PROJECT REPORT ON

FUNCTIONAL ANALYSIS OF VANET BASED ON AODV AND DSDV PROTOCOLS USING SUMO AND NS2

Report submitted to the SASTRA Deemed to be University as the requirement for the course

CSE302: COMPUTER NETWORKS

Submitted by

ADABALA BHASKARA SURYA PRAKASH
(REG NO.: 124160088, BTech ECE-CPS)

DECEMBER 2022

SASTRA
ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION
DEEMED TO BE UNIVERSITY
(U/S 3 of the UGC Act, 1956)
THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

SCHOOL OF ELECTRICAL AND ELECTRONICS ENGINEERING
THANJAVUR, TAMIL NADU, INDIA – 613 401

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act,1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING

THANJAVUR, TAMIL NADU, INDIA-613401

BONAFIDE CERTIFICATE

Certified that this project work entitled **"FUNCTIONAL ANALYSIS OF VANET BASED ON AODV AND DSDV PROTOCOLS USING SUMO AND NS2"** submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech.is a bonafide record of the work doneby Shri **ADABALA BHASKARA SURYA PRAKASH (124160088, B. Tech ECE-CPS**) during the academic year 2022-23, in the School of Electrical and Electronics engineering.

Project Based Work Viva Voce held on_____

Examiner – I **Examiner – II**

ACKNOWLEDGEMENT

First of all, we would like to show our sincere gratitude to Prof. **Dr S Vaidhyasubramaniam**, Vice Chancellor, SASTRA Deemed University, who provided all facilities and constant encouragement during our study and we extend our sincere thanks to Prof. **R. Chandra Mouli**, Registrar, SASTRA Deemed University for providing the opportunity to pursue this project. It is our privilege to express our sincerest regards to our project coordinator, **Dr. K. Thenmozhi**, Dean (SEEE), Sridhar K, Associate Dean (ECE) and **Dr. R. John Bosco Balaguru**, Dean (Sponsored Research) who motivated us during the project.

We owe a debt of most profound gratitude to our faculty **Dr. RAJESH A** for his valuable inputs, able guidance, encouragement, wholehearted cooperation throughout our project on the topic 'FUNCTIONAL ANALYSIS OF VANET BASED ON AODV AND DSDV PROTOCOLS USING SUMO AND NS2'.

We take this opportunity to thank all our lecturers who have directly or indirectly helped our project.

I also thank all the Teaching and Non - Teaching faculty, and all other people who have directly or indirectly helped me through their support, encouragement and all other assistance extended for completion of my project and for successful completion of all courses during my academic life at SASTRA Deemed University.

LIST OF CONTENTS

LIST OF FIGURES

NOTATIONS

VANET - Vehicular Ad-hoc Network

AODV - Ad-hoc On-demand Distance Vector

DSDV - Destination Sequenced Distance Vector

DSR - Dynamic Source Routing

SUMO - Simulation of Urban Mobility

NS 2 - Network Simulator

OBU - On Board Unit

AU - Application Unit

RSU - Road Side Unit

V2V - Vehicle to Vehicle communication

V2I - Vehicle-to-road infrastructure

V2B - Vehicle-to-broadband cloud

RREQ - Route Request Packet

RREP - Route Reply Packet

OSM – Open Street Map

ABSTRACT

Vehicular Ad-hoc Networks (VANET) are special type of Mobile Ad-hoc network (MANET), in which vehicles on the road acts as the nodes of the network. Today, VANET finds multiple applications as an Intelligent Transportation System (ITS). VANET's dynamic network architecture and node movement characteristics distinguish it from other types of ad-hoc networks. Routing protocol design in VANET is an important and necessary topic. Dynamic topology changes reduce routing lifetime and therefore routing in VANETs is complex.

Mobility model influences in routing protocols, packet delivery and make a communication. Mobility models therefore play a significant role in influencing the performance of VANET. Thus the main motive of this project is to create a mobility model, based on the Ad hoc on demand Distance Vector routing protocol (AODV) and Destination Sequenced Distance vector routing protocol (DSDV) to make communication between vehicles and analyse their impact on packet delivery, end to end delay, throughput in VANET using SUMO and NS-2.

Keywords: VANET, Routing Protocols: AODV and DSDV, SUMO, NS-2

CHAPTER – 1
INTRODUCTION

1.1 What is VANET?

A Vehicular Ad-hoc Network is a technology in which moving nodes within a network create a mobile network. VANET (Vehicular Ad-hoc Network) is an application of wireless technology used for communication between vehicles and between vehicles with its surrounding environment infrastructure i.e. a Roadside Unit (RSU) in addition to providing drivers and passengers with essential information, this communication method also enables safety applications to improve road safety and provide a comfortable driving experience. User and provider are terms used to describe two different entities. Provider provides the services while the user uses them. Depending on their role in the network, RSUs and OBUs act as providers or users.

A system of VANET includes three main components: an On Board Unit **(OBU)**, an Application Unit **(AU)**, and a Roadside Unit **(RSU)**.

i. **On Board Unit (OBU):**
An On-Board Unit (OBU) is a hardware installed in every vehicle. OBUs are usually mounted on the vehicles for exchanging information with RSUs or with other OBUs. A radio frequency antenna and processor are connected to the transceiver like a router. It not only sends information, but also forwards information to other OBUs. AUs receive support from AUs in the form of service programs. It can support multiple wireless communication technologies as a means of interacting with all external components.

ii. **Application Unit (AU):**
An AU is an in-vehicle device that is used with the application provided by the provider in order to communicate with the OBU. In addition to safety applications, the AU can also run via a regular device; for example, a personal digital assistant (PDA) running online services. OBU and Application Units are linked together using wired or wireless technology. This allows OBU to access the Internet and exchange and receive data.

iii. **Road Side Unit (RSU):**
RSUs are typical devices that are installed in specific positions, such as near parking lots, gas stations etc. Besides providing safety information to the user, it can be used to prevent accidents since the device is connected to the internet. Only authenticated users who have authority can only access the information. We use pseudonyms, mix zones, ad hoc anonymity, and silent periods as techniques.

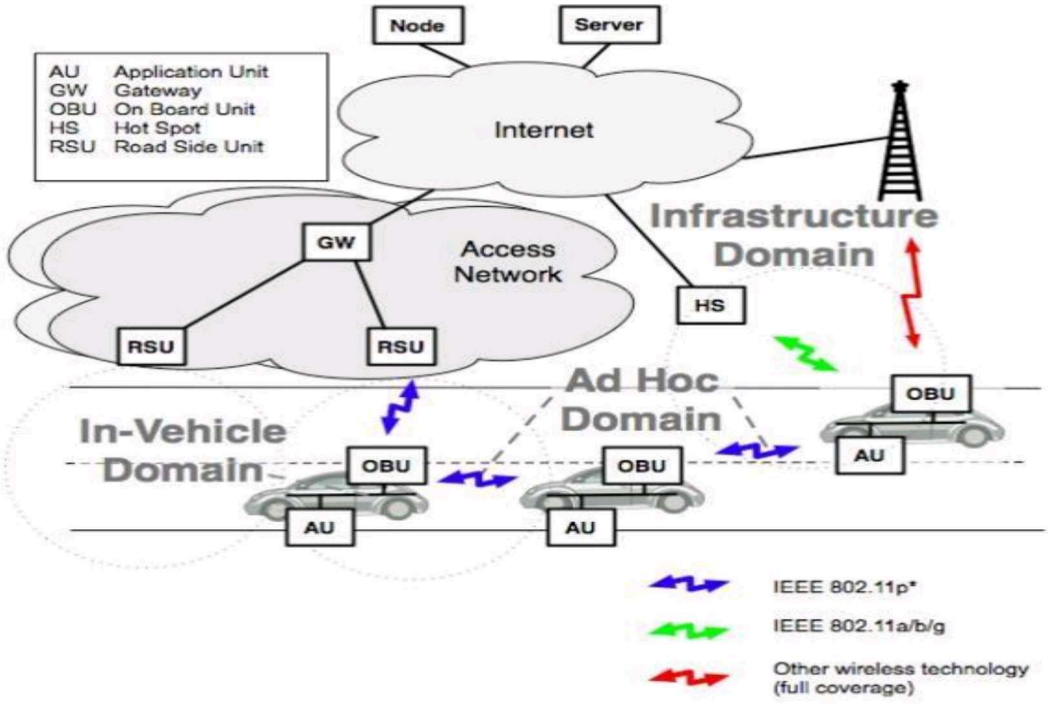


Fig.1 Architecture of VANET is depicted in Figure

1.1 .1 Another form of VANET architecture

The other form of VANET architecture is communication architecture where communication types are characterized into 4 sections which are briefed as:

i. **In-vehicle communication:** It detects the inner system data or performance of the vehicle and determines factors such as driver exhaustion or drowsiness etc. Determination of such factors and their extent is crucial for public safety as well as driver safety

ii. **Vehicle to Vehicle communication (V2V):** Data exchange between different vehicles to support drivers by communicating warnings and other important information to each other. V2V communication does not rely on fixed infrastructure for data exchange and is useful for distribution and security applications.

iii. **Vehicle-to-road infrastructure (V2I) communication:** This communication takes place between moving vehicles and fixed roadside infrastructure in order to gather data. It provides updates related to environmental sensing and monitoring such as real time traffic updates or weather updates.

iv. **Vehicle-to-broadband cloud (V2B) communication:** This allows communication of vehicles over broadband connections such as 3G/4G. This allows the broadband cloud to contain more traffic information and other data, improving driver assistance and vehicle tracking.

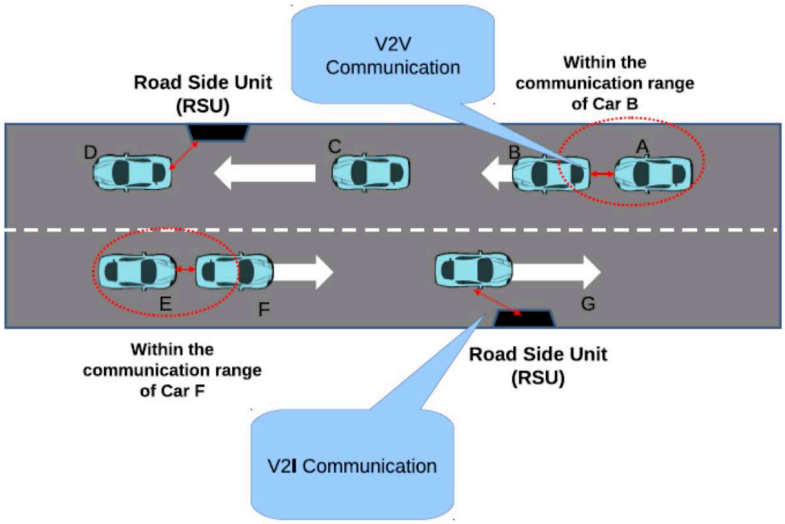


Fig2.VANET Communication architecture

1.2 VANET APPLICATIONS

VANET have a wide range of applications for the safety of drivers and passengers for a better mode of transport. As it's a dynamic network, there are lot of scenarios and use cases possible. The applications are categorized as safety, traffic efficiency and infotainment applications.

1.2.1 Safety Applications

The aim is to warn drivers at the right time about dangerous situations in the road in order to enhance driving safety. Some examples of safety use cases and their related requirements are briefed in the following, we provide a brief description of the main use cases:

- **Cooperative Forward Collision Warning**: the goal of this use case is to assist drivers to prevent rear-end collisions with other vehicles. In fact, rear-end collisions are generally caused by driver disturbance or sudden braking. To avoid accidents, affected vehicles exchange relevant information such as position, speed and direction. The vehicle warns the driver if dangerous situations (e.g. insufficient safety distance) are detected.
- **Pre-Crash Sensing/Warning:** Unlike the Cooperative Forward Collision Warning, this use case assumes that a crash is unavoidable and will take place. In such cases, participating vehicles exchange information with neighboring vehicles in an efficient manner so that vehicle actuators can be used more effectively.
- **Hazardous Location V2V Notification:** the purpose of this use case is to share information about dangerous locations on the road way (like potholes, bottleneck, etc.) between vehicles in certain area. To do this, vehicles that perceive a hazardous location use that information to optimize their safety systems and transmit that information to nearby vehicles in the area. The information is then gradually shared with other affected vehicles through his V2V communications. Information about road hazards can also be sent to RSU by external service providers. RSU transmits information to some vehicles within range. Vehicles that receive the information can then pass the information on to other vehicles via V2V communication.

1.2.2 Traffic efficiency applications

The purpose of the Traffic Efficiency application is to improve the utilization of the transportation system by providing traffic-related information to drivers.
For this purpose, traffic information must be exchanged through the VANET. As a result, road users and road operators each benefit from reduced travel times and reduced road construction and maintenance costs. This application is useful in an emergency as it allows ambulances to find the shortest way to avoid traffic jams and reach their destination. This application could help in emergency cases like Ambulance can find its shortest path avoiding traffic to reach its destination.

- **Enhanced Route Guidance and Navigation:** allows infrastructure owners to collect large area traffic data that can later be used to predict traffic congestion on their roads. The predicted information is sent to the vehicle via RSU. Drivers are thus informed of current and projected traffic conditions across the region, expected delays in reaching their destinations, and better routes that may exist to avoid traffic jams. This will undoubtedly improve the efficiency of the entire transportation system.
- **Green Light Optimal Speed Advisory:** Provides information such as the location of signalized intersections and the signal timing of vehicles approaching intersections (signal switch timing), contributing to smooth driving and stoppage prevention. When the vehicle receives this information at the right time, it can calculate the optimal speed to reach the intersection when the light is green, so the driver does not have to slow down or stop the vehicle. This fact can lead to significant improvements in traffic flow and fuel economy.

1.2.3 Infotainment and others

Use cases not related to safety or traffic efficiency fall into this category. Some of these use cases provide entertainment or information to drivers on a regular basis. Others are transparent to the driver and play an important role in improving vehicle function.

- **Internet Access in Vehicle:** it allows drivers and eventually passengers to access the Internet via the VANET. In this case, RSUs act as internet gateways.
- **Point of Interest Notification:** This allows dealers and advertising agencies to advertise their business promotions to nearby vehicles. For this purpose, RSU sends advertising information (location, opening hours, prices, etc.) to the vehicles in contact. Advertisements received are filtered by each vehicle in relation to the driver's profile and context to display relevant advertisements to the driver.

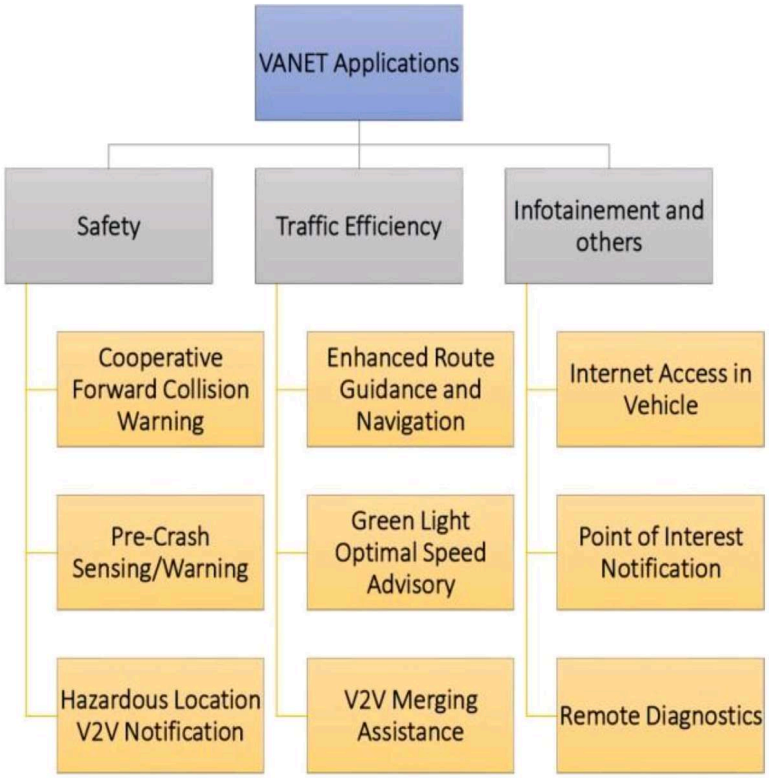


Fig 3. Applications of VANET

CHAPTER – 2
ROUTING PROTOCOLS

The coordinating set of algorithms and rules known as routing protocols are responsible for adjusting transmission to the most optimized path from source to destination. The nature of VANETs network is highly dynamic which makes the routing process challenging and complicated.

Factors that affect routing protocol quality include end-to-end delay, power consumption, and transmission speed. Finding the most optimized and advantageous routing protocol design is an important approach for developing the overall quality of service between nodes in such dynamic vehicular networks. In VANET, the number of nodes in the network is quite large, so storing all possible routes to other nodes is very expensive. Movement of VANETs nodes is restricted by road segments and traffic rules. Localized Routing all nodes participate in forming routing tables for proactive routing protocols. Reactive routing protocols, on the other hand, require all nodes to participate in the initial flooding to find all possible routes to the destination.

For routing protocols to be efficient and adapt to the rapidly changing network nodes positions, the following criteria should be satisfied in VANETs.

- **Low Latency** Security Compliance
- **High Reliability** Achieved by minimizing packet collisions.
- **Flexibility** Measures to provide the highest quality of service while maintaining large vehicle densities and fast-moving responsiveness.
- **Driver Behavior** Network messages directly affect driver response, resulting in changes in network topology. This effect needs to be carefully examined and explained.
- **Comfort Message** Emergency security messages should have a higher priority than other message types and this should be considered when designing routing protocols for VANET.
- **Hierarchical Routing** Minimizing network units into smaller units (clusters) has been shown to minimize routing table size, improve quality of service, reduce latency, and reduce overhead. Hierarchical routing, however, effectively expands the addressing field of a network.

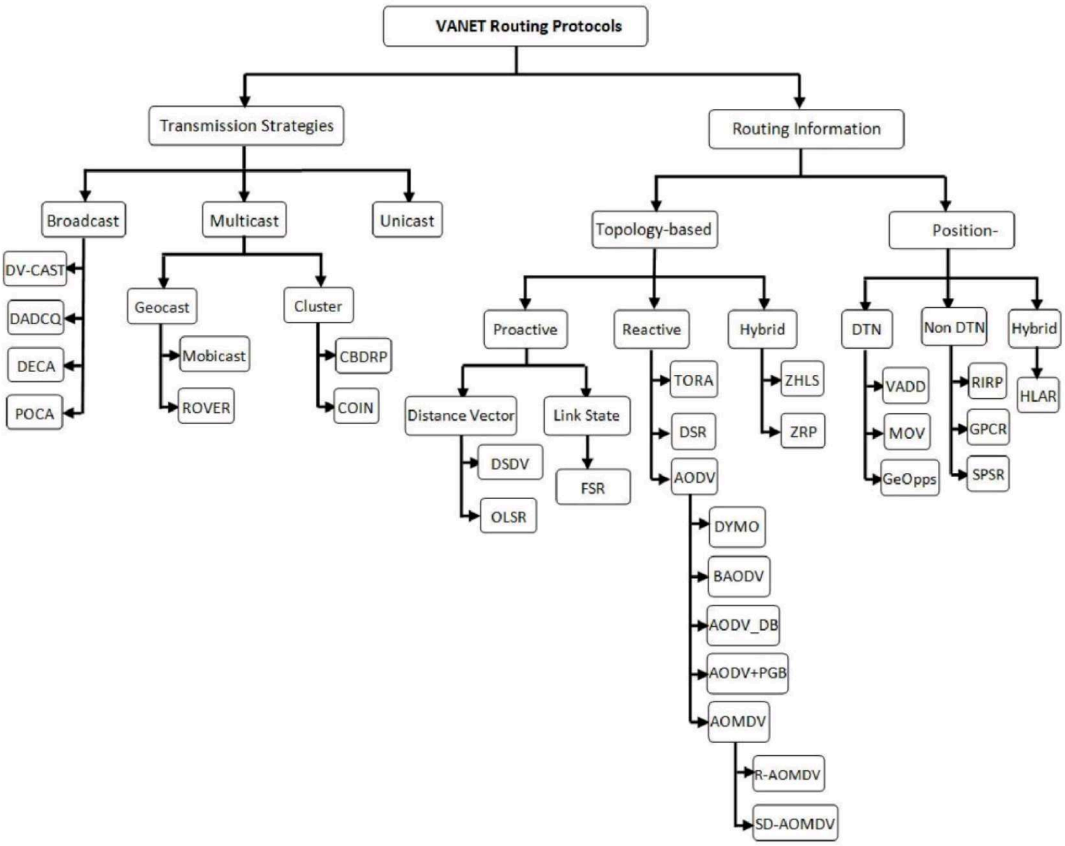


Fig 4. Classification of Routing Protocol

There are numerous routing protocols which can be applied on VANETs. They are classified based on several factors and each uses different kind of algorithms. In this project we will look mainly into 2 routing protocols one from reactive and other from proactive subdivisions of Topology based routing protocols with uses routing information.

2.1 Topology-based protocols

Topology based routing protocols are traditional VANET routing protocol. The main concept is to send datagrams from source to destination using connection information. The protocols are divided into three categories: proactive (table-driven), and reactive (on-demand) and hybrid routing protocols.

2.1.1 Proactive protocols

Proactive protocols use routing tables to select all possible paths to all nodes. Each table shows the current node, next hop, and destination node regardless of whether the route is taken. In addition to computing algorithms, the proactive protocol uses two methods to select the shortest path. They are: Distance vector and Link state.

2.1.1.1 Destination Sequence Distance Vector Routing (DSDV)

The DSDV protocol uses a distance vector strategy in conjunction with the shortest route algorithm (Bellman-Ford algorithm) to find the fastest path through the routing table to the desired destination node. The advantage of DSDV is that it avoids redundant traffic paths with delays, reduces control signalling overhead, and allows route selection to reach each node. DSDV does not use multiple path selection leading to a destination node and cannot handle network congestion. Therefore, it is less efficient overall, and in large networks, the overhead caused by DSDV increases due to the redundancy of updated routing tables, and repeated versions can occur of the previous one. Because of these shortcomings, the Randomized DSDV protocol (R-DSDV) is used to handle network congestion along with DSDV.

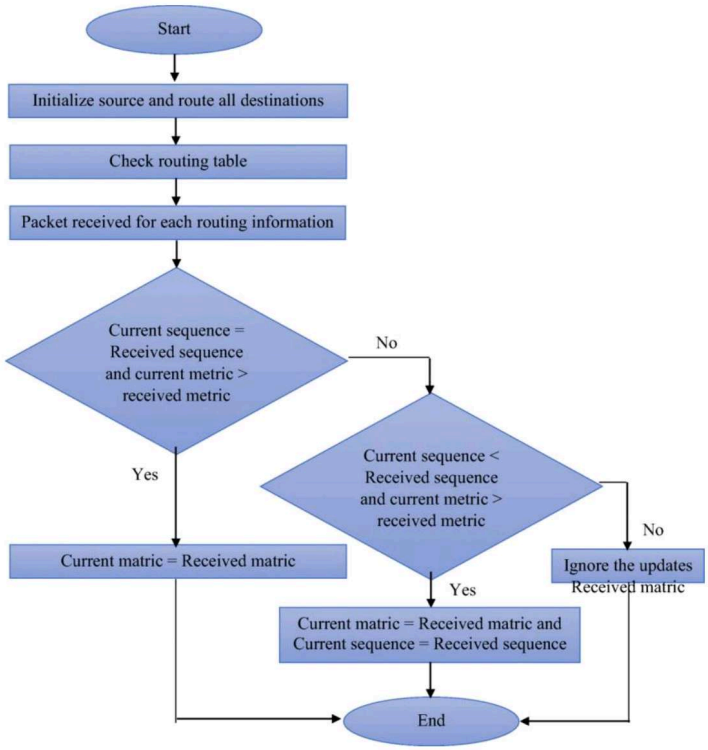


Fig 5. DSDV Routing protocol Algorithm

2.1.2 Reactive Protocols

Reactive routing protocols are also called as on demand routing because routes are used only when they are needed. The entire network will be flooded with request messages, until they reaches the destination node. RRP uses two main methods to determine and maintain routes: Incremental Search Method (ISM) and RRP uses Surroundings Repair Method (SRM). ISM is used instead of broadcast based method to discover new routes and is therefore more bandwidth-efficient. SRM detects link breaks as each node tracks its nexthop neighbour's and enters them into its routing table.

2.1.2.1 Dynamic Source Routing (DSR)

DSR is a multi-hop protocol that initiates a route lookup when a supposed route cannot reach the source. Packets intended to find a route are sent and checked each time they arrive at a new node. This "route request" packet contains a unique identifier and the addresses of the source and destination nodes. If the receiving node cannot forward it, it adds its own address to the packet's route record and forwards it to the adjacent node. Network overhead is reduced because route request packets are inspected only from nodes whose addresses are not stored in the record. If the destination address is found (either directly or through a node that knows about it), the source node follows the sequence of route records to get the response.

2.1.2.2 Ad Hoc on Demand Distance Vector (AODV)

AODV is like a combination of DSDV and DSR protocols but with some contrast. First, data packets only store the address of the destination node, which improves efficiency, especially on sparse and rapidly changing networks. Additionally, AODV's response packets carry an identification number along with the destination node address, which as explained is not the case for DSR. However, this makes AODV slower as the routes are not cached.

The routing table of the AODV protocol contains a set of information that helps in identifying a route from a source to the a destination, such as destination, next hop, number of hops, destination sequence number, active neighbour's for a route and the expiration time (lifetime) for the routing table entry. Route discovery in AODV is done by broadcasting two packets: Route Request Packet (RREQ) and Route Reply Packet (RREP) along with route error message (RREQ).

Routing Request:

When a node in the network needs to send a data packet to it's a destination, a route request packet(RREQ) is flooded throughout the network. An RREQ contains the six fields including source address, request ID, source Sequence number, destination address, destination sequence number and hop count.

Route Reply:

If a node on the network is the destination or if a node on the network has a valid route to the destination, the node sends a reply message (RREP) back to the source. When the source node receives the RREP from the destination node it modifies its routing table and sends the data. If a source node receives multiple route-reply packets from the same destination, the node chooses the shortest route to the destination. In this route discovery operation, the source node sends an RREQ (Route Request Packet) throughout the network.

Each node in the network usually knows about its neighbours by using local broadcasts called HELLO messages. This HELLO message helps in ensure that the next hop is an active route.

Route Error:

Route error messages is used when a node detects a link breakage in an active route. Each node maintains an ancestor list, containing the IP address for each neighbour that are likely to use it as a next hop towards each destination. When a link loss is detected, a RERR message is used to notify other nodes of the link loss.

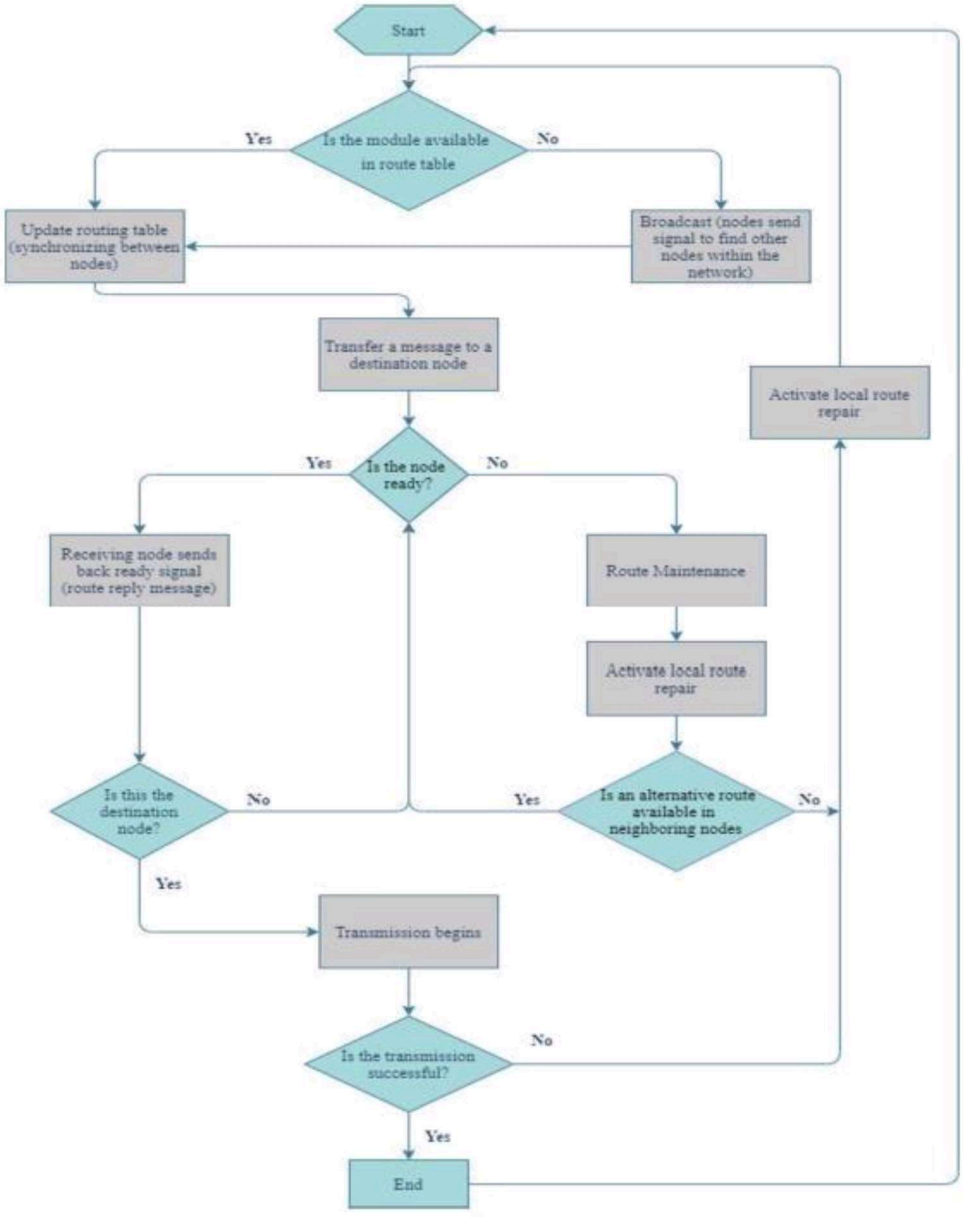


Fig 6. AODV Routing protocol Algorithm

CHAPTER – 3

VANET SIMULATORS

Implementation of VANETs simulation modelling in real-world take lot of energy, resources and cost made it a tough task yet a simpler one for computer software simulators. The nature of VANET simulations are based on simulating mobility patterns, network protocols and wireless channel impairments, complicates the process of simulation.

Simulation of Urban Mobility (SUMO):

SUMO is an open source computer simulator that uses microscopic modelling i.e. vehicles are modelled individually following their own route. SUMO supports continuous and time-discrete vehicle movement, traffic rules, and switchable multi-lane roads. Extra plugins and tools are available through SUMO in order to assist in importing and processing results. Mapfiles format like OpenStreetMaps (OSM) and XML-Descriptions can be imported to SUMO. Following are steps to implement simulation in SUMO:

1. Mobility Model
2. Node (filename.nod.xml)
3. Edge (filename.edge.xml)
4. Type(filename.type.xml)
5. To create a network using the above parameters, we merge all the files using command prompt window by giving the command

   netconvert --node-files vanet.nod.xml --edge-files vanet.edg.xml -t vanet.type.xml -o vanet.net.xml

6. Network (filename.net.xml) - script obtained by combining the node file and edge file along with the type file.
7. Configuration (file.netc.cfg)
8. Create Vehicle (file.rou.xml)
9. Configuration (file.sumo.cfg) - This file is the one running in sumo-gui
10. Visualization (file.sumo.tr) - This file will be helpful in creating the tcl file for ns2.
11. Run Simulation (no of vehicles emitted and running with simulation time will be shown)

Network Simulator Version 2 (NS2):

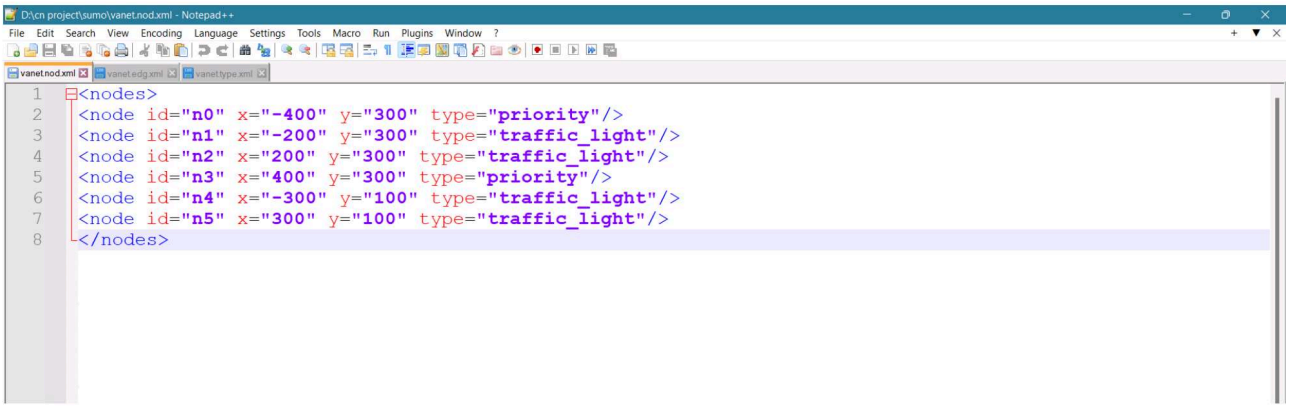
The NS series are discrete network simulators designed primarily for academic research and it can simulate wired and wireless networks. The simulation sequence begins with a map topology definition, defining major obstacles and path obstacles.NS-2 can be used to implement network protocols like UDP and TCP, traffic source behaviour such as FTP, Telnet, Web and router queues management mechanism such as drop tail, CBQ and many more. Model development, including all previously modelled nodes, such as point-to-point device links, node and link configurations, then specifies default values, e.g. size of packets sent, execution, and resulting time. Performance analysis to perform statistical analysis on the stamped data to complete the output and finally the model's graphical visualization using tools such as GNUplot, MATplot and XGRAPH.

CHAPTER – 4

SCRIPTS AND NS-2 SOURCE CODES

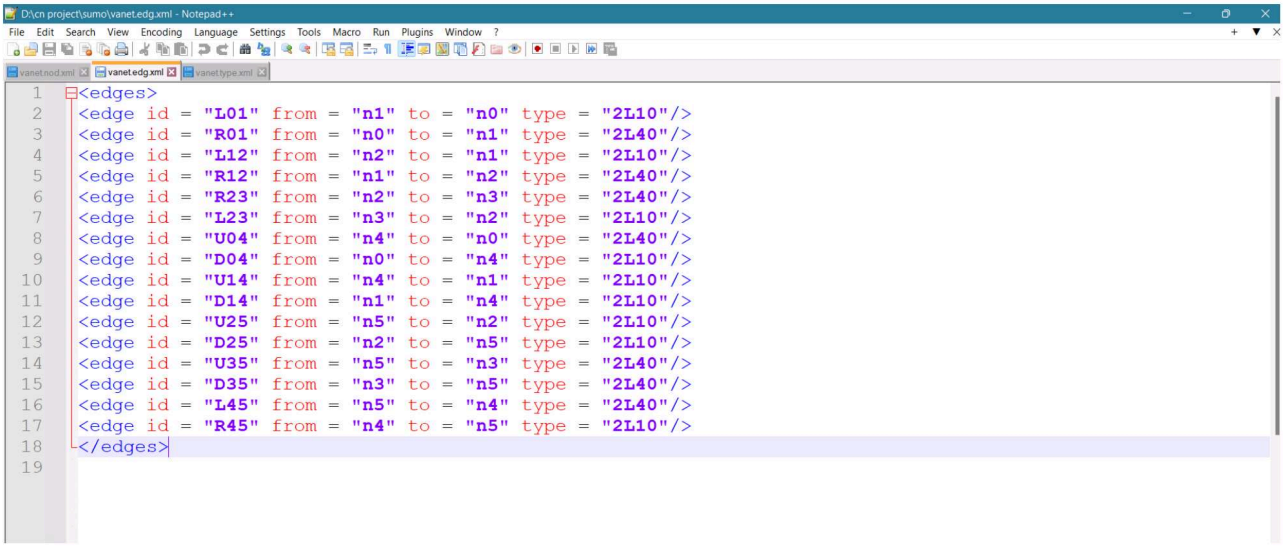
4.1 SUMO script:

4.1.1. Node File (vanet.nod.xml)



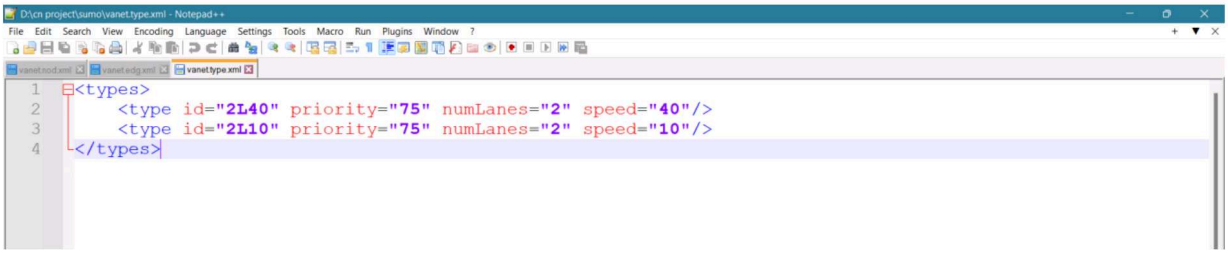

```
<nodes>
<node id="n0" x="-400" y="300" type="priority"/>
<node id="n1" x="-200" y="300" type="traffic_light"/>
<node id="n2" x="200" y="300" type="traffic_light"/>
<node id="n3" x="400" y="300" type="priority"/>
<node id="n4" x="-300" y="100" type="traffic_light"/>
<node id="n5" x="300" y="100" type="traffic_light"/>
</nodes>
```

4.1.2. Egde file (vanet.edge.xml)




```
<edges>
<edge id = "L01" from = "n1" to = "n0" type = "2L10"/>
<edge id = "R01" from = "n0" to = "n1" type = "2L40"/>
<edge id = "L12" from = "n2" to = "n1" type = "2L10"/>
<edge id = "R12" from = "n1" to = "n2" type = "2L40"/>
<edge id = "R23" from = "n2" to = "n3" type = "2L40"/>
<edge id = "L23" from = "n3" to = "n2" type = "2L10"/>
<edge id = "U04" from = "n4" to = "n0" type = "2L40"/>
<edge id = "D04" from = "n0" to = "n4" type = "2L10"/>
<edge id = "U14" from = "n4" to = "n1" type = "2L10"/>
<edge id = "D14" from = "n1" to = "n4" type = "2L10"/>
<edge id = "U25" from = "n5" to = "n2" type = "2L10"/>
<edge id = "D25" from = "n2" to = "n5" type = "2L10"/>
<edge id = "U35" from = "n5" to = "n3" type = "2L40"/>
<edge id = "D35" from = "n3" to = "n5" type = "2L40"/>
<edge id = "L45" from = "n5" to = "n4" type = "2L40"/>
<edge id = "R45" from = "n4" to = "n5" type = "2L10"/>
</edges>
```

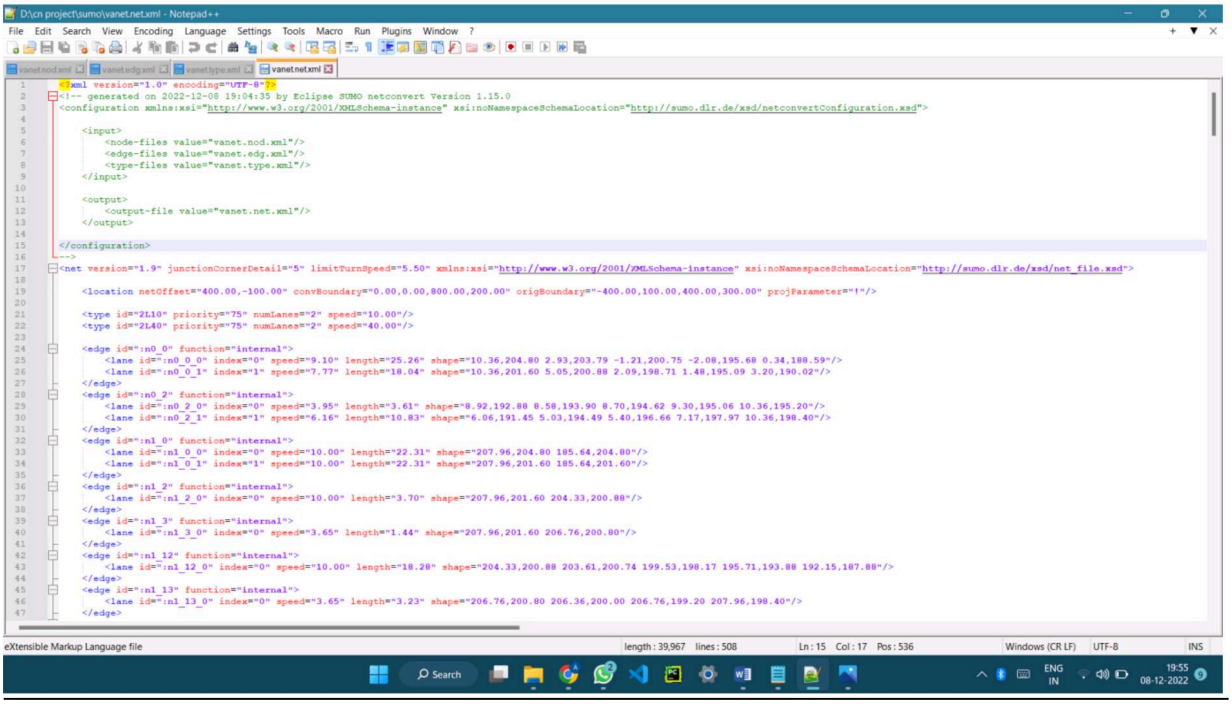
4.1.3. Type file(vanet.type.xml)



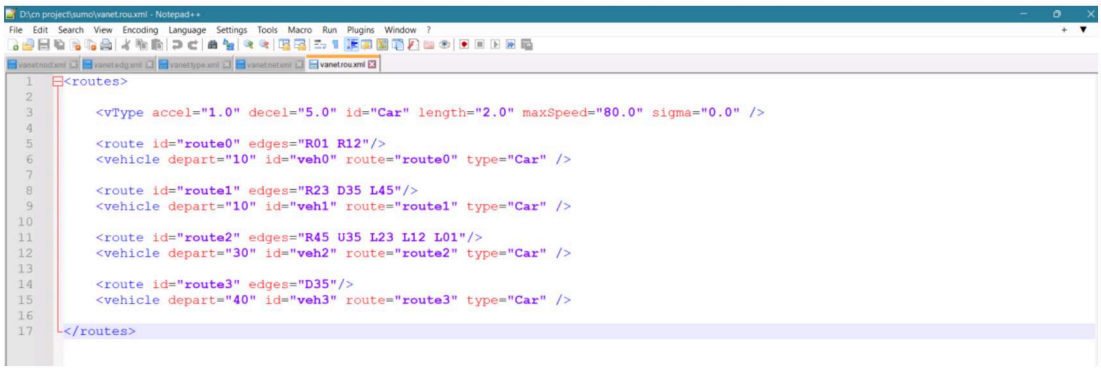
In command prompt give below command to generates a network file.

netconvert --node-files vanet.nod.xml --edge-files vanet.edg.xml -t vanet.type.xml -o vanet.net.xml

4.1.4 Network file (vanet.net.xml)

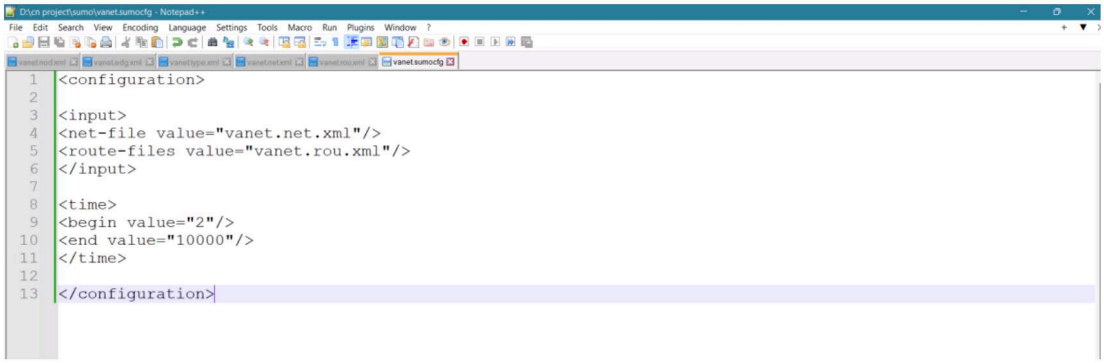


4.1.5. Route file (vanet.rou.xml)



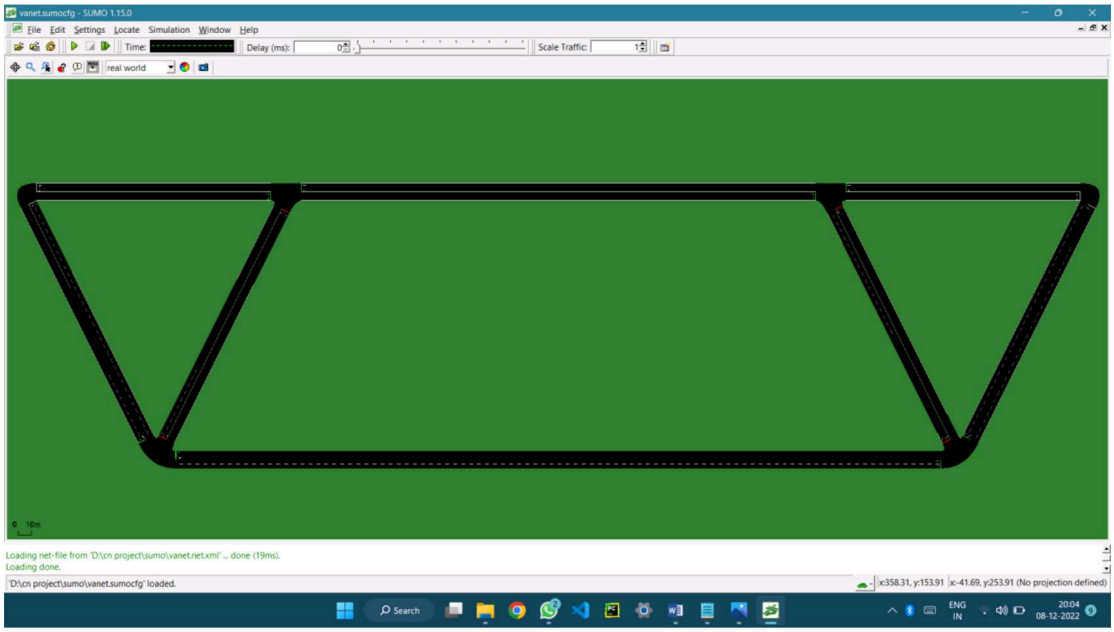
```
<routes>

    <vType accel="1.0" decel="5.0" id="Car" length="2.0" maxSpeed="80.0" sigma="0.0" />

    <route id="route0" edges="R01 R12"/>
    <vehicle depart="10" id="veh0" route="route0" type="Car" />

    <route id="route1" edges="R23 D35 L45"/>
    <vehicle depart="10" id="veh1" route="route1" type="Car" />

    <route id="route2" edges="R45 U35 L23 L12 L01"/>
    <vehicle depart="30" id="veh2" route="route2" type="Car" />

    <route id="route3" edges="D35"/>
    <vehicle depart="40" id="veh3" route="route3" type="Car" />

</routes>
```

4.1.6. Configuration file (vanet.sumo.cfg)



```
<configuration>

<input>
<net-file value="vanet.net.xml"/>
<route-files value="vanet.rou.xml"/>
</input>

<time>
<begin value="2"/>
<end value="10000"/>
</time>

</configuration>
```

4.1.7.Now open the SUMO file generated by configuration file. It directly opens the simulation in SUMO GUI.



Run the simulation with delay 90ms and observed the movement of vehicles from one node to another and movement of vehicles based on traffic signals.4

4.2.1 At node 1 (JUNCTION 1):

4.2.2 At node 4 (JUNCTION 4):

Fig7. snapshots of SUMO GUI

4.3 Exporting to NS-2:

1. The network setup was done in SUMO GUI for a better understanding purpose of movement of vehciles from node to node and we can observe traffic simulation in this GUI.

2. We need to export this sumo file to NS-2 for analysing the network simulation

3. Open terminal and type the following commands
4.

sumo –c vanet.sumo.cfg –fcd-output vanet.sumo.xml

5. Copy the sumo xml files to ubuntu, give the path and type the following command to get tcl files

--fcd-input vanet.sumo.xml –ns2config-output vanet.tcl –ns2activity-output acticity.tcl –ns2mobility-output mobility.tcl

6. This will generate 3 tcl files: vanet.tcl, activity.tcl and mobility.tcl

7. The generated tcl file is thus modified as per the networking paramters like routing protocols,Mac layer, physical layer and link layer.

8. Given below the vanet.tcl file by changing routing parameters to AODV and DSDV and simulation parameters are mention in upcoming pages.

4.3.1 IMPLEMENTATION OF AODV ROUTING PROTOCOLS:

```
# This script is created by NSG2 beta1
#===================================
#   Simulation parameters setup
#===================================
set val(chan)   Channel/WirelessChannel    ;# channel type
set val(prop)   Propagation/TwoRayGround   ;# radio-propagation
model
set val(netif)  Phy/WirelessPhy            ;# network interface type
set val(mac)    Mac/802_11                 ;# MAC type
set val(ifq)    Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)     LL                         ;# link layer type
set val(ant)    Antenna/OmniAntenna        ;# antenna model
set val(ifqlen) 50                         ;# max packet in ifq
set val(nn)     17                         ;# number of mobilenodes
set val(rp)     AODV                       ;# routing protocol
set val(x)      904                        ;# X dimension of
topography
set val(y)      878                        ;# Y dimension of
topography
set val(stop)   25.0                       ;# time of simulation end
#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]
#Setup topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
#Open the NS trace file
set tracefile [open AODV.tr w]
$ns trace-all $tracefile
#Open the NAM trace file
set namfile [open AODV.nam w]
$ns namtrace-all $namfile
$ns namtrace-all-wireless $namfile $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel
#===================================
#     Mobile node parameter setup
#===================================
$ns node-config -adhocRouting  $val(rp) \
                -llType        $val(ll) \
                -macType       $val(mac) \
                -ifqType       $val(ifq) \
                -ifqLen        $val(ifqlen) \
                -antType       $val(ant) \
                -propType      $val(prop) \
                -phyType       $val(netif) \
```

```
                -channel        $chan \
                -topoInstance   $topo \
                -agentTrace     ON \
                -routerTrace    ON \
                -macTrace       ON \
                -movementTrace ON
#=================================
#        Nodes Definition
#=================================
#Create nodes
set n0 [$ns node]
$n0 set X_ 150
$n0 set Y_ 766
$n0 set Z_ 0.0
$ns initial_node_pos $n0 20
set n1 [$ns node]
$n1 set X_ 282
$n1 set Y_ 653
$n1 set Z_ 0.0
$ns initial_node_pos $n1 20
set n2 [$ns node]
$n2 set X_ 145
$n2 set Y_ 638
$n2 set Z_ 0.0
$ns initial_node_pos $n2 20
set n3 [$ns node]
$n3 set X_ 211
$n3 set Y_ 493
$n3 set Z_ 0.0
$ns initial_node_pos $n3 20
set n4 [$ns node]
$n4 set X_ 414
$n4 set Y_ 502
$n4 set Z_ 0.0
$ns initial_node_pos $n4 20
set n5 [$ns node]
$n5 set X_ 465
$n5 set Y_ 639
$n5 set Z_ 0.0
$ns initial_node_pos $n5 20
set n6 [$ns node]
$n6 set X_ 301
$n6 set Y_ 778
$n6 set Z_ 0.0
$ns initial_node_pos $n6 20
set n7 [$ns node]
$n7 set X_ 488
$n7 set Y_ 764
```

```
$n7 set Z_ 0.0
$ns initial_node_pos $n7 20
set n8 [$ns node]
$n8 set X_ 641
$n8 set Y_ 758
$n8 set Z_ 0.0
$ns initial_node_pos $n8 20
set n9 [$ns node]
$n9 set X_ 616
$n9 set Y_ 637
$n9 set Z_ 0.0
$ns initial_node_pos $n9 20
set n10 [$ns node]
$n10 set X_ 740
$n10 set Y_ 652
$n10 set Z_ 0.0
$ns initial_node_pos $n10 20
set n11 [$ns node]
$n11 set X_ 558
$n11 set Y_ 475
$n11 set Z_ 0.0
$ns initial_node_pos $n11 20
set n12 [$ns node]
$n12 set X_ 683
$n12 set Y_ 531
$n12 set Z_ 0.0
$ns initial_node_pos $n12 20
set n13 [$ns node]
$n13 set X_ 804
$n13 set Y_ 498
$n13 set Z_ 0.0
$ns initial_node_pos $n13 20
set n14 [$ns node]
$n14 set X_ 346
$n14 set Y_ 352
$n14 set Z_ 0.0
$ns initial_node_pos $n14 20
set n15 [$ns node]
$n15 set X_ 572
$n15 set Y_ 346
$n15 set Z_ 0.0
$ns initial_node_pos $n15 20
set n16 [$ns node]
$n16 set X_ 720
$n16 set Y_ 355
$n16 set Z_ 0.0
$ns initial_node_pos $n16 20
#=================================
```

```
#        Generate movement
#=================================
$ns at 1.0` " $n2 setdest 500 300 10 "
$ns at 10.0` " $n2 setdest 600 500 30 "
$ns at 1.0 " $n11 setdest 363 287 30 "
$ns at 8.0 " $n11 setdest 695 220 25 "
#=================================
#        Agents Definition
#=================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n3 $tcp0
set sink2 [new Agent/TCPSink]
$ns attach-agent $n8 $sink2
$ns connect $tcp0 $sink2
$tcp0 set packetSize_ 1500

#Setup a TCP connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n14 $tcp1
set sink3 [new Agent/TCPSink]
$ns attach-agent $n10 $sink3
$ns connect $tcp1 $sink3
$tcp1 set packetSize_ 1500
#=================================
#      Applications Definition
#=================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 1.0 "$ftp0 start"
$ns at 15.0 "$ftp0 stop"
#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns at 2.0 "$ftp1 start"
$ns at 20.0 "$ftp1 stop"
#=================================
#      Termination
#=================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam AODV.nam &
    exit 0
```

```
}
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns at $val(stop) "\$n$i reset"
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

4.3.2 IMPLEMENTATION OF DSDV ROUTING PROTOCOL:

```
#==================================
#     Simulation parameters setup
#==================================
set val(chan)   Channel/WirelessChannel    ;# channel type
set val(prop)   Propagation/TwoRayGround   ;# radio-propagation
model
set val(netif)  Phy/WirelessPhy            ;# network interface type
set val(mac)    Mac/802_11                 ;# MAC type
set val(ifq)    Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)     LL                         ;# link layer type
set val(ant)    Antenna/OmniAntenna        ;# antenna model
set val(ifqlen) 50                         ;# max packet in ifq
set val(nn)     17                         ;# number of mobilenodes
set val(rp)     DSDV                       ;# routing protocol
set val(x)      904                        ;# X dimension of
topography
set val(y)      878                        ;# Y dimension of
topography
set val(stop)   25.0                       ;# time of simulation end
#==================================
#        Initialization
#==================================
#Create a ns simulator
set ns [new Simulator]
#Setup topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
#Open the NS trace file
set tracefile [open DSDV.tr w]
$ns trace-all $tracefile
#Open the NAM trace file
set namfile [open DSDV.nam w]
$ns namtrace-all $namfile
$ns namtrace-all-wireless $namfile $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel
#==================================
#     Mobile node parameter setup
#==================================
$ns node-config -adhocRouting  $val(rp) \
                -llType        $val(ll) \
                -macType       $val(mac) \
                -ifqType       $val(ifq) \
                -ifqLen        $val(ifqlen) \
                -antType       $val(ant) \
                -propType      $val(prop) \
                -phyType       $val(netif) \
                -channel       $chan \
```

```
                -topoInstance  $topo \
                -agentTrace    ON \
                -routerTrace   ON \
                -macTrace      ON \
                -movementTrace ON
#=================================
#        Nodes Definition
#=================================
#Create nodes
set n0 [$ns node]
$n0 set X_ 150
$n0 set Y_ 766
$n0 set Z_ 0.0
$ns initial_node_pos $n0 20
set n1 [$ns node]
$n1 set X_ 282
$n1 set Y_ 653
$n1 set Z_ 0.0
$ns initial_node_pos $n1 20
set n2 [$ns node]
$n2 set X_ 145
$n2 set Y_ 638
$n2 set Z_ 0.0
$ns initial_node_pos $n2 20
set n3 [$ns node]
$n3 set X_ 211
$n3 set Y_ 493
$n3 set Z_ 0.0
$ns initial_node_pos $n3 20
set n4 [$ns node]
$n4 set X_ 414
$n4 set Y_ 502
$n4 set Z_ 0.0
$ns initial_node_pos $n4 20
set n5 [$ns node]
$n5 set X_ 465
$n5 set Y_ 639
$n5 set Z_ 0.0
$ns initial_node_pos $n5 20
set n6 [$ns node]
$n6 set X_ 301
$n6 set Y_ 778
$n6 set Z_ 0.0
$ns initial_node_pos $n6 20
set n7 [$ns node]
$n7 set X_ 488
$n7 set Y_ 764
$n7 set Z_ 0.0
```

```
$ns initial_node_pos $n7 20
set n8 [$ns node]
$n8 set X_ 641
$n8 set Y_ 758
$n8 set Z_ 0.0
$ns initial_node_pos $n8 20
set n9 [$ns node]
$n9 set X_ 616
$n9 set Y_ 637
$n9 set Z_ 0.0
$ns initial_node_pos $n9 20
set n10 [$ns node]
$n10 set X_ 740
$n10 set Y_ 652
$n10 set Z_ 0.0
$ns initial_node_pos $n10 20
set n11 [$ns node]
$n11 set X_ 558
$n11 set Y_ 475
$n11 set Z_ 0.0
$ns initial_node_pos $n11 20
set n12 [$ns node]
$n12 set X_ 683
$n12 set Y_ 531
$n12 set Z_ 0.0
$ns initial_node_pos $n12 20
set n13 [$ns node]
$n13 set X_ 804
$n13 set Y_ 498
$n13 set Z_ 0.0
$ns initial_node_pos $n13 20
set n14 [$ns node]
$n14 set X_ 346
$n14 set Y_ 352
$n14 set Z_ 0.0
$ns initial_node_pos $n14 20
set n15 [$ns node]
$n15 set X_ 572
$n15 set Y_ 346
$n15 set Z_ 0.0
$ns initial_node_pos $n15 20
set n16 [$ns node]
$n16 set X_ 720
$n16 set Y_ 355
$n16 set Z_ 0.0
$ns initial_node_pos $n16 20
#=================================
#        Generate movement
```

```
#=================================
$ns at 1.0` " $n2 setdest 500 300 10 "
$ns at 10.0` " $n2 setdest 600 500 30 "
$ns at 1.0 " $n11 setdest 363 287 30 "
$ns at 8.0 " $n11 setdest 695 220 25 "
#=================================
#        Agents Definition
#=================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n3 $tcp0
set sink2 [new Agent/TCPSink]
$ns attach-agent $n8 $sink2
$ns connect $tcp0 $sink2
$tcp0 set packetSize_ 1500
#Setup a TCP connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n14 $tcp1
set sink3 [new Agent/TCPSink]
$ns attach-agent $n10 $sink3
$ns connect $tcp1 $sink3
$tcp1 set packetSize_ 1500
#=================================
#        Applications Definition
#=================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 1.0 "$ftp0 start"
$ns at 15.0 "$ftp0 stop"
#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns at 2.0 "$ftp1 start"
$ns at 20.0 "$ftp1 stop"
#=================================
#        Termination
#=================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam DSDV.nam &
    exit 0
}
for {set i 0} {$i < $val(nn) } { incr i } {
```

```
    $ns at $val(stop) "\$n$i reset"
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

4.3.3 Analysis.awk

```
BEGIN{
      recv_size = o
      sTime = 1e6
        spTime = 0
        NumofRecd = 0
}

{
    event = $1
    time = $3
    node_id = $5
    packet = $19
    pkt_id = $41
    flow_id = $39
    packet_size = $37
    flow_type = $45

    if (packet == "AGT" && sendTime[pkt_id] == 0 && (event == "+" ||
event == "s")){
        if (time < sTime){
            sTime = time;
        }
        sendTime[pkt_id] = time
        this_flow = flow_type
    }
    if (packet = "AGT" && event = "r"){
        if (time > spTime){
            spTime = time;
        }
        recv_size = recv_size + packet_size
        recvTime[pkt_id] = time
        NumofRecd = NumofRecd + 1
    }
}
END{
    if (NumofRecd == 0){
        printf("no packets recieved")
    }
    printf("start time %d\n",sTime)
    printf("stop time %d\n",spTime)
    printf("recieved packet time %d\n",NumofRecd)
    printf("throughput in kbps %f\n", (NumofRecd/(spTime-
sTime)*(8/1000)))
}
```

CHAPTER-5

NS-2 SIMULATION SET-UP AND SNAPSHOTS:

NS2 configuration parameters used for the test are listed below:

NS2 version	2.35
Parameter	Value
Initial simulation time	300sec
Channel Type	wireless
Network Interface Type	Wireless
MAC Layer Protocol	IEEE 802.11p
Mobiles Nodes	16
Traffic Type	TCP
Packet Size	512 bytes
Number of lanes	2

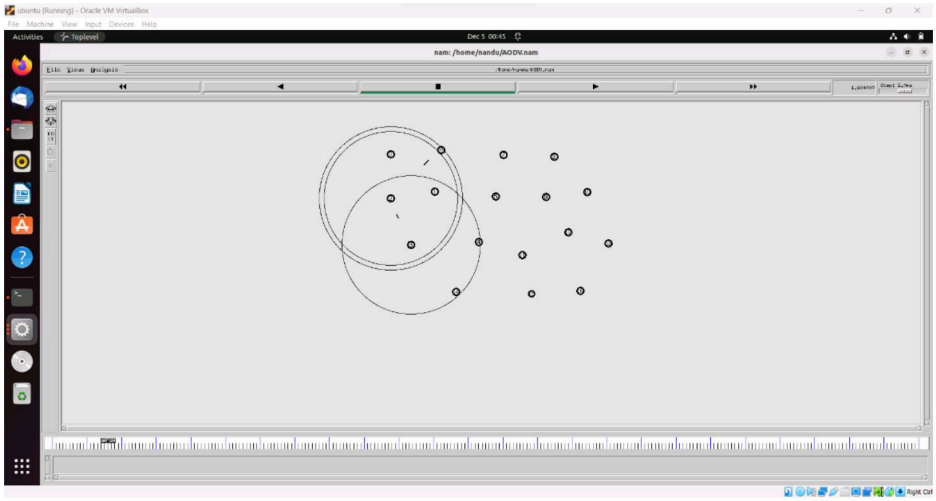


Fig 8. In middle of simulation in NS-2

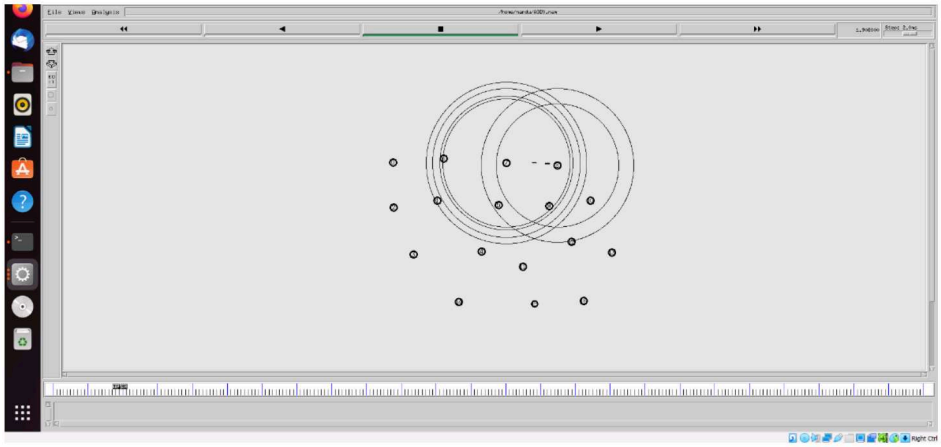


Fig 9. In middle of simulation in NS-2

CHAPTER – 6

RESULTS AND PERFORMANCE METRIC

6.1 PARAMETERS:

1. **Throughput:** The total number of transmitted packets that can reach and be received at the destination node. Throughput is measured in data packets/second or bytes/sec. The simulation result show the total number of packets (KB/sec) received at the destination.

   Throughput = (Total number of packets * Packet Size in bits) / (Sum of all Delay)

2. **End to End Delay:** Calculated as the time taken for a packet to travel from the source node to the destination node, or the total time it takes for a packet to travel from the source end to the destination end.

   Average end to end delay = (Sum of all end-to-end delays)/ (Total number of packets)

3. **Packet Delivery Ratio**: PDR can be described as the ratio of data packets actually received by the receiver to the data packets originally sent by the sender.

   Pdr = (Number of packets delivered successfully)/ (Total number of packets)

6.2 GRAPH GENERATION:

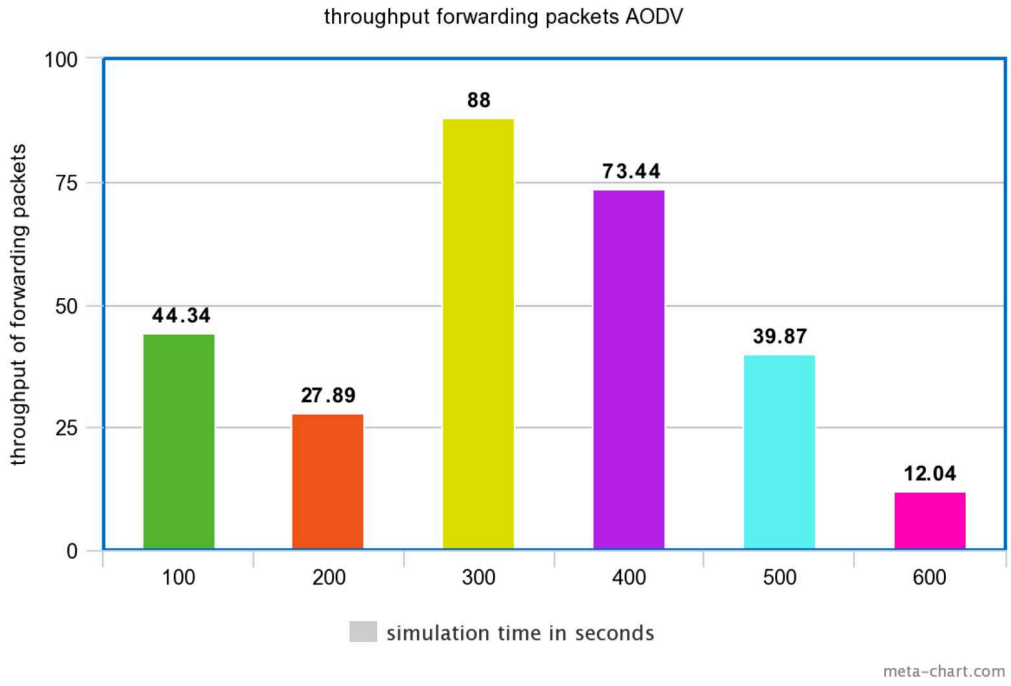
1. By using the source code i.e analysis.awk script we generate the throughput calculations.
2. Give the following command in terminal
   sudo ./ns graph tcl
3. XGraphes are generated and throughput values are observed.
4. Values at 300 sec of simulation with packet size of 512 bytes are tabulated below for both AODV and DSDV protocols.

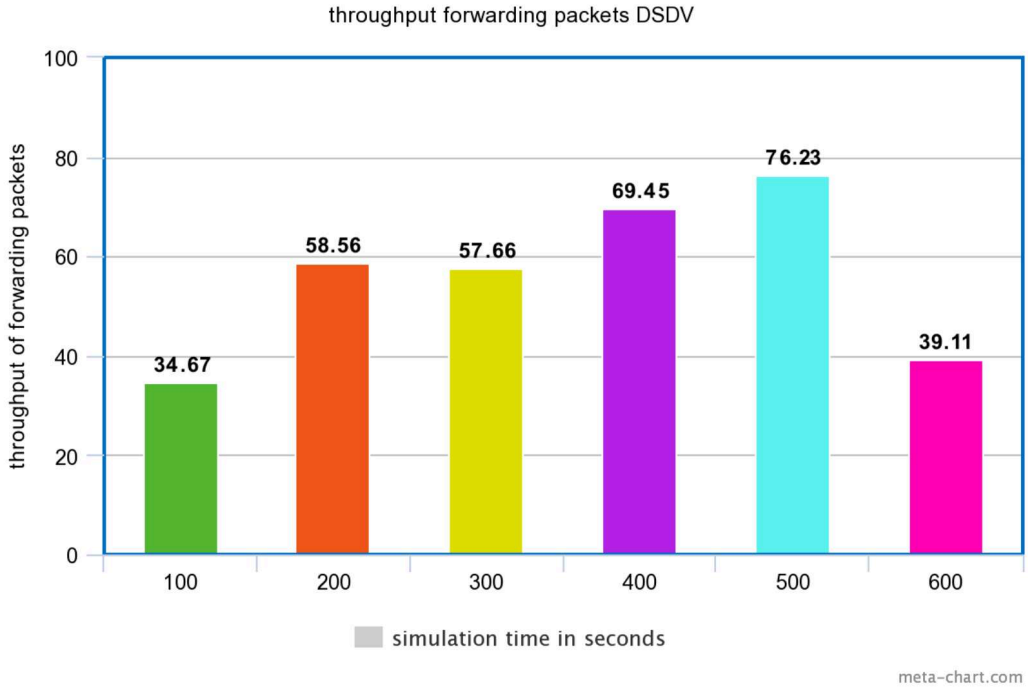
	Throughput	Average end-end delay	pdr
AODV	88.00	0.324	98.41
DSDV	57.66	0.075	97.56

Table : Packet size=512bytes, time=300sec

Following graphs show values of throughput and average end to end delay obtained at different simulation time for both AODV and DSDV

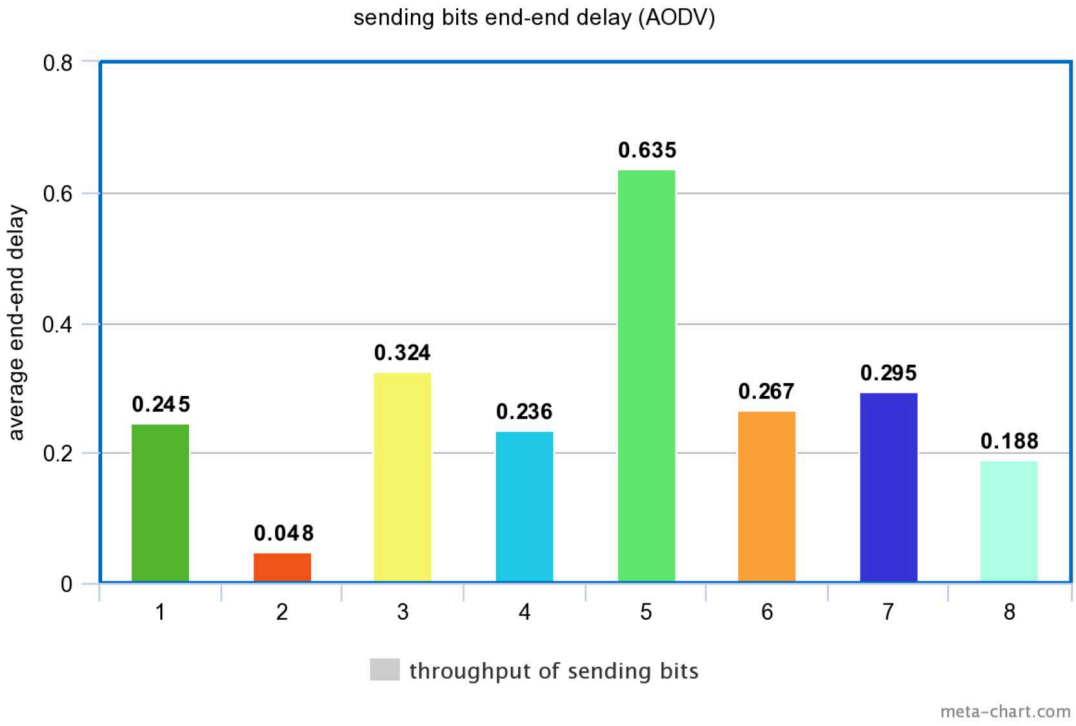
6.3 Throughput: AODV vs DSDV

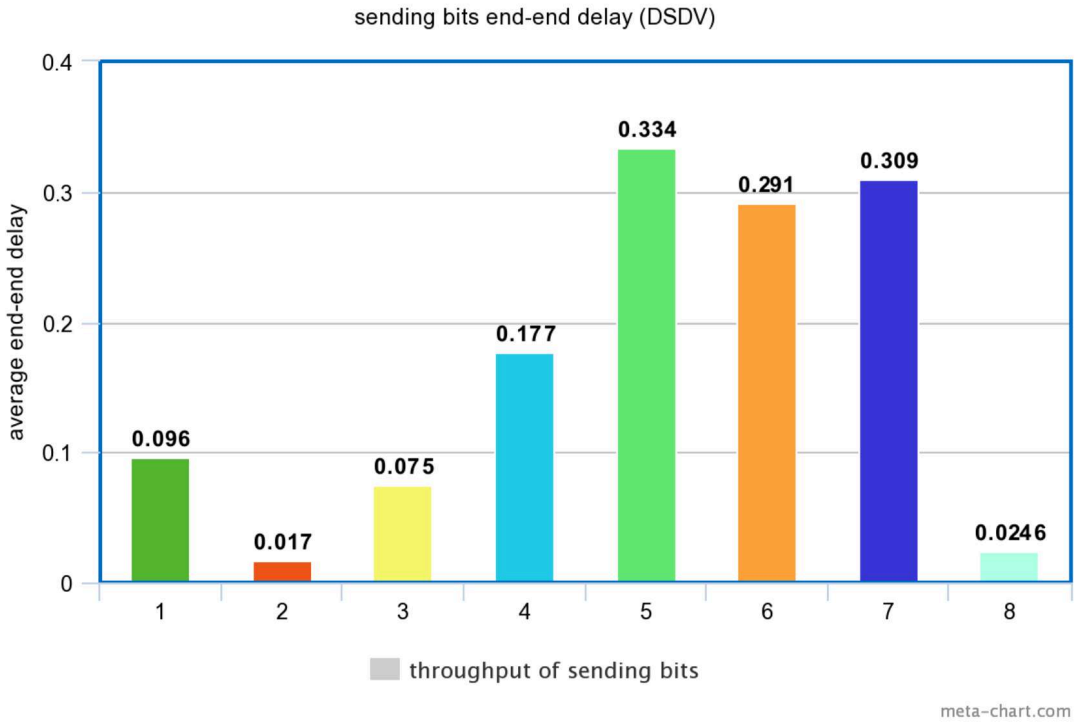




From the above obtained throughput values for a different time period of simulation, it can be observed that AODV is better DSDV as DSDV routing table broadcast periodically at regular intervals.

6.4 Delay Calculation: (AODV vs DSDV)





From the above graphs, we can conclude that AODV protocol have delay higher than DSDV. This is because AODV is a reactive routing protocol that will react if the source node was changed.

CHAPTER – 7

CONCLUSION

The main agenda of this project is to perform functional analysis on topology based routing protocols AODV and DSDV and evaluate these routing protocols with different parameters in VANET. In this project we used SUMO along with NS2 simulator to simulate AODV, DSDV routing protocols in realistic mobility model. We focused only on topology based routing protocols. We have investigated how different topology routing protocols affect the high mobility of VANETs.

After the research, I found that the throughput is better with AODV than with DSDV whereas AODV protocol have delay higher than DSDV.

CHAPTER – 8

FUTURE PLANS

In VANET, due to high mobility of nodes, the network route path changes frequently and it depends on urban road infrastructure. So it is necessary to consider a realistic and specific road map. Future studies will involve increasing network scale and simulating real-world traffic to evaluate the applications of VANET based on this. The performance of routing protocols on real time traffic can be evaluated by utilising OpenStreetMap, SUMO GUI and then exporting it to NS-2.

CHAPTER – 9

REFERENCES

1. Abhishek Singh and Anil. K. Verma, “Performance Analysis of AODV and DSDV in Simulation Based Map,” International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Volume 2, Issue 6, June 2013.

2. C. Namitel, G.Vishal," Comparative Analysis of AODV, DSR and DSDV Routing Protocols for VANET City Scenario” International Journal on Recent and Innovation Trends in Computing and Communication, Volume-2, Issue-6, June 2014

3. S. R. Das, R. Castaneda, and J. Yan, “Simulation based performance evaluation of mobile ad hoc network routing protocols,” ACM/Baltzer Mobile Networks and Applications (MONET) Journal, vol. 5, Issue 3, pp.179- 189, Sep 2000

4. C. E. Perkins, E. M. Belding-Royer, and S. Das, “Ad hoc on-demand distance vector (AODV) routing,” Network Working Group, RFC 3561, July 2003.

5. Simulation of Urban Mobility (SUMO)," [Online]. Available: http://sumo.sourceforge.net

6. Tajinder Kaur, A. K. Verma, “Simulation and Analysis of AODV routing protocol in VANET”, International Journal of Soft Computing and Engineering (IJSCE), Volume-2, Issue-3, July 2012

7. Ahmed, E., & Gharavi, H. (2018). Cooperative vehicular networking: A survey. IEEE Transactions on Intelligent Transportation Systems, 19(3), 996-1014.

8. Alasmary,W.; Zhuang,W.Mobility impact in IEEE 802.11p infrastructure less vehicular networks. Ad Hoc Netw.2012, 10, 222–230.

9. NAM(Network Animator) http://www.isi.edu/nsnam/nam

10. Marwa Altayeb, and Imad Mahgoub. 2013. "A Survey of Vehicular Ad hoc Networks Routing Protocols." International Journal of Innovation and Applied Studies (Innovative Space of Scientific Research Journals) 3 (3): 829-846. http://www.issrjournals.org/ijias/