**NS2 WIRELESS NETWORK SIMULATION**

A PROJECT REPORT

***Submitted by***

BL.EN.U4AIE19007 APOORVA.M

BL.EN.U4AIE19041 TANUJ.M

BL.EN.U4AIE19068 AISHWARYA.V

***for the course***

***19AIE302- Advanced Computer Networks***

***Guided and Evaluated by***

***Mr.Rajesh M***

**IN**

**ARTIFICIAL INTELLIGENCE ENGINEERING**



AMRITA SCHOOL OF ENGINEERING, BANGALORE

**BANGALORE 560 035**

**PROBLEM STATEMENT**

**Question 2:**

Implement the new routing protocol which modifies the existing AODV protocol.

The change required is in the packet format: An extra information: Packet number – a value assigned when a packet is send need to be included in the packet. Follow the steps mentioned in the document provided modifying AODV protocol.

To verify the working of modified protocol, simulate a wireless network consisting of 6 mobile nodes (n0 – n5). Use the modified routing protocol.

Note:

1. Develop the codes, test it.
2. Submit the code, a report consisting of the problem statement, details and results snapshots.
3. Prepare a PPT consisting of the details.

**THEORY**

AODV Routing Protocol in NS2 – Ad Hoc On-Demand Distance Vector is a routing protocol for ad hoc mobile networks with large numbers of mobile nodes. ADHOC creates routes between nodes only when the routes are requested by the source nodes. Ad Hoc Network gives the network flexibility to enter or leave the network at their own will.

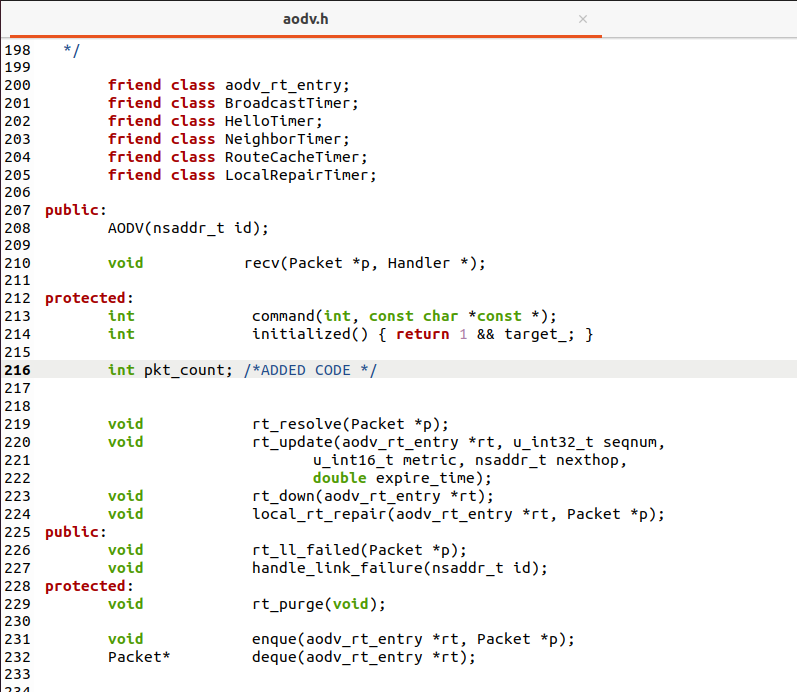
The AODV protocol builds routes between nodes only if they are requested by source nodes. AODV is therefore considered an on-demand algorithm and does not create any extra traffic for communication along links. The routes are maintained as long as they are required by the sources. They also form trees to connect multicast group members. AODV makes use of sequence numbers to ensure route freshness. They are self-starting and loop-free besides scaling to numerous mobile nodes.

In AODV, networks are silent until connections are established. Network nodes that need connections broadcast a request for connection. The remaining AODV nodes forward the message and record the node that requested a connection. Thus, they create a series of temporary routes back to the requesting node. A node that receives such messages and holds a route to a desired node sends a backward message through temporary routes to the requesting node. The node that initiated the request uses the route containing the least number of hops through other nodes. The entries that are not used in routing tables are recycled after some time. If a link fails, the routing error is passed back to the transmitting node and the process is repeated.

The AODV convention creates connections between hubs only if source hubs mention them. As a result, AODV is considered an on-request calculation, and no additional traffic is created for correspondence along joins. The experts keep the courses up to date as required. They also build tree structures to link multicast bunch members. To ensure course freshness, AODV employs grouping numbers. Apart from scaling to various portable hubs, they are self-starting and circle open. Networks in AODV are silent until associations are formed. Organization hubs that use associations send out a request for membership. The extra AODV hubs are directed to the information and document hub that listed a link. They create a sequence of impermanent courses that lead back to the mentioning hub in this way.

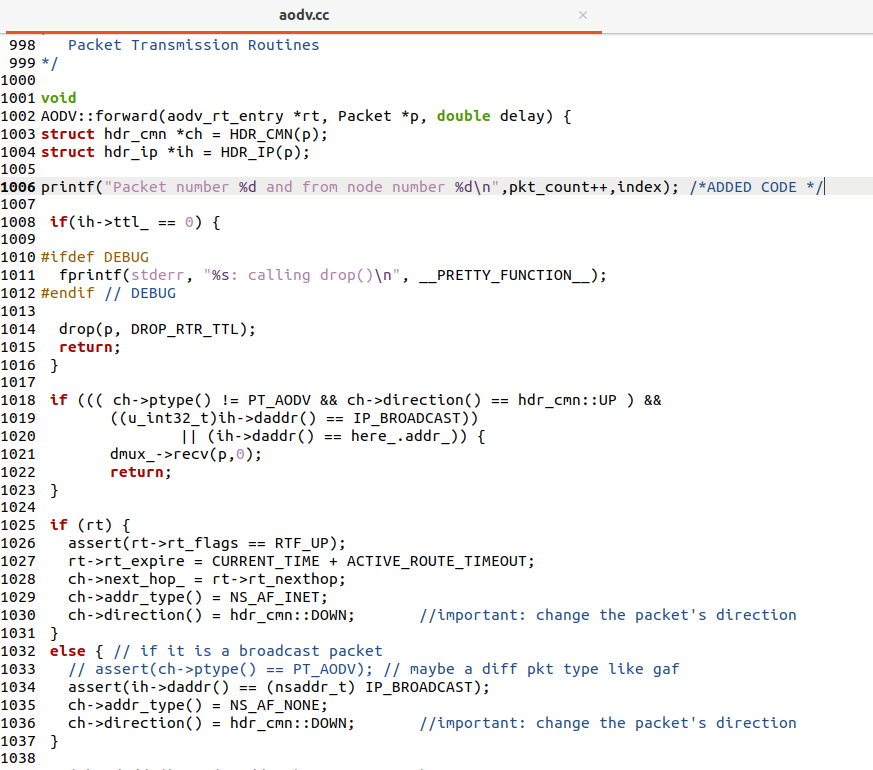
**SCRIPT AND RESULTS:**

**Added changes in (aodv.h) inbuilt code:**

****

**Added changes in (aodv.cc) inbuilt code:**

****

****

**TCL SCRIPT:**

# Define options

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) 6 ;# number of mobilenodes

set val(rp) AODV ;# routing protocol

set val(x) 400 ;# X dimension of topography

set val(y) 400 ;# Y dimension of topography

set val(stop) 150 ;# time of simulation end

set ns [new Simulator]

set tracefd [open project.tr w]

set namtrace [open simwrls.nam w]

$ns trace-all $tracefd

$ns namtrace-all-wireless $namtrace $val(x) $val(y)

# set up topography object

set topo [new Topography]

$topo load\_flatgrid $val(x) $val(y)

create-god $val(nn)

# configure the nodes

$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) \

-channelType $val(chan) \

-topoInstance $topo \

-agentTrace ON \

-routerTrace ON \

-macTrace OFF \

-movementTrace ON

for {set i 0} {$i < $val(nn) } { incr i } {

set n($i) [$ns node]

}

# Provide initial location of mobilenodes

$n(0) set X\_ 40.0

$n(0) set Y\_ 280.0

$n(0) set Z\_ 0.0

$n(1) set X\_ 80.0

$n(1) set Y\_ 340.0

$n(1) set Z\_ 0.0

$n(2) set X\_ 90.0

$n(2) set Y\_ 230.0

$n(2) set Z\_ 0.0

$n(3) set X\_ 170.0

$n(3) set Y\_ 340.0

$n(3) set Z\_ 0.0

$n(4) set X\_ 200.0

$n(4) set Y\_ 200.0

$n(4) set Z\_ 0.0

$n(5) set X\_ 260.0

$n(5) set Y\_ 350.0

$n(5) set Z\_ 0.0

# Set a TCP connection between n(1) and n(31)

set tcp [new Agent/TCP/Newreno]

$tcp set class\_ 2

$tcp set packetSize\_ 1000

set sink [new Agent/TCPSink]

$ns attach-agent $n(2) $tcp

$ns attach-agent $n(3) $sink

$ns connect $tcp $sink

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ns at 10.0 "$ftp start"

$ns at 125.0 "$ftp stop"

set tcp2 [new Agent/TCP/Newreno]

$tcp2 set class\_ 2

$tcp2 set packetSize\_ 600

set sink2 [new Agent/TCPSink]

$ns attach-agent $n(1) $tcp2

$ns attach-agent $n(4) $sink2

$ns connect $tcp2 $sink2

set ftp2 [new Application/FTP]

$ftp2 attach-agent $tcp2

$ns at 50.0 "$ftp2 start"

#defining heads

$ns at 0.0 "$n(0) label CH"

$ns at 0.0 "$n(1) label Source2"

$ns at 0.0 "$n(2) label Source1"

$ns at 0.0 "$n(4) label Destination2"

$ns at 0.0 "$n(3) label Destination1"

$ns at 100.0 "$n(5) setdest 385.0 228.0 5.0"

$ns at 60.0 "$n(2) setdest 200.0 20.0 5.0"

$ns at 30.0 "$n(3) setdest 115.0 85.0 5.0"

$ns at 45.0 "$n(1) setdest 375.0 80.0 5.0"

$ns at 89.0 "$n(4) setdest 167.0 351.0 5.0"

$ns at 78.0 "$n(0) setdest 50.0 359.0 5.0"

#Color change while moving from one group to another

$ns at 73.0 "$n(2) delete-mark N2"

$ns at 73.0 "$n(2) add-mark N2 pink circle"

$ns at 124.0 "$n(3) delete-mark N11"

$ns at 124.0 "$n(3) add-mark N11 purple circle"

$ns at 87.0 "$n(4) delete-mark N26"

$ns at 87.0 "$n(4) add-mark N26 yellow circle"

$ns at 92.0 "$n(1) delete-mark N14"

$ns at 92.0 "$n(1) add-mark N14 green circle"

# Define node initial position in nam

for {set i 0} {$i < $val(nn)} { incr i } {

# 20 defines the node size for nam

$ns initial\_node\_pos $n($i) 20

}

# Telling nodes when the simulation ends

for {set i 0} {$i < $val(nn) } { incr i } {

$ns at $val(stop) "$n($i) reset";

}

# ending nam and the simulation

$ns at $val(stop) "$ns nam-end-wireless $val(stop)"

$ns at $val(stop) "stop"

$ns at 150.01 "puts \"end simulation\" ; $ns halt"

proc stop {} {

global ns tracefd namtrace

$ns flush-trace

close $tracefd

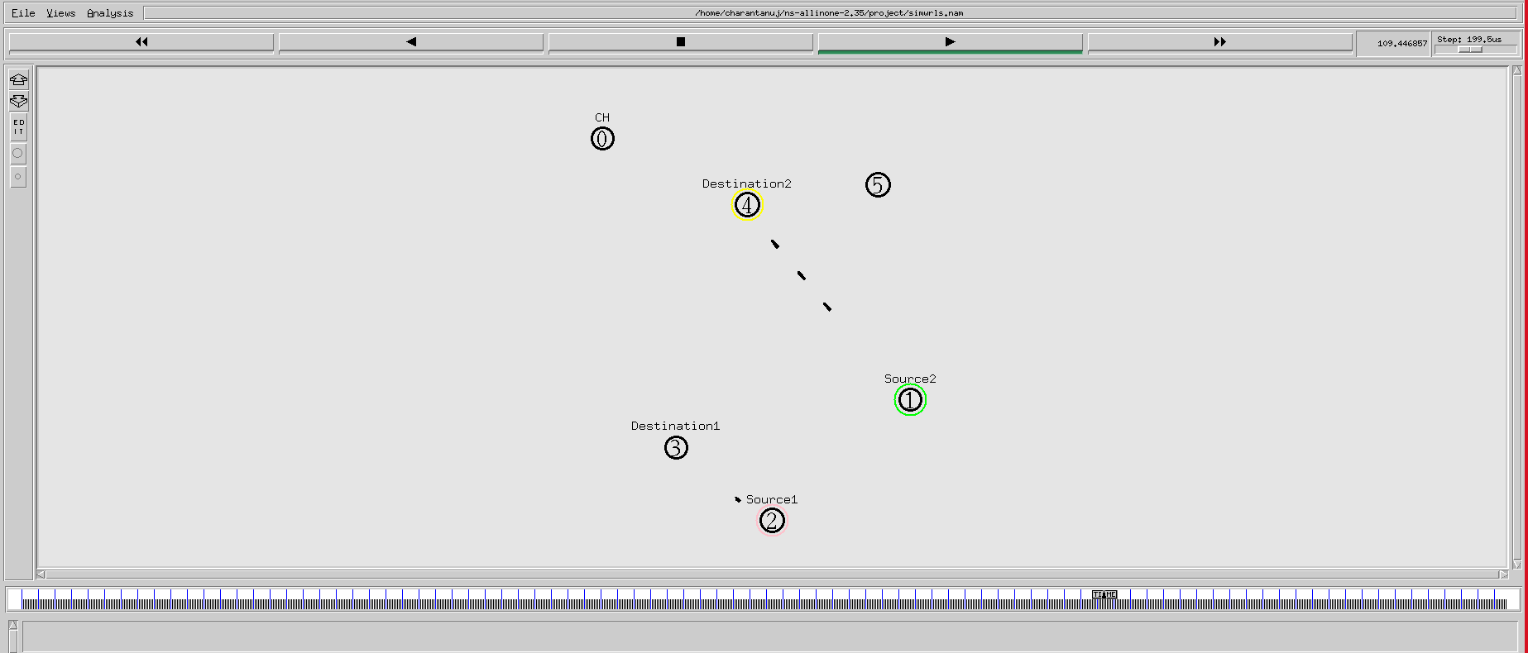
close $namtrace

exec nam simwrls.nam &

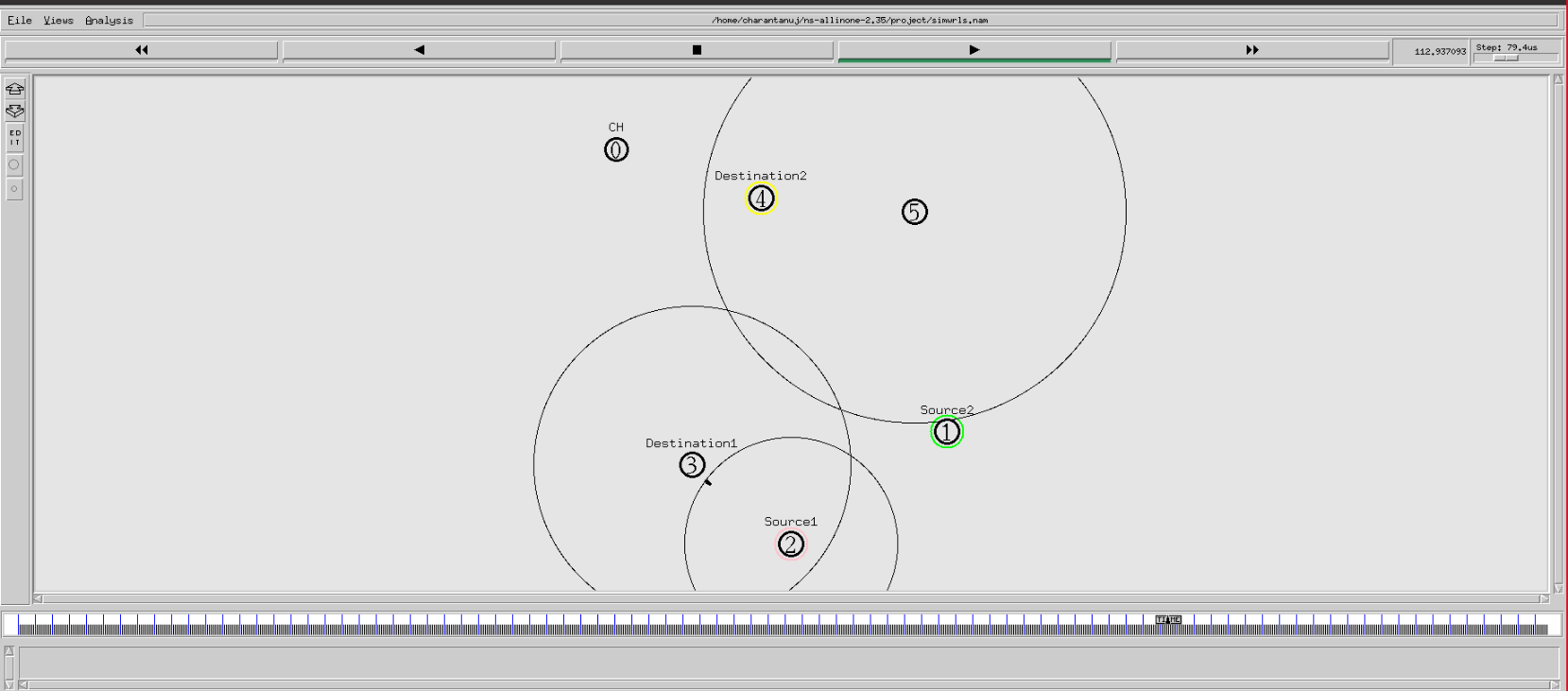
}

$ns run

**Screenshots of NAM Output:**

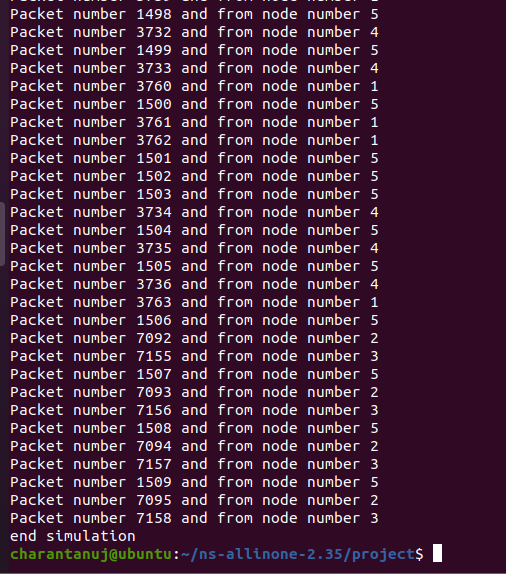


**Fig 1**



**Fig 2**

**Screenshots of TERMINAL Output:**

****

**Fig 3**