Eexperiment 6

Aim: "Simulate transmission of ping messages over a network topology and capture the Round Trip Time".

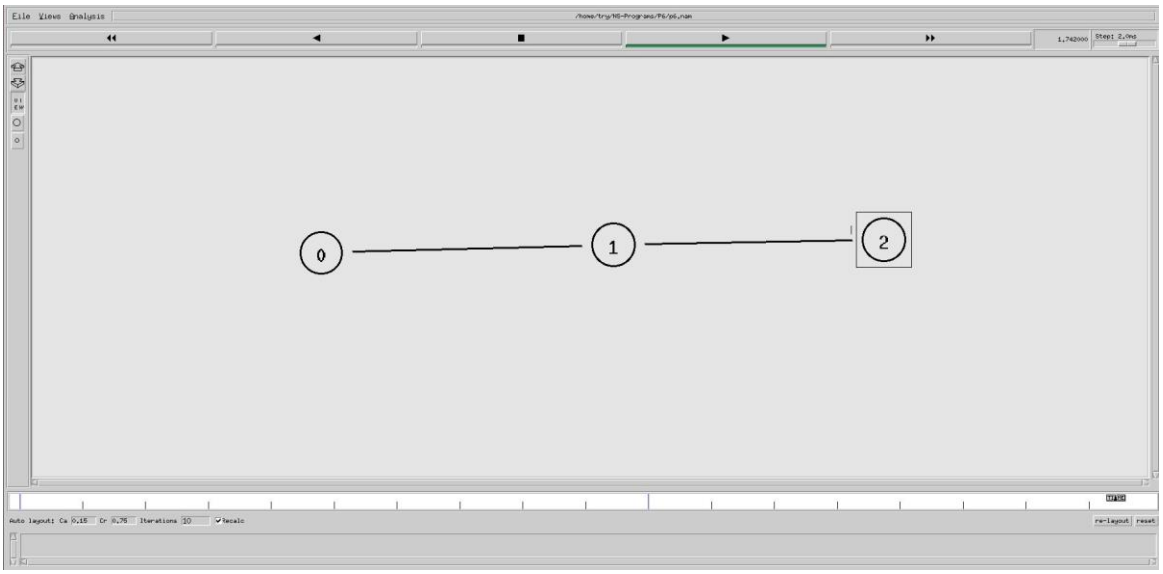
TCL Code

```
#Create Simulator
set ns [new Simulator]
#Open trace and NAM trace file
set ntrace [open p6.tr w]
$ns trace-all $ntrace
set namfile [open p6.nam w]
$ns namtrace-all $namfile
#Finish Procedure
proc Finish {} {
global ns ntrace namfile
#Dump all trace data and close the file
$ns flush-trace
close $ntrace
close $namfile
#Execute the nam animation file
exec nam p6.nam &
exit 0
}
#Create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
#Define the recv function for the class 'Agent/Ping'
#instproc adds class method called "RECEIVE" to calculate RTT
Agent/Ping instproc recv {from rtt} {
#instvar adds instance variable, and brings them to the local scope
$self instvar node_
#RTT is the length of time it takes for a signal to be sent plus the length of time it takes for an acknowledgement of that signal to be received.
puts "Node $from received ping answer from Node [$node_ id] with Round Trip Time of $rtt ms"
}
#Create two ping agents and attach them to n(0) and n(2)
set p0 [new Agent/Ping]
$ns attach-agent $n0 $p0
set p1 [new Agent/Ping]
```

```
$ns attach-agent $n2 $p1
$ns connect $p0 $p1
#Schedule events
$ns at 0.2 "$p0 send"
$ns at 0.4 "$p1 send"
$ns at 1.2 "$p0 send"
$ns at 1.7 "$p1 send"
$ns at 1.8 "Finish"
#Run the Simulation
$ns run
```

Terminal and NAM Outputs

```
try@try:~/NS-Programs/P6$ ls
p6.nam  P6-Old  p6.tcl  p6.tr
try@try:~/NS-Programs/P6$ ns p6.tcl
Node 2 received ping answer from Node 0 with Round Trip Time of 42.0 ms
Node 0 received ping answer from Node 2 with Round Trip Time of 42.0 ms
Node 2 received ping answer from Node 0 with Round Trip Time of 42.0 ms
Node 0 received ping answer from Node 2 with Round Trip Time of 42.0 ms
try@try:~/NS-Programs/P6$
```



Experiment 7

Aim: "Simulate a 6-node network to implement dynamic routing algorithm and verify its functionality".

TCL Code

```
#Create a simulator object
set ns [new Simulator]
#Tell the simulator to use dynamic routing
#Distance vector routing is an asynchronous algorithm in which node x sends the copy of its distance vector to all its neighbors. When node x receives the new distance vector from one of its #neighboring vector, v, it saves the distance vector of v and uses the Bellman-Ford equation to update its own distance vector.
$ns rtproto DV
#Open the nam trace file
set nf [open p7.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
global ns nf
$ns flush-trace
#Close the trace file
close $nf
#Execute nam on the trace file
exec nam p7.nam &
exit 0
}
#Create seven nodes
for {set i 0} {$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"
$ns at 5.0 "finish"
#Run the simulation
$ns run
```

NAM Output