

4.4 Resullts

The simulation results of protocol have been achieved on 10, 60 and 100 nodes.

1. Packet loss- The no of packet loss in NCMDSDV is less as compared to the DSDV and FSR. We received more packets at the receiver end.

Number of Nodes	DSDV	FSR	NCMDSDV
10	6075	9892	5993
60	4390	9586	4123
100	4398	8350	4089

Table 4.5 Packet Loss

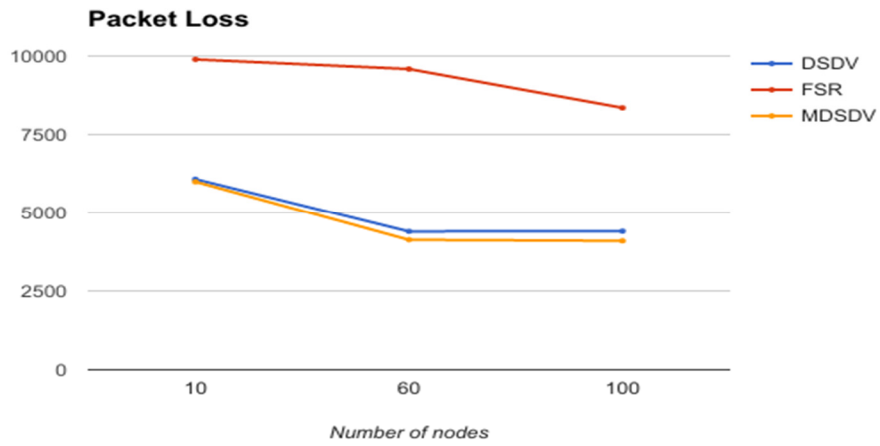


Fig 4.3 Packet Loss

From the above results, NCMDSDV provide the lees no of packet loss than the DSDV and FSR. When the 60 number of nodes are used for simulation, the packet loss in traditional DSDV is 4390, on ther other hand in the purposed NCMDSDV the no of packets lost in simulation are 4123.

2. Packet delivery ratio- The packet delivery ratio in the NCMDSDV is more as compared to the other pro-active protocols. As the no. of nodes increased, packet delivery ratio is getting decreased but it is more than the other protocols.

Number of Nodes	DSDV	FSR	NCMDSDV
10	80.9642	81.2418	81.9844
60	78.9234	75.5863	79.9624
100	75.4861	74.9822	76.8642

Tbale 4.6 Packet Delivery Ratio

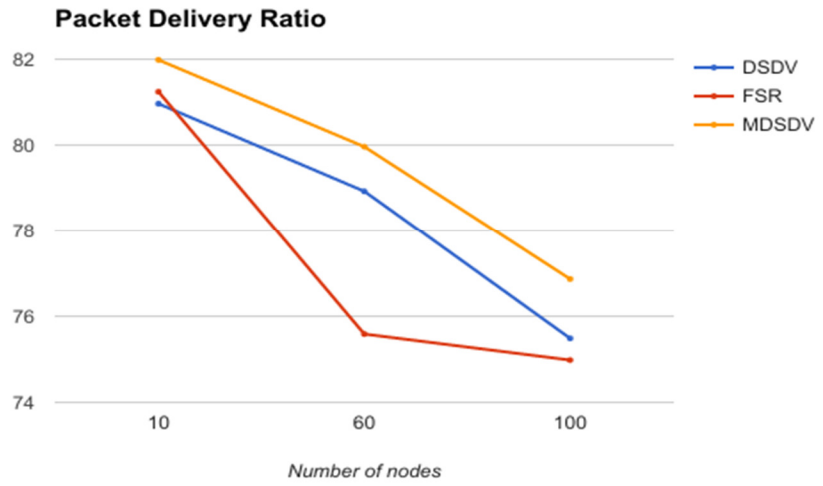


Fig 4.4.Packet Delivery Ratio

The packet delivery ratio for evry routing protocol is decrsed as the no of nodes increased. So the density of network describs the packet delivery ratio. In NCMDSDV, the packet delivery ratio is still higher than the DSDV and FSR for all no of nodes.

3. END-To-END Delay- The NCMDSDV has less end to end delay than the DSDV for all no of nodes. The FSR has highest end to end delay as comapred to DSDV and NCMDSDV.

Number of Nodes	DSDV	FSR	NCMDSDV
10	0.1473	0.1461	0.1435
60	0.1536	0.1643	0.1518
100	0.1592	0.1684	0.1569

Table 4.7 End-To-End Delay

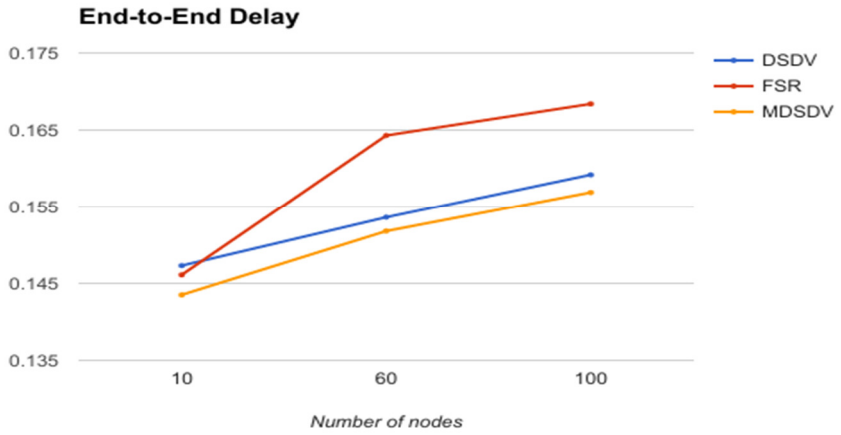


Fig 4.5. End-to-End Delay

The End-to-End delay is the time taken by the data packet to reach the destination. The less end-to-end delay means the fater simulation. Packet takes less time to reach at the destination. NCMDSDV has less end-to-end delay.

4. Throughput- Throughput is the retio of the data packet received at the reciver end to the time taken by the completion of simulation.

Number of Nodes	DSDV	FSR	NCMDSDV
10	398.7826	399.2482	401.8297
60	398.1849	388.5823	399.9698
100	397.2983	385.4129	399.0127

Table 4.8 Throughput

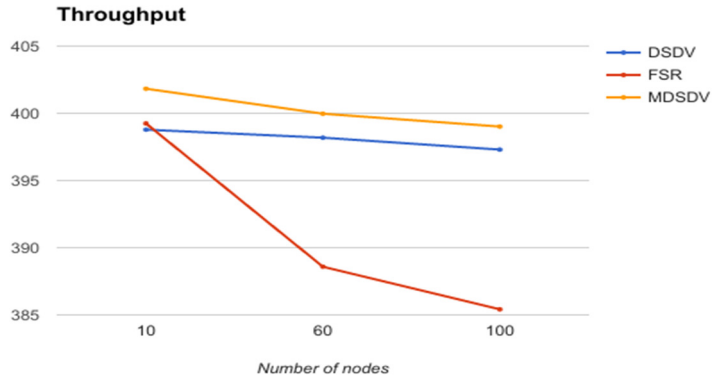


Fig 4.6. Throughput

The purposed NCMDSDV protocol provide the higher throughput in all no of nodes. In 100 number of nodes, DSDV gives the throughput 397.2983 and the NCMDSDV gives 399.0127 throughout value.

4.5 Result Justification

The NCMDSDV shows quite improvement in the packet loss compared to other pro-active protocols. For all the protocols the performance is increased as the number of nodes increased. The end-to-end delay value is getting less as size of the network increased.

A new protocol has been proposed, which is based on the DSDV routing protocol called the Neighbor Coverage Multipath DSDV. This protocol gives the solution for the link failures. The multipath to the destination will be created in case of link failure. NCMDSDV has been proposed to use the multiple paths to destination in case of link failures. So, the two new fields “Second hop” and “Link-id” has been added in the protocols. These fields used find the non-linked path between the source and destination. This also generates the broadcast failure message and error messages. This helps in finding the alternate paths for packet from source to destination.

All the simulations of the DSDV routing protocol and FSR routing protocol has been carried out using the NS2. The result of these two have been compared with the Neighbor Coverage Multipath DSDV. For the output, we can conclude that NCMDSDV has better packet delivery ratio and throughput than the existing DSDV and FSR. The simulation time of NCMDSDV is faster as it has less end-to-end delay and packet loss. At the receiver end, we receive the more packets. The precision value of any protocol defined the efficiency of the protocol. In case of NCMDSDV, it has the higher precision value than the DSDV and FSR. This mean it is more efficient than the any other pro-active routing protocol.

My contribution- In this paper, I worked on the Implementation of the Proposed routing protocol.

5. Proposed Modified Dynamic Source Routing (M-DSR)

5.1 Proposed Protocol- A new protocol has been implemented by making changes and modifications in the on the Dynamic Source Routing (DSR). This is named as Modified Dynamic Source Routing (M-DSR)

5.2 Problem Statement-

The source first checks if the required route is available before sending data packets. It checks through the route cache. If the route is available to use, source sends the routing information inside the packet along with the data packet. If the route is not available, source node broadcast a route request (RREQ) packet. A node that receives the RREQ packet checks its route cache and replies from its cache if it has route to destination. If that node is not destination node, it appends its own address to the route record field of the RREQ packet and broadcast the RREQ locally to its neighbors. When the destination node receives the RREQ packet, it generates a route reply packet (RREP) that has the list of addresses received in RREQ and send it back to along this path to the source and update the routing information.

Each node on the route confirm that the packet has been received by the next node in the route and retransmit the packet if necessary. Even after limited number of retransmissions, packet is still not getting delivered, it is considered the link from this node to next hop is broken. The route error packet (RERR) is to the source node indicating that this link is broken. Source must use alternate route from its route cache or discover a new route to the destination.

In Dynamic Source Routing, whenever there is link failure, A route error packet is generated. This error packet is sent back to source node, to establish the new routing path in network. Upon receiving the error packet, source nodes start sending the RREQ packet to find the route again. In this process, intermediate nodes used mostly for the communication. This is because these nodes are the shortest range from the source node. There are still other nodes available, that are not fully utilized, even having low traffic. So, the bandwidth in the network is not utilized fully. The packet delivery ration got reduced cause of the delay occur in network. To overcome, a congestion counter is added in the DSR protocol. It checks the congestion on current node.

5.3 Proposed Method- In the proposed Modified DSR (M-DSR), path is determined for the communication by the congestion counter. It decreases the packet congestion while delivering the data packet. M-DSR checks the stress on the current node in the routing table and each time counter is checked when processing the RREQ packet and while reply with the Route Reply Packet (RREP). The congestion flag is added in the route packet. The congestion flag is set to true based on the congestion in the communication. Whenever there is link break or RREQ receive at the source end, congestion flag set to zero.

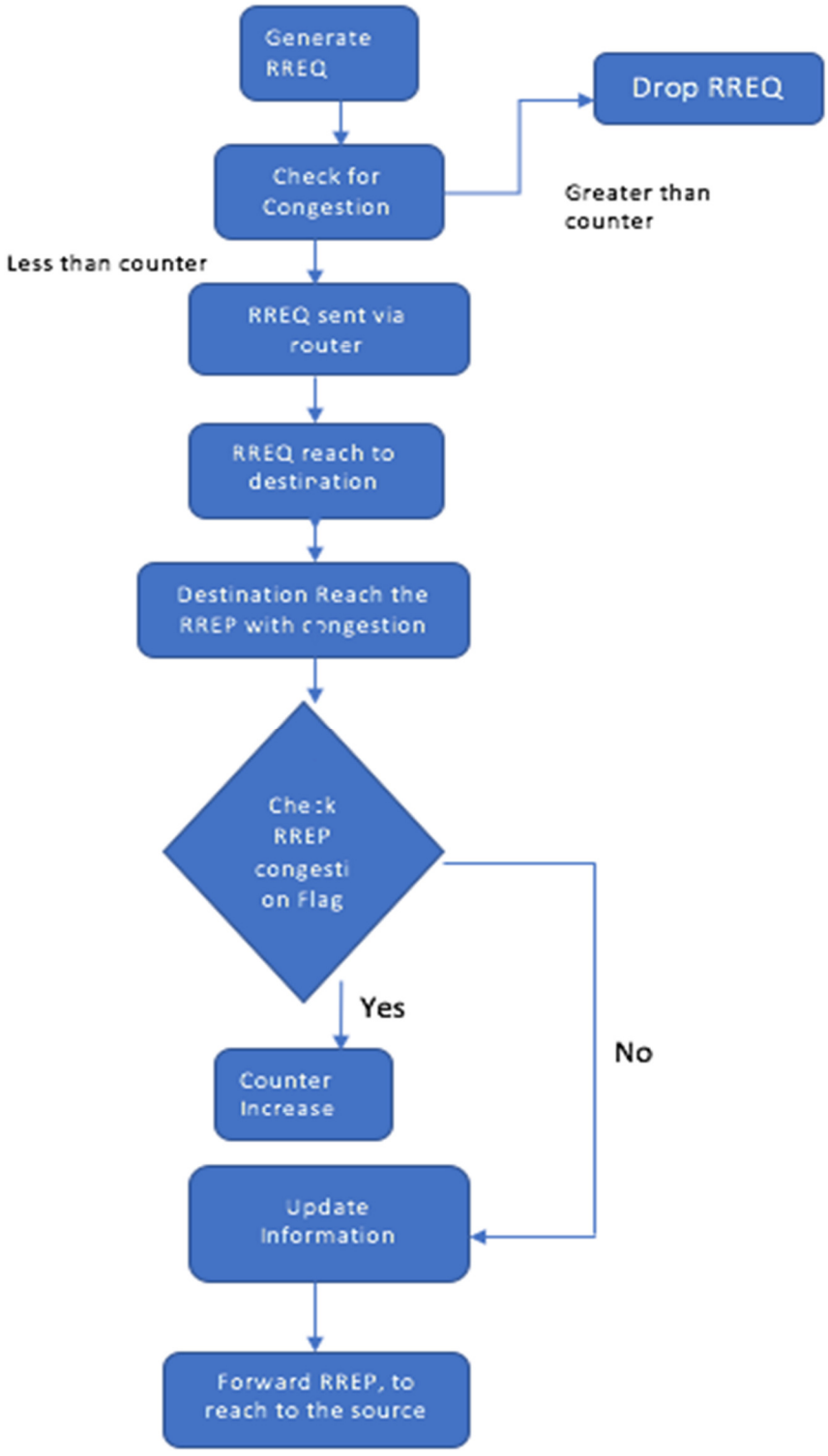


Fig 5.1. Modified DSR

The source node flooded the RREQ packet in the entire network. When RREQ packet arrives at the intermediate node, router checks the congestion level at this point. It compares the congestion counter with pre-determined value, if it is less than some pre-determined value, routing table will update and forward the RREQ to next counter. On other case, it will drop the RREQ packet. When the route request packet reaches at destination, it will generate the route reply packet. In the header of route reply packet, a congestion flag is added. When the source node is establishing the route, it generates the route reply packet. When the neighbor nodes maintain the route. The destination node set the congestion flag true, when it receives the RREQ packet form the source node. The Route reply packet unicast back to the source node. The router checks the congestion flag. If the congestion flag is true, counter gets incremented. If the flag is not true, counter keeps the same. Router updates the information in the routing table. [9]

In M-DSR, a 32-bit congestion control flag is added in the header of route reply packet. The different congestion control entries are added in the router.

1. Initial Congestion Counter

   When the routing table is initiated, the congestion counter is generated in the table and value of counter is set to “ZERO”.

2. Increment the congestion counter

   When the node receives the route reply packet from destination, it checks the value of congestion counter. If the congestion flag is set “true”, counter is incremented by “1”. Unless the packet will be drop.

3. Decrement the congestion counter

   The new entry is added into the table called life time. When the life time is expired, counter subtracts by 1.

   Whenever there is a link failure between source node and the destination node, a route error package will be sent to the source node. The counter with broken node will be subtracted by 1.

4. Reset the congestion counter

Whenever any node is removed from network due to any reason, the counter will be reset to “ZERO”.

Destination IP-Address	Destination Sequence Number	Origin IP ADDRESS	Life Time	Congestion Flag

Table 5.1 Congestion Counter

The congestion flag is carried by the Route Reply packet. This flag is bi-directional back to the source node. The flag check the node, which is currently used for completing the communication and carried the RREP. The congestion counter will increment its value after the routing table detect the flag. So in the Route Reply Packet format, a congestion flag is also added.

Destination IP-Address
Destination Sequence Number
Originator IP Address
Lifetime
Congestion Flag

Table 5.2 New Route Reply Packet

So, when the RREQ package occur at the node, the congestion counter is compared with pre-determined value. This process is done by the router. If the comparison is less than counter, router will update the routing table information and forward to next router. If the value is more than the counter, the RREQ packet will be drop. When the RREQ packet reach at the destination, a RREP packet will generated by the destination node. As shown in above figure, a congestion flag is also added to the header of RREP packet. The RREP packet is generated from source to destination and from a neighbor node to maintain the route. When the RREQ packet is received by correct destination, it set the congestion flag true. When destination reply with RREP, router checks the congestion flag. If the congestion flag is true, counter will increment otherwise it will keeps the same value. The information will be updates in routing table.

6. Simulation Tools

6.1 NS3

Network Simulator 3 is a network simulator used for the research and educational purpose. Network Simulator 3 is licensed under the GNU GPLv2 license. It is the free software compatible with the Ubuntu operating system.

NS-3 is a real-time network simulator, its infrastructure allows the development of the simulation models realistic. This feature allows the system to connect with the real world and allow to make the changes in existing applications.

Ns3 simulation is easy to use and debug. This type of system help the entire simulation work flow to collect the data from trace files and do the analysis. This makes the work flow easier. NS3 also supports the real-time scheduler. The real-time scheduler uses the loop simulation for interaction with the real-time systems. Furthermore, in real time scheduler, operator can send and receive the data packet on the real network devices. NS-3 work as the frame-work to add the link effects between the virtual machines.

The NS-3 supports both the IP and non-IP based networks. TCP performance and Mobile AD-Hoc routing protocols are one of the mostly used by the researchers. But the major focus area is the wireless/IP simulations which have the models for the Wi-fi, Wi-max and other wireless systems. Mostly the wireless systems are used for layer 1 and 2.

NS3 Installation

INSTALLING NS3 IN UBUNTU 14.04:

I. Ubuntu is installed run following command opening your terminal.
II. To install prerequisites type given below command

sudo apt-get install gcc g++ python python-dev mercurial bzr gdb valgrind gsl-bin libgsl0-dev libgsl0ldbl flex bison tcpdump sqlite sqlite3 libsqlite3-dev libxml2 libxml2-dev libgtk2.0-0 libgtk2.0-dev uncrustify doxygen graphviz imagemagick texlive texlive-latex-extra texlive-generic-extra texlive-generic-recommended texinfo dia texlive texlive-latex-

extra texlive-extra-utils texlive-generic-recommended texi2html python-pygraphviz python-kiwi python-pygoocanvas libgoocanvas-dev python-pygccxml

It will ask password ,give your system password

III. Download NS-3.26 given below link

https://www.nsnam.org/ns-3-26/download/

IV. ns-allinone-3.26.tar copy and paste into Ubuntu Home location

then extract the ns-allinone-3.26.tar file

V. next terminal to type the command:

cd ns-allinone-3.26/

VI. Then you can find build.py along with other files so type the command

./build.py

If the build is successful, then it will give output "Build finished successfully".

VII. To build with waf so move to ns-3.25 so type command: cd ns-3.26

sudo ./waf

display your ns3 supported modules like(AODV,DSR,CSMA,etc...)

These above seven steps to ns3 installation completed!

Features of NS3

1. NS3 uses the C++ using the Python Bindings.
2. In NS3, Compilation time is very less due to the change in the hardware. The modern hardware capabilities make it faster to compile the code.
3. In NS#, we can write the simulation script as the C++ language, this in not available in the NS2.

4. As this system supports the memory management function of C++, so the function such as new, delete, malloc are available in the NS3.
5. The automatic de-allocation of objects is supported in NS3, it is used when dealing with the data packet objects.
6. A data packet contains the buffer of bytes and the collection of bytes called the meta-data.
7. The information is added to the buffer using the subclasses. For example, Header, used to add the information in beginning of the buffer and Trailer add the information in the end.
8. NS3 has better memory management as compared to the NS2.
9. It is easier in NS3 to determine the whether the specific header is attached or not.
10. NS3 operates on the package PyViz, which is the real-time visualization environment system works on the python. [10]
11. Ns3 use the classes such as core-module.h and network-modules.h.

The comparison between NS2 and NS3

NS2	NS3
NS2 uses the TCL as scritping language	NS3 uses the C++ and phython
It uses nam animator	Use Pyhton visualizer nad NetAnim for the animation
It is not actively mainted	Ns3 actively maintained and supoorted

Used for simulation	Used for simlulation and emulation also
Recompilation is long and more chnaces of fail	Recompilation is fast
Mac protocol is fixed	Mac protocl can be defiend
Manet simulation only	Manet and Noc Simulation
Not give power consumption	Easy to get power consmuption

Table 6.1. NS2 - NS3

6.2 NetAnim (Network Animator)

NetAnim is an offline network animator tool which now ships along with the ns-allinone-3.xx package. It can animate the ns-3 network simulation using an XML trace file that is generated as an output during simulation. So, the necessary steps for creating this XML trace file and setting its related attributes should be done in the ns-3 simulation code itself. [11]

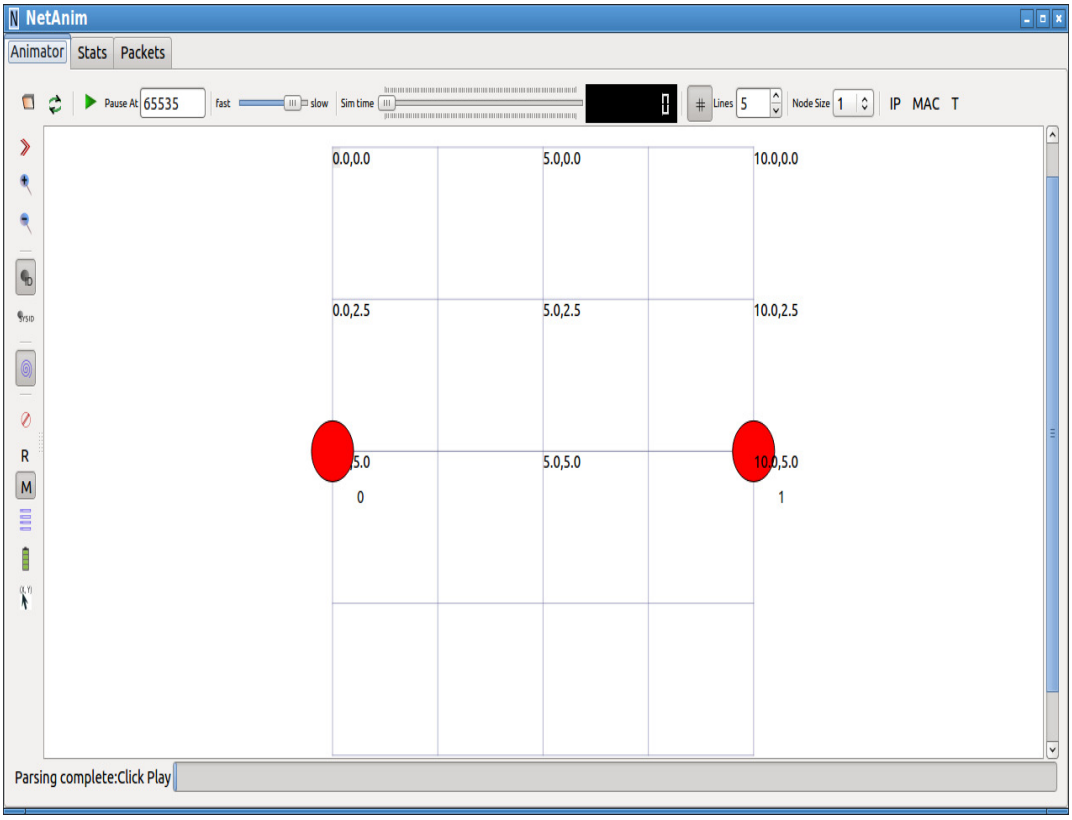


Fig 6.1. Network Animator

7. Simulation

7.1 Simulation Setup- For the simulation of Modified DSR, network simulator 3 has been used. To check the better performance than the DSR and AODV some parameters have been setup. The network simulator for wireless network is compatible with the Ubuntu operating system.

Parameter	Value
Operating System	Ubuntu 14.4
Simulator	NS-3
Channel Type	Wireless
No. of Nodes	30, 50
Radio Propagation	Two- Ray
MAC Protocol	802.11
Simulation Area	500*500
Routing Protocol	DSR
Node Speed	10m/s
Data Packet Size	512

Table 7.1 Simulation Parameters

7.2 Performance Metrics- The performance of three routing protocols AODV, DSR and M-DSR gas been calculated based on four metrics.

1. Packet Loss
2. Packet Delivery Ratio
3. End to End Delay

4. Throughput

1. Packet Loss- Packet loss in any routing protocols can be determined by the number of packets transmitted minus the number of packets received We can get total no packets lost during the transmission.

   Packet loss= Packets Transmitted – Packets Received

2. Packet Delivery Ratio- It is the ratio of number of packets received to the total number of packets transmitted.

   Packet Delivery Ratio= No of packets received/ no of packets transmitted.

3. End-to-End Delay- It the time taken by the routing protocols for the packets delivered from a one node to destination node.

   End-to-end delay = Time at which packets received - time at which packet has been sent

4. Throughput- Throughput describes the efficiency of any routing protocols. It is the ration of data packets received to the time taken by the simulation.

7.3 Network Simulation- The simulation of nodes has been carried out for 30 and 50 no of nodes. The no of nodes decides the density of network.

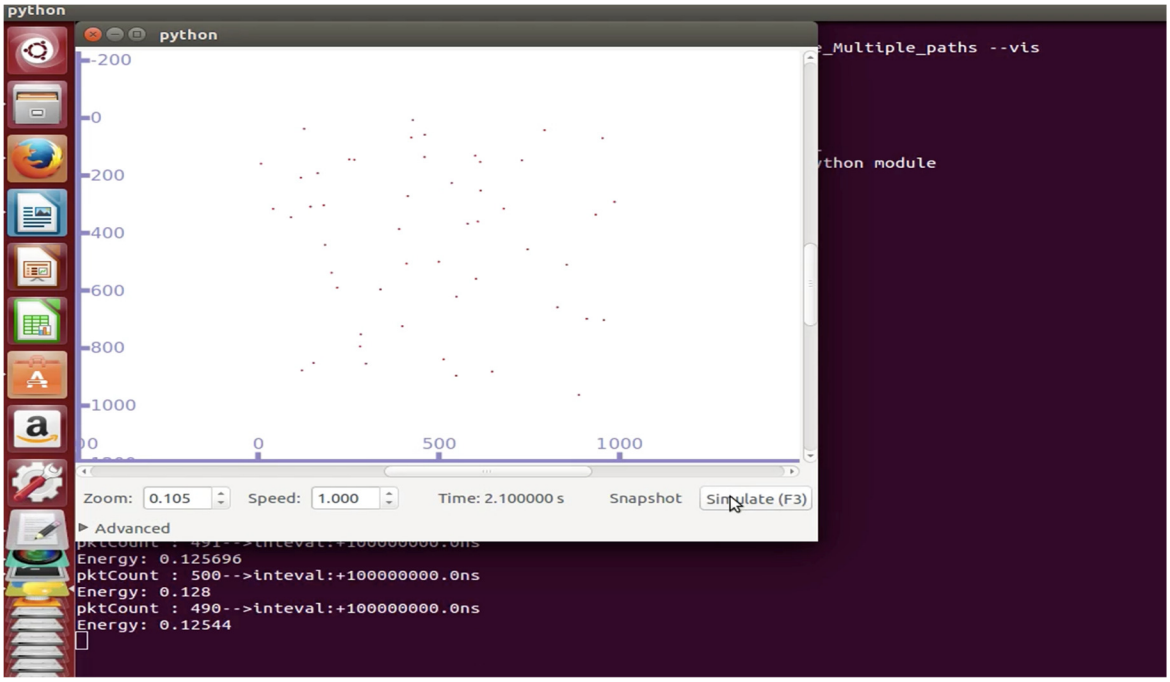


Fig 7.1 Simulation Window

The above figure shows the simulation window for the routing protocol. The compilation of the code has been start by typing

Sudo .waf (file_name.vis)

In NS-3, we use the .waf command to build the system in the operating system.

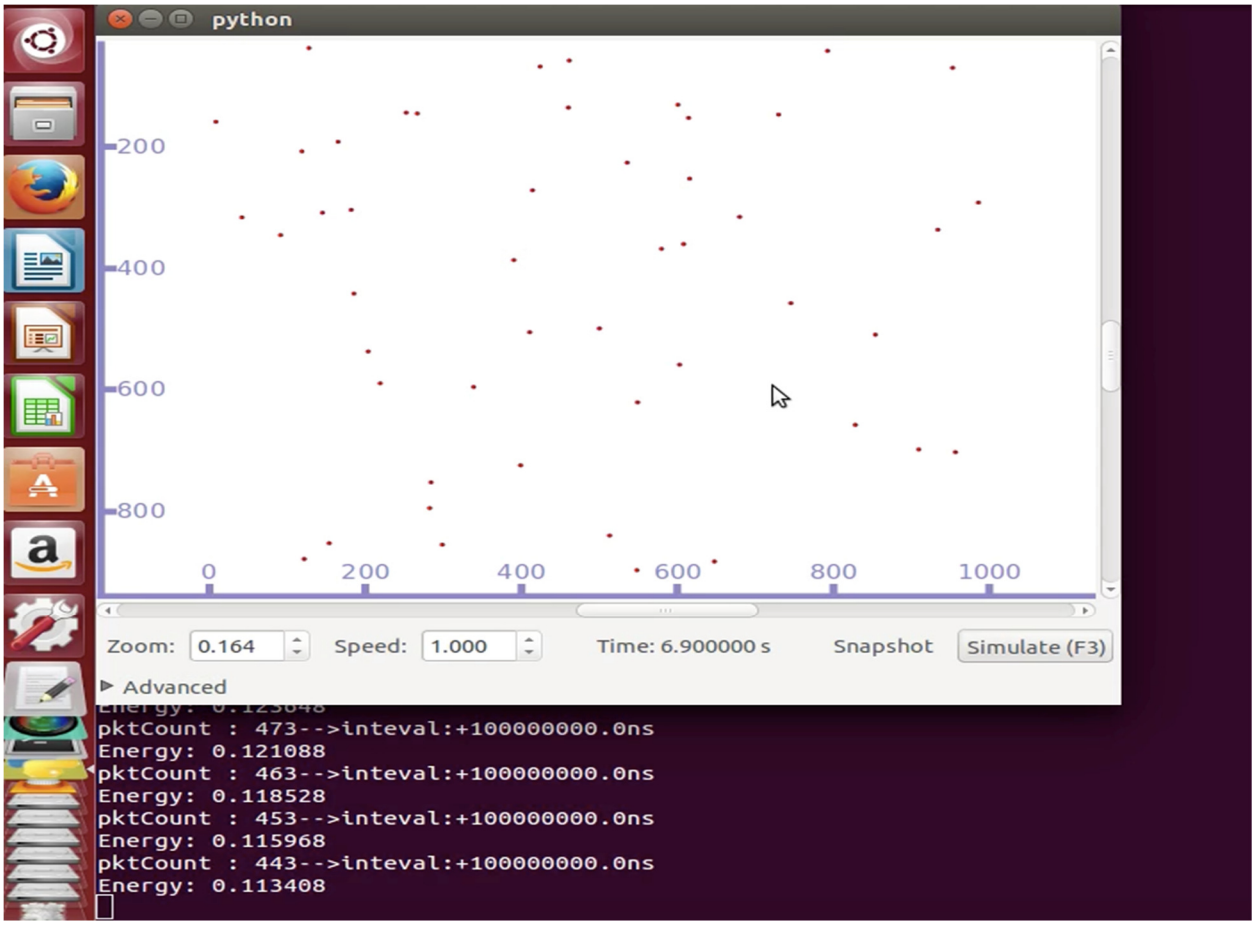


Fig 7.2 Simulation

The compilation window in NS3, also show the no of packet count that has been transmitted along with the energy consumption.

7.4 NetAnim- For the Visual animation of the working network, NS-3 provide the tool netAnim. It uses the trace file for the animation. The XML trace file generated by the output from the compilation.

- Open the Network Animator- In the terminal go the Ns3 folder in your computer. Navigate to the NetAnim3.107 folder. Use the following commands.

  -cd ns-allinone 3.26

  -cd NetAnim-3.107

  -cd ./NetAnim

  A window will be open like below. This is the basic window of NetAnim.

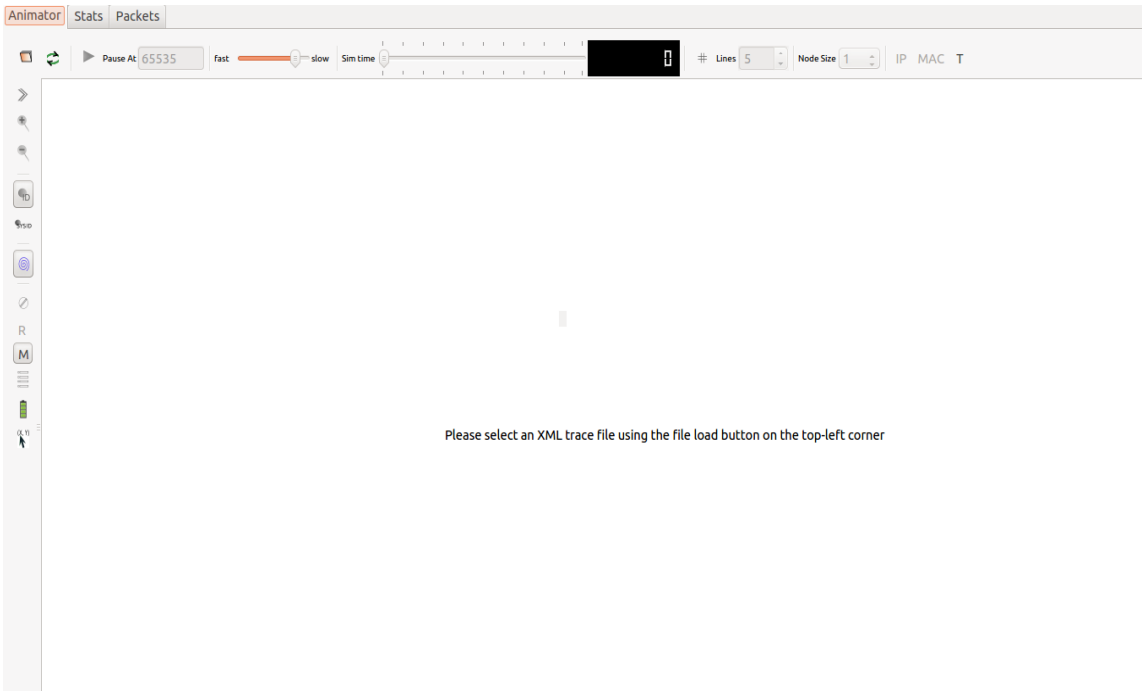


Fig 7.3 Netanim

- After this, we need the XML trace file to load in the NetAnim. The XML file is generated by the simulation output.

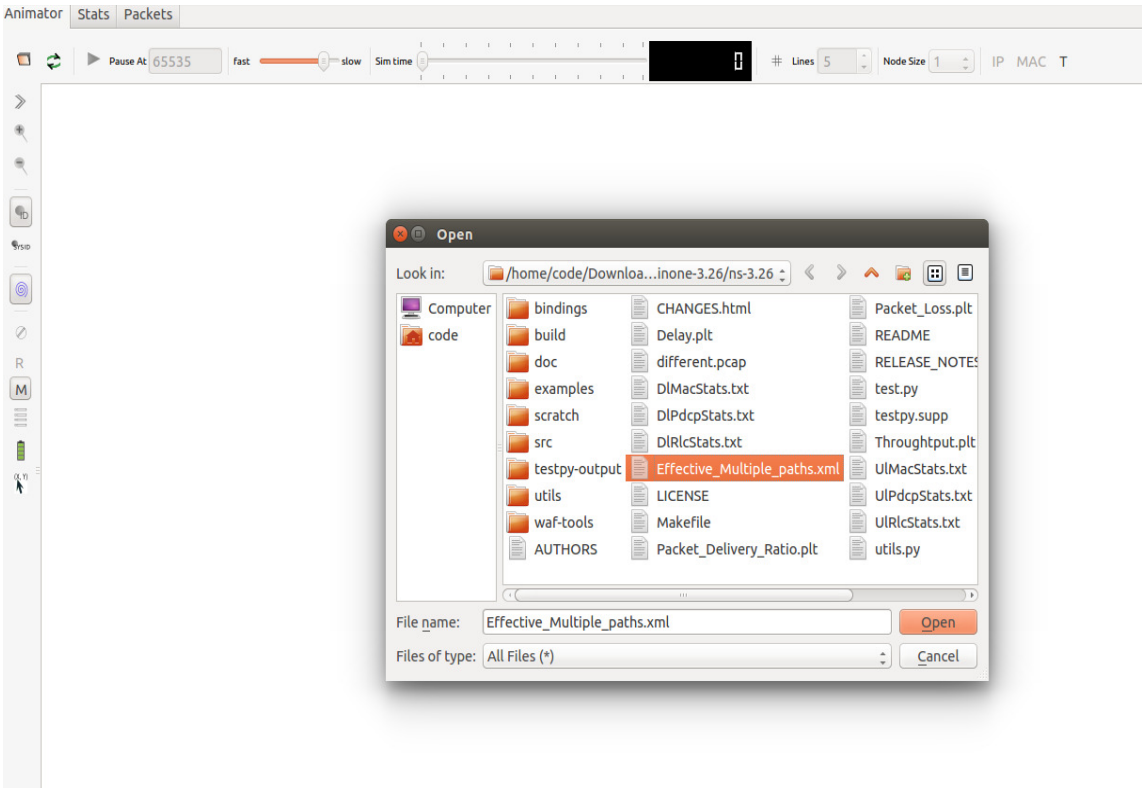


Fig 7.4 Loading XML

- As seen in above figure .XML file is loaded in the system.
- By pressing the Play button simulation will be started.

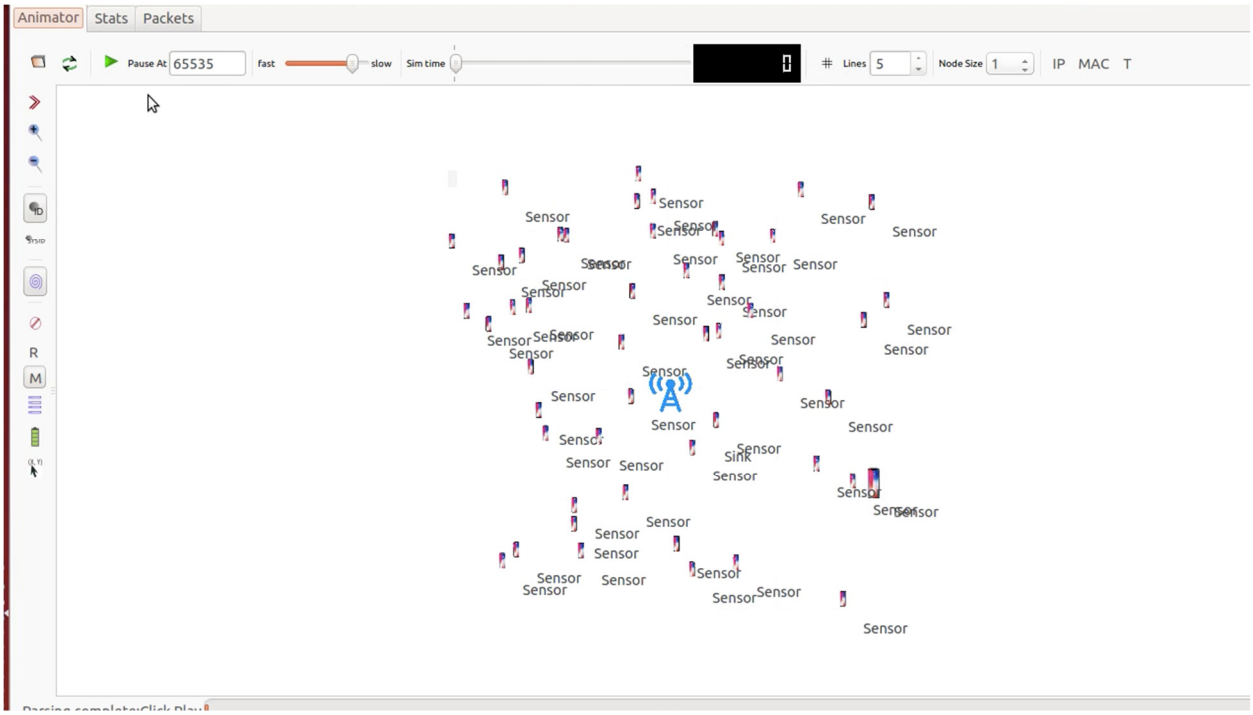


Fig 7.5 NetAnim Simulation

The above figure is the Basic window in the NetAnim after selecting the XML file. After selecting the XML trace file, all the no of nodes will show in the animation window. The no nodes are shown as the sensor. In the NetAnim, we can also choose the custom icon for the sensors.

7.5 Simulation of Nodes

After pressing the play button in the simulation window, we can see the communication between the different nodes. It also gives us opportunity to the simulation time, control the speed of simulation.

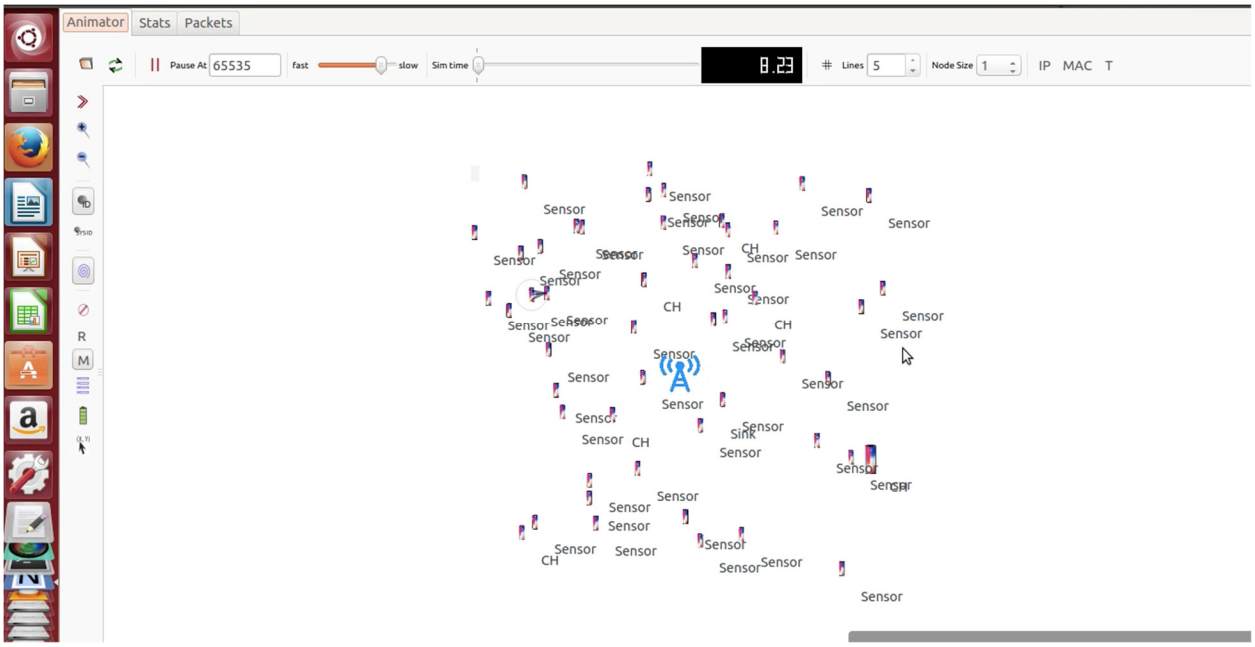


Fig 7.6 NetAnim Simulation

7.6 Results- The results of the simulation of the AODV, DSR, and M-DSR has been calculated. Results has been proposed based on four performance metrics.

[1] Packet loss- It is the difference between the number of packet transmitted to the no of packets received.

No of Nodes	DSR	AODV	M-DSR
30	2600	3147	2478
50	1954	2893	1456

Table 7.2 Packet Loss

Form the above table, we can see that packet loss in the M-DSR is less compared to the both reactive protocol DSR and AODV. When the no of nodes is 50, the packet loss in DSR is 1954 whereas in the modified DSR packet loss is 1456.

[2] Packet Delivery Ratio- It is the ratio of number of packets received to the total number of packets transmitted.

Packet Delivery Ratio= No of packets received/ no of packets transmitted.

No of Nodes	DSR	AODV	M-DSR
30	75.6%	73.1%	75.2%
50	78.8%	75.5%	79.3%

Table 7.3 Packet Delivery Ration

The packet delivery ratio in M-DSR is more than the DSR and AODV. Modified DSR give the better performance for 30 no of nodes and 50 no of nodes.

[3] End-to-End Delay- It the time taken by the routing protocols for the packets delivered from a one node to destination node.

End-to-end delay = Time at which packets received – time at which packet has been sent

No of Nodes	DSR	AODV	M-DSR
30	75.6%	73.1%	75.2%
50	78.8%	75.5%	79.3%

Table 7.4 End-To End Delay

The modified DSR give the better performance than other routing protocols in End-to-End delay also.

[4] Throughput- Throughput describes the efficiency of any routing protocols. It is the ration of data packets received to the time taken by the simulation.

No of Nodes	DSR	AODV	M-DSR
30	75.6%	73.1%	75.2%
50	78.8%	75.5%	79.3%

Table 7.5 Throughput

The overall efficiency of the routing protocols has been checked out using the throughput of the routing protocols. Higher the throughput, higher the efficiency. The M-DSR offers the higher throughput as the no of nodes increased.

8. Conclusion

In this proposal, we purpose a new reactive protocol based on the DSR routing protocol. This protocol used to overcome the problem of congestion in the existing routing protocol. This has been achieved by adding the congestion counter in the routing protocol. The congestion counter compare the value of counter with some pre-defined value. Based on the algorithm in the protocols, the value of counter is increased or decreased. The congestion flag is used in RREP packet also, to check the congestion while back from source to destination. If there is any link failure or path is broken, counter will be reset.

The simulation has been carried out and compared the result with other reactive protocol DSR and AODV. Based on the four-performance metrics like packet loss, end-to-end delay, packet delivery ratio and throughput, the Modified DSR provide the better performance in all the four aspects.

My contribution- In this project work, I worked on the implementation of algorithm in the DSR. Complete the proposed routing algorithm by following the required algorithm. I also work on the simulation and results

9 References

1. Mobile Ad-hoc Networks, WIKIPEDIA. (n.d.). Retrieved from https://en.wikipedia.org/wiki/Mobile_ad_hoc_network
2. Abdelshafy, M., & King, P. (n.d.). Dynamic Source Routing under Attacks.
3. Bai, Y., Mai, Y., & Dr.Wang, N. (2017). Performance Comparison and Evaluation of the Proactive and Reactive Routing Protocols for MANETs.
4. Mahajan, R., & Jagtap, R. (2013). Energy Efficient Routing Protocols for Mobile Ad-Hoc Networks. *International Journal of Science and Modern Engineering (IJISME), 1*(3), 23196386th ser.
5. H. Jhajj, R. Datla and N. Wang, "Design and Implementation of An Efficient Multipath AODV Routing Algorithm for MANETs," *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2019, pp. 0527-0531.
6. Rahman, M. A., Anwar, F., Naeem, J. and Abedin, M. S. M. 2010. “A Simulation Based Performance Comparison of Routing Protocol on Mobile Ad-hoc Network (Proactive, Reactive and Hybrid).” International Conference on Computer and Communication Engineering, 11-13
7. Yefa Mai, Yuxia Bai, Nan Wang. “Performance Comparison and Evaluation of the Routing Protocols for MANETs Using NS3”, Journal of Electrical Engineering 5 (2017) 187-195 doi: 10.17265/2328-2223/2017.04.003
8. Mukhija, A. (2001). *Reactive Routing Protocol for Mobile Ad-Hoc Networks* (Unpublished master's thesis). Indian Institute of Technology, Delhi.
9. Y. Mai, F. M. Rodriguez and N. Wang, "CC-ADOV: An effective multiple paths congestion control AODV," *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, 2018, pp. 1000-1004. doi: 10.1109/CCWC.2018.8301758
10. https://www.nsnam.org/
11. Jorg, D. O. 2003. “Performance Comparison of MANET Routing Protocols in Different Network Sizes.” University of Berne, Switzerland.

Appendix-

Code

```
uint32_t packetsReceived;
Multicrieteria newobj;
void GetDistance_From (NodeContainer node1, NodeContainer node2){
Ptr<MobilityModel> model1 = node1.Get(0)->GetObject<MobilityModel>();
Ptr<MobilityModel> model2 = node2.Get(10)->GetObject<MobilityModel>();
distance1 = model1->GetDistanceFrom (model2);}
void delaycalc(int bits,double dist,int mystatus){
if(mystatus==status){
Initialtime=(Simulator::Now ()).GetSeconds ()-Initialtime;}}
void CheckPDR (double recivpk){}
void Identification_Timecalc (int Identification_Timeval){
kbs = (Identification_Timeval/10000)-kbs;}
void ReceivePacket (Ptr<Socket> socket){
Ptr<Packet> pckt;
while (pckt = socket->Recv ()){
rate="512Mbps";
Ptr<MobilityModel> model1 = BaseNode.Get(0)->GetObject<MobilityModel>();
Ptr<MobilityModel> model2 = MobileNodes.Get(10)->GetObject<MobilityModel>();
distance1 = model1->GetDistanceFrom (model2);
bytesTotal = pckt->GetSize();
Identification_Timevalue += pckt->GetSize();
delaycalc(bytesTotal,distance1,1);
```

```
Identification_Timecalc (Identification_Timevalue);
pk=pk+1;
CheckPDR (pk);}}
void compare_Minimum(double dis){
if(ds>dis){
ds=dis;}}
void getNearbynode(NodeContainer nod,double x1,double y1){
int nn;
for(uint32_t i=0;i<nod.GetN ();i++){
Ptr<ConstantPositionMobilityModel> FCMob = nod.Get(i)-
>GetObject<ConstantPositionMobilityModel>();
Vector m_position = FCMob->GetPosition();
double x=m_position.x;
double y=m_position.y;
double xx=x1-x;
double yy=y1-y;
double x2=(xx*xx);
double y2=(yy*yy);
double sx=sqrt(x2);
double sy=sqrt(y2);
double dis=(sx+sy);
compare_Minimum(dis);
if(ds==dis){
nn=i;}}
std::cout<<"minimum Distance:" <<nn<<std::endl;}
```

```
std::cout<<"Energy: "<<energy<<"\n";}
```

```
static void GenerateTraffic (Ptr<Socket> socket, uint32_t pktSize,uint32_t pktCount, Time
pktInterval ){
if (pktCount > 0){
socket->Send (Create<Packet> (pktSize));
std::cout<<"pktCount : "<<pktCount<<"-->inteval:"<<pktInterval <<"\n";
int b=pktCount*512;
energy=(double)trans_recv;
energyReceive(b,1);
energyTransmit( b,ds,1);
Simulator::Schedule (pktInterval, &GenerateTraffic,socket, pktSize,pktCount-1, pktInterval);
}else{
socket->Close ();}}
void GETCH(NodeContainer c){
int *ch =newobj.proposed_ALG(50,MobileNodes);
for (uint32_t j = 0; j < MobileNodes.GetN (); ++j){
if(j%9==0){
if(ch>0){std::cout<<*(ch);}
anim->UpdateNodeDescription (MobileNodes.Get (j), "CH" );}
else{anim->UpdateNodeDescription (MobileNodes.Get (j), "Sensor" );}}}
void PKTtrans(NodeContainer c , NodeContainer d){
double min = 0.0;
double max = 15.0;
Ptr<UniformRandomVariable> x = CreateObject<UniformRandomVariable> ();
x->SetAttribute ("Min", DoubleValue (min));
x->SetAttribute ("Max", DoubleValue (max));
int value = x->GetValue ();
rnd=value;
TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");
Ptr<Socket> recvSink = Socket::CreateSocket (d.Get (rnd), tid);
```

```
InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80);
recvSink->Bind (local);
recvSink->SetRecvCallback (MakeCallback (&ReceivePacket));
rnd=rnd+2;
Ptr<Socket> source = Socket::CreateSocket (d.Get (rnd), tid);
InetSocketAddress remote = InetSocketAddress (Ipv4Address ("255.255.255.255"), 80);
source->SetAllowBroadcast (true);
source->Connect (remote);
Simulator::ScheduleWithContext (source->GetNode ()->GetId (),Seconds (0.1),
&GenerateTraffic,source, packetSize, numPackets,interPacketInterval);}
void PKTtrans1(NodeContainer c , NodeContainer d){
TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");
Ptr<Socket> recvSink = Socket::CreateSocket (d.Get (rnd), tid);
InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80);
recvSink->Bind (local);
recvSink->SetRecvCallback (MakeCallback (&ReceivePacket));
Detection objs;
objs.detect(rnd);
rnd=rnd+2;
Ptr<Socket> source = Socket::CreateSocket (d.Get (rnd), tid);
InetSocketAddress remote = InetSocketAddress (Ipv4Address ("255.255.255.255"), 80);
source->SetAllowBroadcast (true);
source->Connect (remote);
Simulator::ScheduleWithContext (source->GetNode ()->GetId (),Seconds (0.1),
&GenerateTraffic,source, packetSize, numPackets, interPacketInterval);}
void PKTtrans2(NodeContainer c , NodeContainer d){
TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");
Ptr<Socket> recvSink = Socket::CreateSocket (d.Get (rnd), tid);
InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80);
```

```
recvSink->Bind (local);
recvSink->SetRecvCallback (MakeCallback (&ReceivePacket));
Detection objs;
objs.detect(rnd);
rnd=rnd+2;
Ptr<Socket> source = Socket::CreateSocket (d.Get (rnd), tid);
```