**EXPERIMENT NO. 3**

**To create a circular topology and implement a routing protocol using NS-2**

**Aim:**

To create a larger network in NS-2 which include following features,

1. Creating large number of nodes and links between them.
2. Implementing a routing protocol.
3. Breaking and making links to demonstrate the working of the routing protocol.

**Requirements:**

Ubuntu Operating System, NS2

**Procedure:**

1. Start a new simulator.
2. Create the output files.
3. Write the finish procedure.
4. Create the topology as shown in Fig. 1 by
   1. Defining nodes and the links
   2. Creating, attaching, and connecting transport layer agents
   3. Creating and attaching application layer agent
   4. Implementing a routing protocol
   5. Scheduling the events (include breaking and making of links)
   6. Starting the simulation
5. Run the simulation
6. Observe the animated output in nam window.



Fig.1

**Implementation:**

Different approach for creating larger number of nodes and linking them.

The following code creates seven nodes and stores them in the array n( ) using ‘for’ loop.

*for {set i 0{ {$i < 7} {incr i} {*

*$ns duplex-link $n($i) $n({expr ($i+1)%7]) 1Mb 10ms DropTail*

*}*

We can also connect the nodes using ‘for’ loop and for a circular topology, %' (modulo) operator can be used.

*for {set I 0{ {$i < 7} {incr i} {*

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

The topology in nam may look strange at times. Using, ‘re-layout’ button we can make the topology look regular.

For using dynamic routing, add the following line at the beginning of Tcl script, after the simulator object has been created.

*$ns rtproto DV*

**Letting the links go down or up**

We let the link between node 1 and 2 (which is being used by the traffic) go ‘down’ for a second

*$ns rtmodel-at 1.0 down $n(1) $n(2)*

And ‘up’ again after a second

*$ns rtmodel-at 2.0 up $n(1) $n(2)*

We can see the routing getting updated and traffic being re-routed.

**Observation:**

Take appropriate snapshots and write observations below each snapshot. Students are expected to take a minimum of four snap shots for different periods of simulations.

**Post Experimental Exercise:** *(take appropriate SS to support your answers)*

1. Break link between 2 and 3 at 3sec time interval and observe the flow of data packets. Note down the observations.
2. Modify the program to send packets from node 2 to node 5. Also observe the result by making the link between node 3 and 4 down for 1 minute.

**Conclusion:** *(to be handwritten on journal sheets)*

*~code*

*#Create a simulator object*

*set ns [new Simulator]*

*#Tell the simulator to use dynamic routing*

*$ns rtproto DV*

*#Open the nam trace file*

*set nf [open out.nam w]*

*$ns namtrace-all $nf*  ***RIYA INDAP,44***

*#Define a 'finish' procedure*

*proc finish {} {*

*global ns nf*

*$ns flush-trace*

*#Close the trace file*

*close $nf*

*#Execute nam on the trace file*

*exec nam out.nam &*

*exit 0*

*}*

*#Create seven nodes*

*for {set i 0} {$i < 7} {incr i} {*

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

*}*

*#Create links between the nodes*

*for {set i 0} {$i < 7} {incr i} {*

*$ns duplex-link $n($i) $n([expr ($i+1)%7]) 1Mb 10ms DropTail*

*}*

*#Create a UDP agent and attach it to node n(0)*

*set udp0 [new Agent/UDP]*

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

*# Create a CBR traffic source and attach it to udp0*

*set cbr0 [new Application/Traffic/CBR]*

*$cbr0 set packetSize\_ 500*

*$cbr0 set interval\_ 0.005*

*$cbr0 attach-agent $udp0*

*#Create a Null agent (a traffic sink) and attach it to node n(3)*

*set null0 [new Agent/Null]*

*$ns attach-agent $n(3) $null0*

*#Connect the traffic source with the traffic sink*

*$ns connect $udp0 $null0*

*#Schedule events for the CBR agent and the network dynamics*

*$ns at 0.5 "$cbr0 start"*

*$ns rtmodel-at 1.0 down $n(1) $n(2)*

*$ns rtmodel-at 2.0 up $n(1) $n(2)*

*$ns at 4.5 "$cbr0 stop"*

*#Call the finish procedure after 5 seconds of simulation time*

*$ns at 5.0 "finish"*

*#Run the simulation*

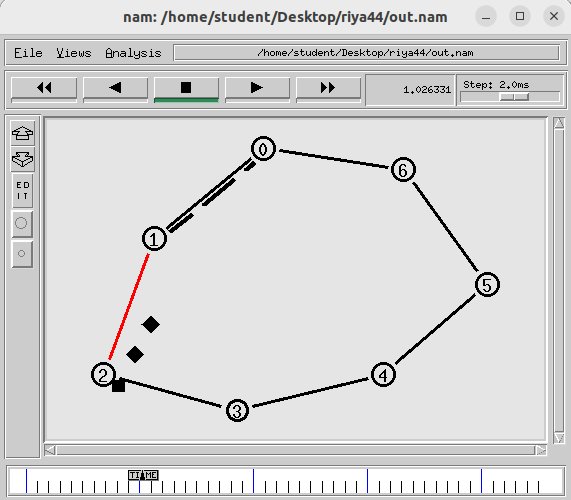
*$ns run*

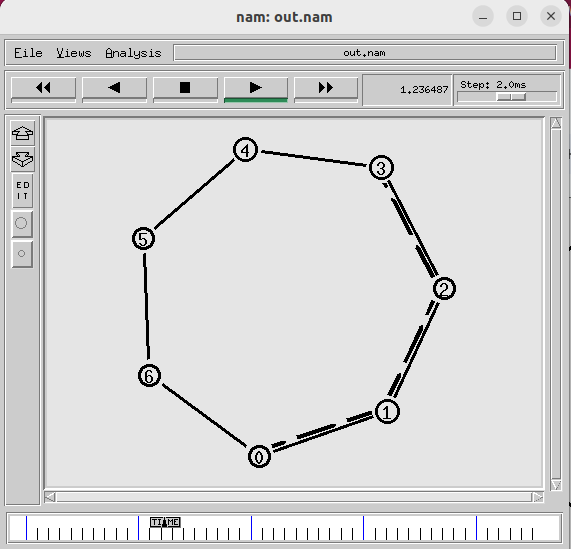
*Observation:*

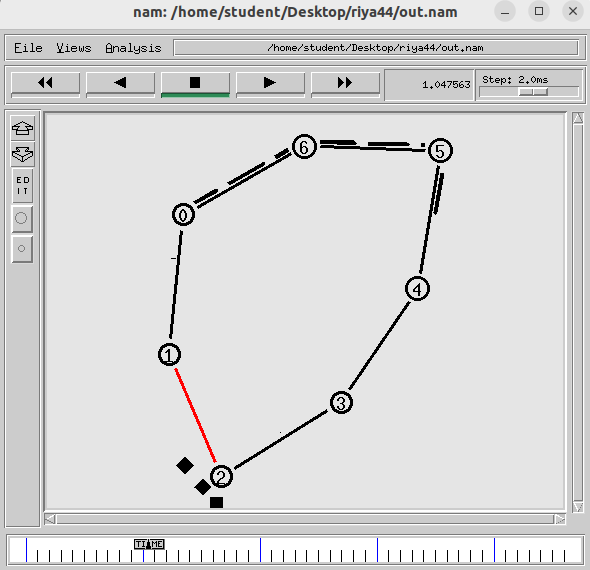
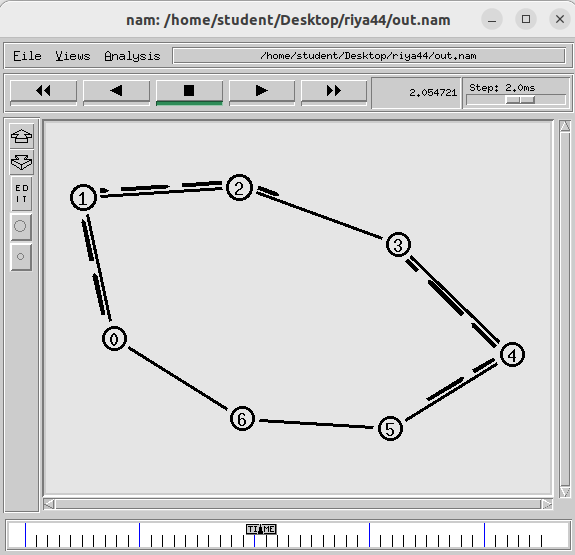
*We created a network where packets are passing As the link between node1 and node2 is broken at one*

*from node 0 to node3. second,so at that instant the packets passing through*

*were dropped and later packets are passed from n0 to n1.*

*[ $ns rtmodel-at 1.0 down $n(1) $n(2) ]*

**



*as the link between node1 and node2 is broken and As the earlier broken link is repaired,so we want the*

*packets are not passing,hence we are providing the packets to move now from the optimum path and hence*

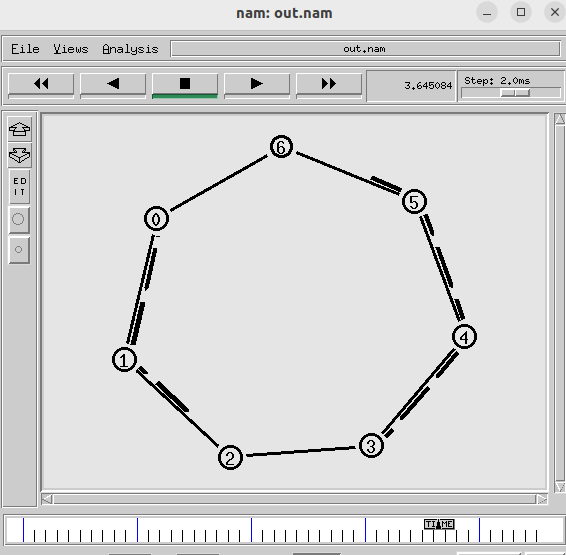
*network with a alternate route.in this whole while , we have invoked the link between node1 and node2 again*

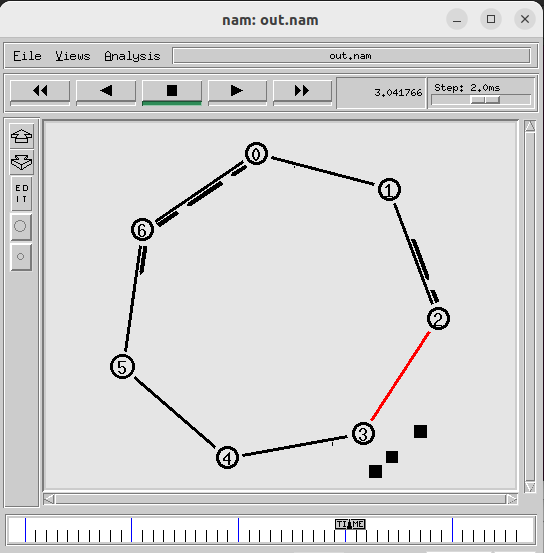
*handshake packets were travelling. at 2nd sec.*

*[ $ns rtproto DV ] [ $ns rtmodel-at 2.0 up $n(1) $n(2) ]*

***~post-experiment***

*1. Break link between 2 and 3 at 3sec time interval and observe the flow of data packets. Note*

*down the observations.*

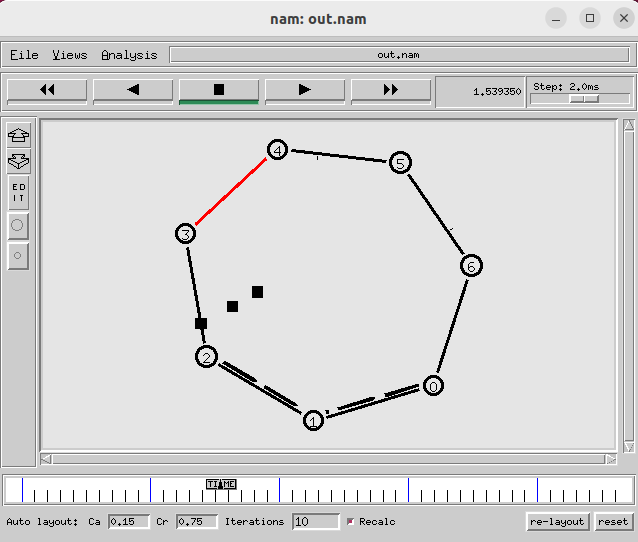
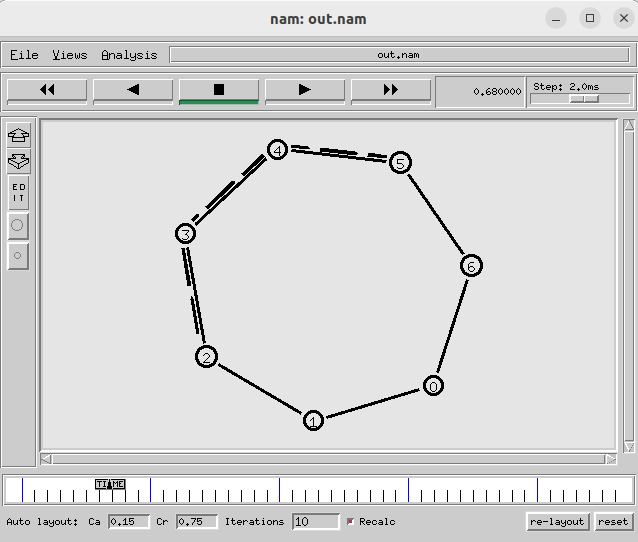


*$ns rtmodel-at 3.0 down $n(2) $n(3) $ns rtmodel-at 3.6 up $n(2) $n(3)*

*We are breaking link between 2 and 3 at 3sec We are waking up the link between 2 and 3 at 3.6 sec*

*2)Modify the program to send packets from node 2 to node 5. Also observe the result by*

*making the link between node 3 and 4 down for 1 minute.*

**

*$ns attach-agent $n(2) $udp0 $ns rtmodel-at 1.5 down $n(3) $n(4)*

*$ns attach-agent $n(5) $null0 $ns rtproto DV*

*sending packets from 2 to 5 node making the link down at 3 and 4th sec*