**EXPERIMENT 8**

**Aim :**

Simulate the effect of using Different Routing Protocols on Network.

**Theory :**

**Distance Vector Routing Algorithm**

Distance Vector Routing is one of the routing algorithm in a Wide Area Network for computing shortest path between source and destination. The Router is one main devices used in a wide area network. The main task of the router is Routing. It forms the routing table and delivers the packets depending upon the routes in the table - either directly or via an intermediate devices.

Each router initially has information about its all neighbours. Then this information will be shared among nodes.

**Dynamic Source Routing Algorithm**

Dynamic Source Routing (DSR) comes under the reactive routing protocol category, as it is capable of discovering the route from source to destination only when required and needed, this protocol uses a process called “Route Discovery Mechanism” that is capable of discovering the route for data packets from source node to destination nodes using intermediate nodes. As like proactive routing protocols such as Global State Routing and Dynamic Sequence Distance Vector Routing no separate table is maintained.

The major change in DSR as compare to GSR and DSDV is, in DSDV after asking a requirement of route from source to destination, path via intermediate nodes is checked for its length. Then a “Re - Request” packet is sent back from destination to source via the smallest route possible in the whole network. The “Re - Request” packet does contains its unique ID also.

**Link State Routing Algorithm**

In Link State Routing, each router shares its knowledge of its neighbourhood with every other router in the internet work.

**(i) Knowledge about Neighbourhood :** instead of sending its entire routing table a router sends info about its neighbourhood only.

**(ii) To all Routers :** each router sends this information to every other router on the internet work not just to its neighbour. It does so by a process called flooding.

**(iii) Information sharing when there is a change :** each router sends out information about the neighbours when there is change.

**# Distance Vector Routing Algorithm**

**Code :**

set ns [new Simulator]

set nr [open aftabdvr.tr w]

$ns trace-all $nr

set nf [open aftabdvr.nam w]

$ns namtrace-all $nf

proc finish { } {

global ns nr nf

$ns flush-trace

close $nf

close $nr

exec nam aftabdvr.nam &

exit 0

}

for { set i 0 } { $i < 12} { incr i 1 } {

set n($i) [$ns node]

}

for {set i 0} {$i < 8} {incr i} {

$ns duplex-link $n($i) $n([expr $i+1]) 1Mb 10ms DropTail

}

$ns duplex-link $n(0) $n(8) 1Mb 10ms DropTail

$ns duplex-link $n(1) $n(10) 1Mb 10ms DropTail

$ns duplex-link $n(0) $n(9) 1Mb 10ms DropTail

$ns duplex-link $n(9) $n(11) 1Mb 10ms DropTail

$ns duplex-link $n(10) $n(11) 1Mb 10ms DropTail

$ns duplex-link $n(11) $n(5) 1Mb 10ms DropTail

set udp0 [new Agent/UDP]

$ns attach-agent $n(0) $udp0

set cbr0 [new Application/Traffic/CBR]

$cbr0 set packetSize\_ 500

$cbr0 set interval\_ 0.005

$cbr0 attach-agent $udp0

set null0 [new Agent/Null]

$ns attach-agent $n(5) $null0

$ns connect $udp0 $null0

set udp1 [new Agent/UDP]

$ns attach-agent $n(1) $udp1

set cbr1 [new Application/Traffic/CBR]

$cbr1 set packetSize\_ 500

$cbr1 set interval\_ 0.005

$cbr1 attach-agent $udp1

set null0 [new Agent/Null]

$ns attach-agent $n(5) $null0

$ns connect $udp1 $null0

$ns rtproto DV

$ns rtmodel-at 10.0 down $n(11) $n(5)

$ns rtmodel-at 15.0 down $n(7) $n(6)

$ns rtmodel-at 30.0 up $n(11) $n(5)

$ns rtmodel-at 20.0 up $n(7) $n(6)

$udp0 set fid\_ 1

$udp1 set fid\_ 2

$ns color 1 Red

$ns color 2 Green

$ns at 1.0 "$cbr0 start"

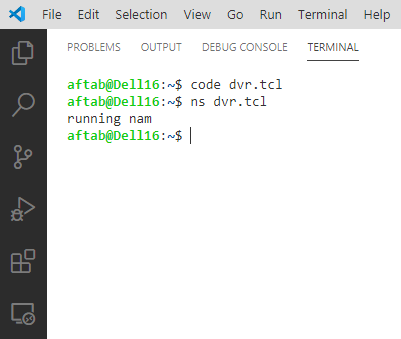
$ns at 2.0 "$cbr1 start"

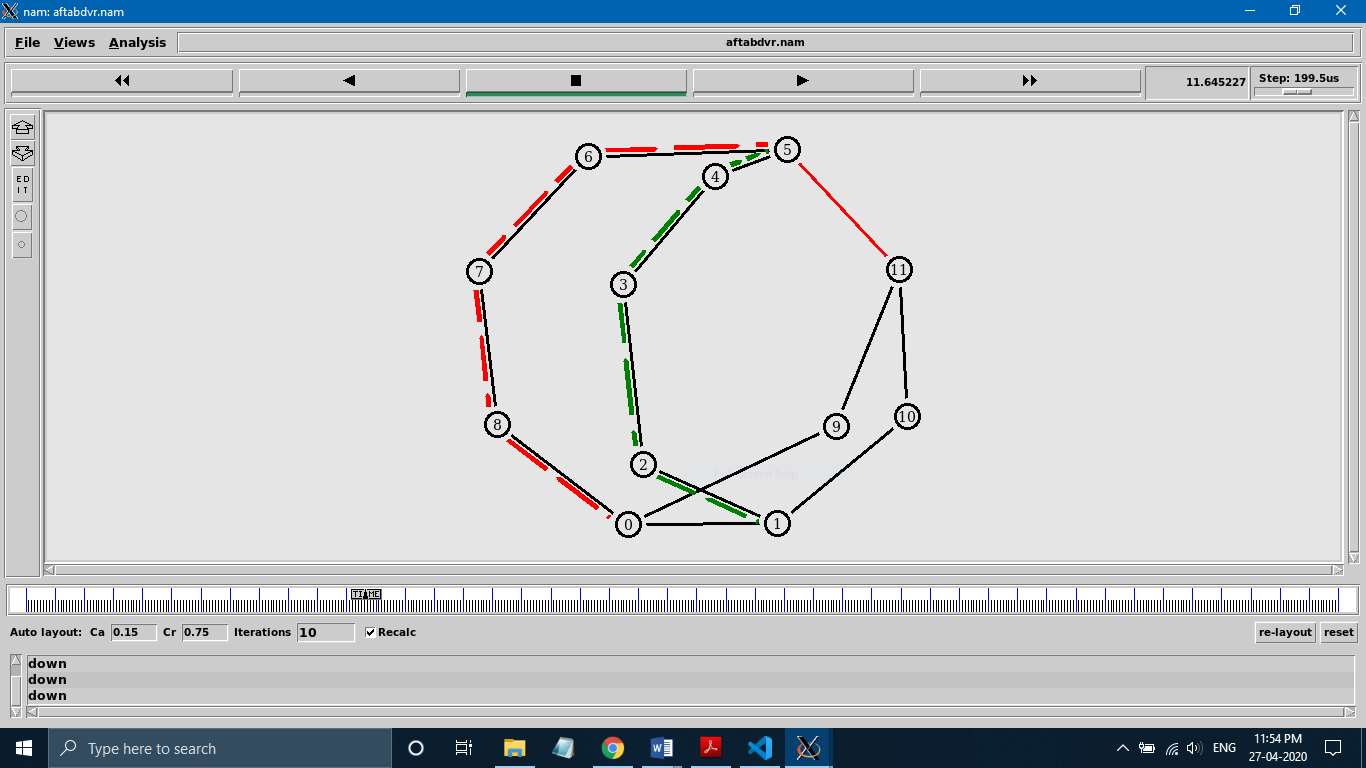
$ns at 45 "finish"

puts "running nam"

$ns run

**Screen Shots :**





**# Dynamic Source Routing Algorithm**

**Code :**

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) CMUPriQueue; #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) 10; #number of mobilenodes

set val(rp) DSR; #routing protocol

set val(x) 500; #X dimension of topography

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

set val(stop) 50; #time of simulation end

set ns [new Simulator]

set tracefd [open simple-dsdv.tr w]

set windowVsTime2 [open win.tr w]

set namtrace [open aftabdsr.nam w]

$ns trace-all $tracefd

$ns use-newtrace

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

set topo [new Topography]

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

create-god $val(nn)

$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 node\_($i) [$ns node] }

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

$node\_($i) set X\_ [expr rand()\*500]

$node\_($i) set Y\_ [expr rand()\*400]

$node\_($i) set Z\_ 0 }

set tcp [new Agent/TCP/Newreno]

$tcp set class\_ 2

set sink [new Agent/TCPSink]

$ns attach-agent $node\_(0) $tcp

$ns attach-agent $node\_(9) $sink

$ns connect $tcp $sink

set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ns at 2.0 "$ftp start"

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

$ns initial\_node\_pos $node\_($i) 30 }

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

$ns at $val(stop) "$node\_($i) reset"; }

$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

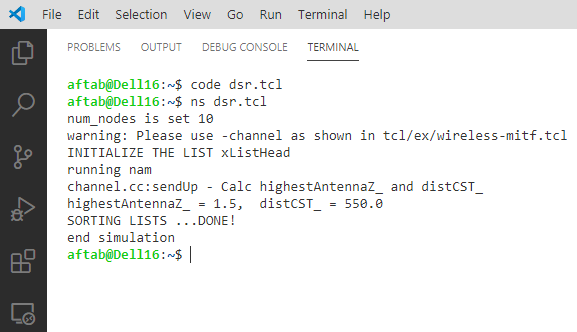
close $tracefd

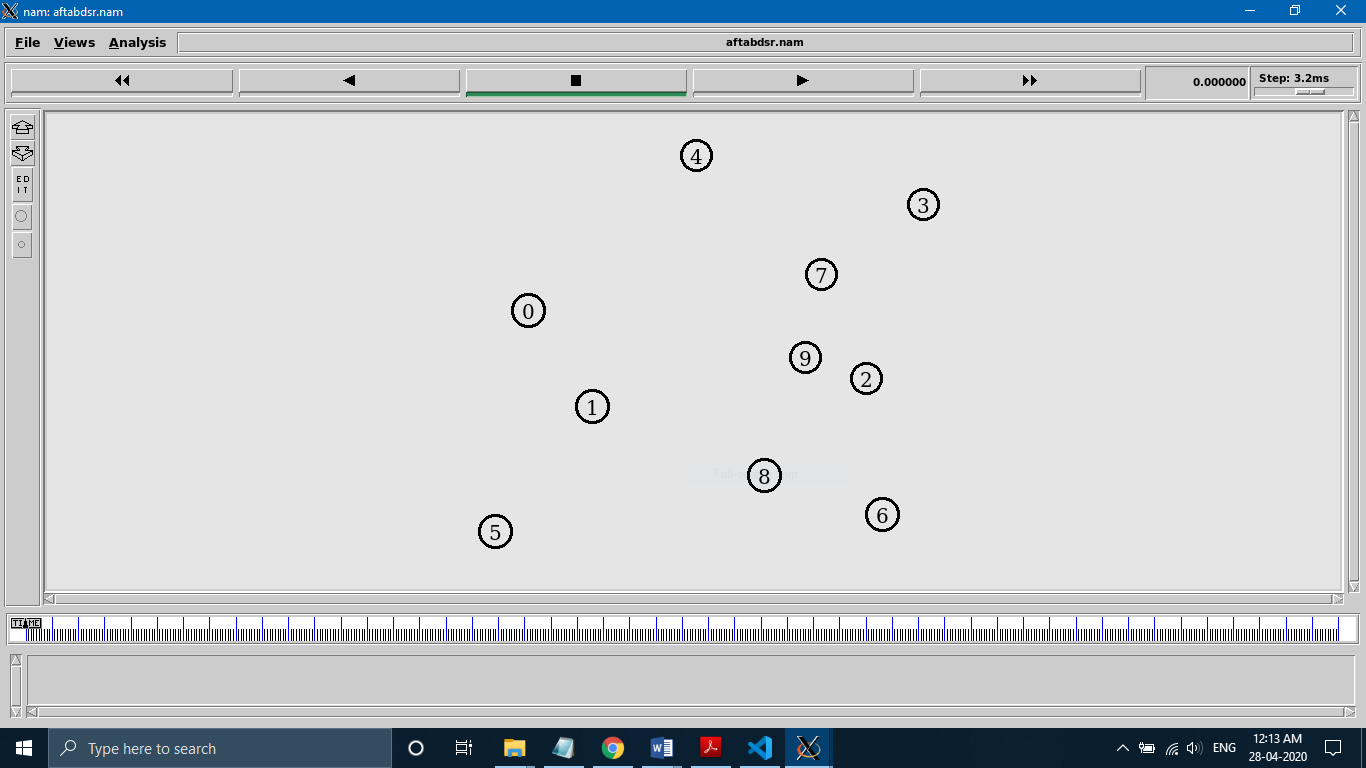
close $namtrace

exec nam aftabdsr.nam & } puts "running nam"

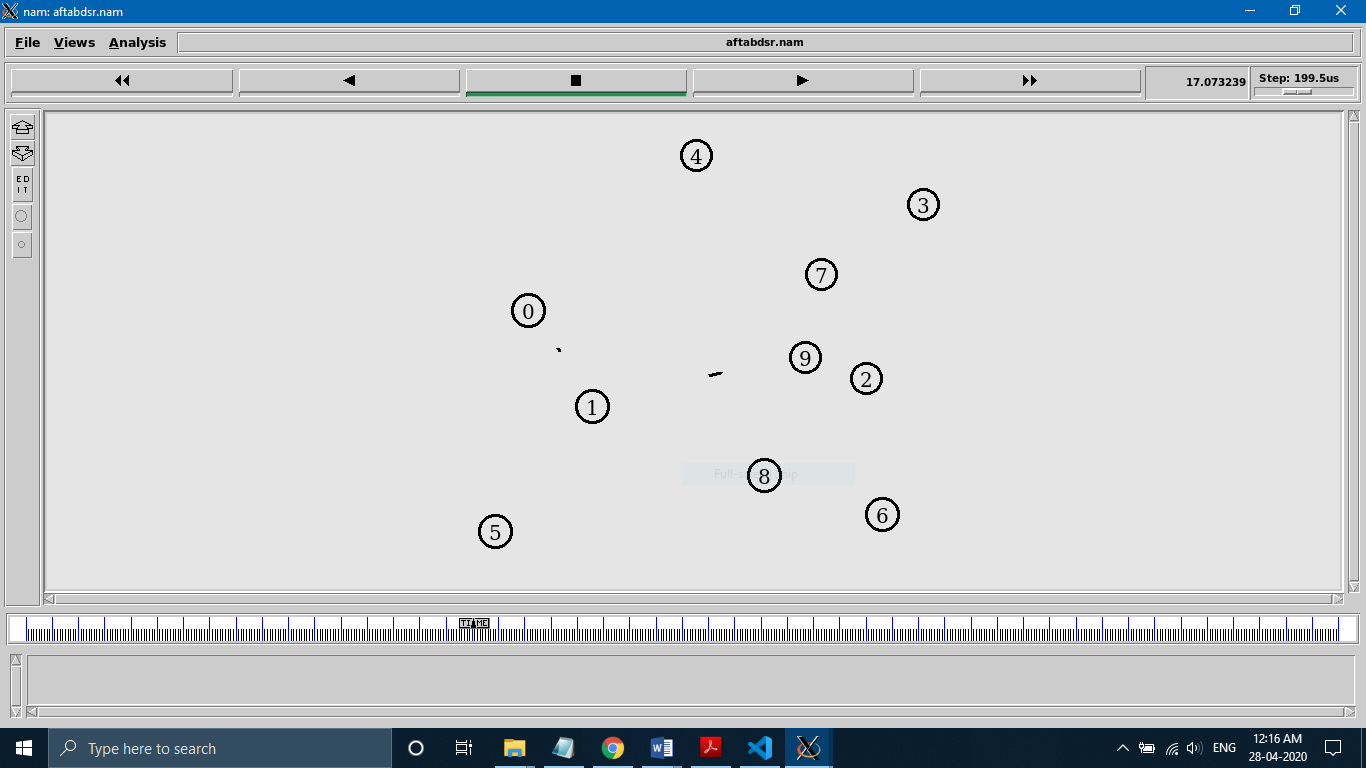
$ns run

**Screen Shots :**

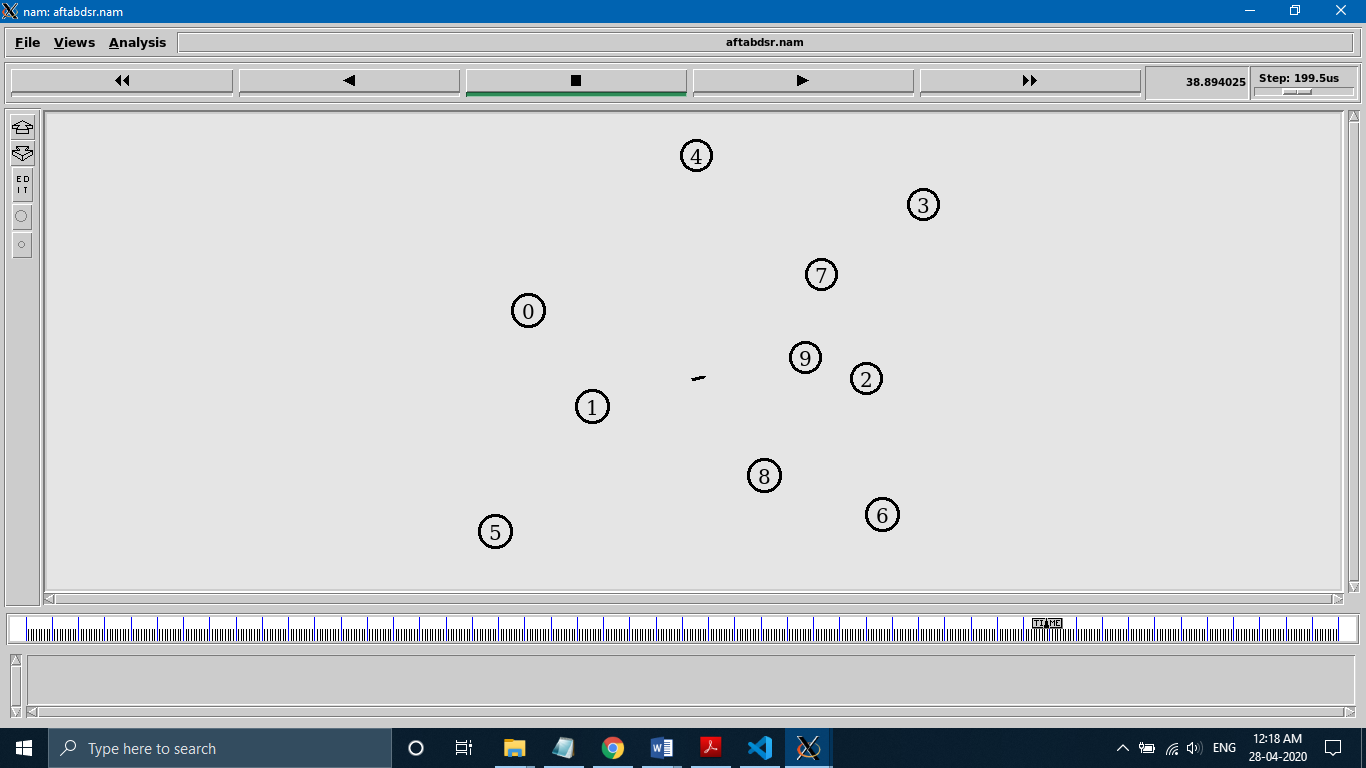




**@time = 0.00 sec**



**@time = 17.07 sec**



**@time = 38.89 sec**

**# Link State Routing Algorithm**

**Code :**

set ns [new Simulator]

set nr [open aftablsr.tr w]

$ns trace-all $nr

set nf [open aftablsr.nam w]

$ns namtrace-all $nf

proc finish { } {

global ns nr nf

$ns flush-trace

close $nf

close $nr

exec nam aftablsr.nam &

exit 0

}

for { set i 0 } { $i < 12} { incr i 1 } {

set n($i) [$ns node]}

for {set i 0} {$i < 8} {incr i} {

$ns duplex-link $n($i) $n([expr $i+1]) 1Mb 10ms DropTail }

$ns duplex-link $n(0) $n(8) 1Mb 10ms DropTail

$ns duplex-link $n(1) $n(10) 1Mb 10ms DropTail

$ns duplex-link $n(0) $n(9) 1Mb 10ms DropTail

$ns duplex-link $n(9) $n(11) 1Mb 10ms DropTail

$ns duplex-link $n(10) $n(11) 1Mb 10ms DropTail

$ns duplex-link $n(11) $n(5) 1Mb 10ms DropTail

set udp0 [new Agent/UDP]

$ns attach-agent $n(0) $udp0

set cbr0 [new Application/Traffic/CBR]

$cbr0 set packetSize\_ 500

$cbr0 set interval\_ 0.005

$cbr0 attach-agent $udp0

set null0 [new Agent/Null]

$ns attach-agent $n(5) $null0

$ns connect $udp0 $null0

set udp1 [new Agent/UDP]

$ns attach-agent $n(1) $udp1

set cbr1 [new Application/Traffic/CBR]

$cbr1 set packetSize\_ 500

$cbr1 set interval\_ 0.005

$cbr1 attach-agent $udp1

set null0 [new Agent/Null]

$ns attach-agent $n(5) $null0

$ns connect $udp1 $null0

$ns rtproto LS

$ns rtmodel-at 10.0 down $n(11) $n(5)

$ns rtmodel-at 15.0 down $n(7) $n(6)

$ns rtmodel-at 30.0 up $n(11) $n(5)

$ns rtmodel-at 20.0 up $n(7) $n(6)

$udp0 set fid\_ 1

$udp1 set fid\_ 2

$ns color 1 Red

$ns color 2 Green

$ns at 1.0 "$cbr0 start"

$ns at 2.0 "$cbr1 start"

$ns at 45 "finish"

puts "running nam"

$ns run

**Screen Shots :**

