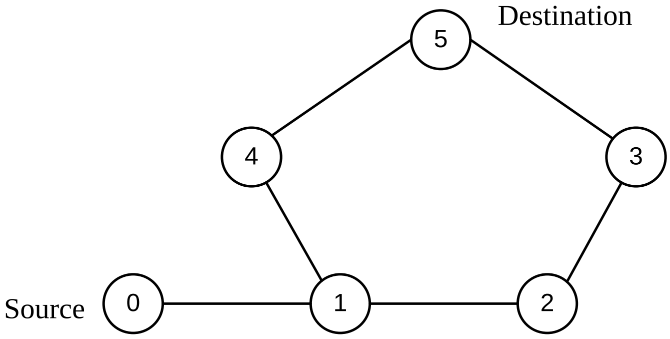
Experiment No. 7

Aim: To implement the given network topology and perform static routing.



Apparatus (software): System with Ubuntu, NS2, NAM

Procedure:

1. Create a Simulator Object.
2. Open the nam trace file.
3. Define a ‘finish’ procedure.
4. Close the Trace file.
5. Execute nam on the Trace file.
6. Create nodes.
7. Create a duplex link between the nodes as per the topology.
8. Create agents for sending and receiving data and attach them to appropriate nodes.
9. Create traffic sources and attach them to appropriate agents.
10. Program when to send data and when to stop sending data.
11. Program down time and up time for the link.
12. Call the finish procedure and run the simulation.
13. Observe the output and save the same.

Program:

```
#Create a simulator object
set ns [new Simulator]

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
        exit 0
}
```

```
#Define different colors for data flows (for NAM)
$ns color 1 Blue

#Create six nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 10ms DropTail
$ns duplex-link $n1 $n4 1Mb 10ms DropTail
$ns duplex-link $n3 $n5 1Mb 10ms DropTail
$ns duplex-link $n4 $n5 1Mb 10ms DropTail

#Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n2 $n3 orient up
$ns duplex-link-op $n1 $n4 orient up-left
$ns duplex-link-op $n3 $n5 orient left-up
$ns duplex-link-op $n4 $n5 orient right-up

#Setup a TCP connection
set tcp0 [new Agent/TCP]
#Attach TCP agent to node n0
$ns attach-agent $n0 $tcp0
#Setup a TCP sink
set sink0 [new Agent/TCPSink]
#Attch TCP sink to node n5
$ns attach-agent $n5 $sink0
#Connect TCP agent with TCP sink
$ns connect $tcp0 $sink0
$tcp0 set fid_ 1

#Setup a FTP over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ftp0 set type_ FTP
```

```
$ns at 0.5 "$ftp0 start"
$ns rtmodel-at 1.5 down $n1 $n4
$ns rtmodel-at 3.0 up $n1 $n4
$ns at 5.5 "$ftp0 stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 6.0 "finish"

#Run the simulation
$ns run
```

Conclusion:

Q. What type of services are used in this program, connection-oriented or connectionless? What protocol is used for same?
Ans.- In this program, connection-oriented services are used. The same is implemented using TCP protocol.

Q. Who sets the static path? What happens in case of link failure? Does it choose the alternate path?
Ans.- A virtual circuit identifier sets the static path. In case of link failure, the alternate path is not chosen. The transmission of packet is halted till the link becomes up.