Lab Assignment on Unit IV and Unit V: (Mandatory Assignment)

Aim: Use network simulator NS2 to implement:

a. Monitoring traffic for the given topology
b. Analysis of CSMA and Ethernet protocols
c. Network Routing: Shortest path routing, AODV.
d. Analysis of congestion control (TCP and UDP).

Requirements: NS2Tool.

Theory:

a. Monitoring traffic for the given topology

```
#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


#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
```

b. Analysis of CSMA and Ethernet protocols

```
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the Trace files
set file1 [open out.tr w]
set winfile [open WinFile w]
$ns trace-all $file1

#Open the NAM trace file
set file2 [open out.nam w]
```

```
$ns namtrace-all $file2

#Define a 'finish' procedure
proc finish {} {
        global ns file1 file2
        $ns flush-trace
        close $file1
        close $file2
        exec nam out.nam &
        exit 0
}

#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]

$n1 color red
$n1 shape box

#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail

set lan [$ns newLan "$n3 $n4 $n5" 0.5Mb 40ms LL Queue/DropTail MAC/Csma/Cd Channel]

# $ns duplex-link $n3 $n4 0.5Mb 40ms DropTail
# $ns duplex-link $n3 $n5 0.5Mb 30ms DropTail

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


#Set Queue Size of link (n2-n3) to 10
# $ns queue-limit $n2 $n3 20
```

```
#Setup a TCP connection
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set window_ 8000
$tcp set packetSize_ 552

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP


#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 0.01mb
$cbr set random_ false

$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 124.0 "$ftp stop"
$ns at 124.5 "$cbr stop"

# next procedure gets two arguments: the name of the
# tcp source node, will be called here "tcp",
# and the name of output file.

proc plotWindow {tcpSource file} {
global ns
set time 0.1
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
```

```
set wnd [$tcpSource set window_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 0.1 "plotWindow $tcp $winfile"


$ns at 5 "$ns trace-annotate \"packet drop\""


# PPP

$ns at 125.0 "finish"
$ns run
```

c. Network Routing: Shortest path routing, AODV.

```
# Definition of the physical layer
#-----------------------------------------------------------------

set val(chan)         Channel/WirelessChannel
set val(prop)         Propagation/TwoRayGround
set val(ant)          Antenna/OmniAntenna
set val(ll)           LL
set val(ifq)          Queue/DropTail/PriQueue
set val(ifqlen)       100
set val(netif)        Phy/WirelessPhy
set val(mac)          Mac/802_11
#-----------------------------------------------------------------
# Scenario parameters
#-----------------------------------------------------------------
set val(nn)           30
set val(rp)           AODV
set val(x)            2000
set val(y)            2000
set val(energymodel)    EnergyModel;
set val(n_ch)         chan_1


###########################################################################
#-----------------------------------------------------------------
#  Set up simulator objects
#-----------------------------------------------------------------

set ns [new Simulator]
set tracefd [open aodv.tr w]
```

```
$ns trace-all $tracefd
$ns use-newtrace
set namtrace [open sim12.nam w]
$ns namtrace-all-wireless $namtrace $val(x) $val(y)
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
#############################################################################
#creating trace files
#############################################################################
set f0 [open out02.tr w]
set f1 [open out12.tr w]
set f2 [open out22.tr w]
set f3 [open out32.tr w]

set f4 [open lost02.tr w]
set f5 [open lost12.tr w]
set f6 [open lost22.tr w]
set f7 [open lost32.tr w]

set f8 [open delay02.tr w]
set f9 [open delay12.tr w]
set f10 [open delay22.tr w]
set f11 [open delay32.tr w]
#############################################################################
#############################################################################
#defining channel parameters
#############################################################################
set chan_1 [new $val(chan)]
set chan_2 [new $val(chan)]
set chan_3 [new $val(chan)]
set chan_4 [new $val(chan)]
#############################################################################
#configuring 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) \
                 -topoInstance $topo \
                 -agentTrace ON \
                 -routerTrace ON \
```

```
		-macTrace ON \
		-movementTrace OFF \
		-channel $chan_1   \
		-energyModel $val(energymodel) \
		-rxPower 0.3 \
		-txPower 0.6 \
		-initialEnergy 90 \
			-rxPower 0.3 \
		-txPower 0.6
#################################################################################
#defining node size and motion
#################################################################################

for {set i 0} {$i < 10} { incr i } {
		set n($i) [$ns node]
		$n($i) random-motion 0
		$n($i) color red
		$ns at 0.0 "$n($i) color red"
		$ns initial_node_pos $n($i) 200
   }
for {set j 10} {$j < 20} { incr j } {
		set n($j) [$ns node]
		$n($j) random-motion 0
		$n($j) color green
		$ns at 0.0 "$n($j) color green"
		$ns initial_node_pos $n($j) 200
   }
for {set k 20} {$k < 30} { incr k } {
		set n($k) [$ns node]
		$n($k) random-motion 0
		$n($k) color blue
		$ns at 0.0 "$n($k) color blue"
		$ns initial_node_pos $n($k) 200
   }
#################################################################################
#
#creating node movement
#################################################################################
#
$ns at 0.0 "$n(4) start"
$ns at 0.0 "$n(5) start"
$ns at 0.0 "$n(6) setdest 445.0 40.0 155.0"
$ns at 0.0 "$n(7) setdest 660.0 18.0 10.0"
$ns at 0.0 "$n(8) setdest 285.0 140.0 105.0"
$ns at 0.0 "$n(9) setdest 860.0 158.0 150.0"
$ns at 0.0 "$n(10) setdest 745.0 940.0 105.0"
$ns at 0.0 "$n(11) setdest 660.0 398.0 180.0"
```

```
$ns at 0.0 "$n(12) setdest 445.0 220.0 105.0"
$ns at 0.0 "$n(13) setdest 60.0 218.0 110.0"
$ns at 0.0 "$n(14) setdest 345.0 40.0 105.0"
$ns at 0.0 "$n(15) setdest 960.0 18.0 10.0"
$ns at 0.0 "$n(16) setdest 445.0 740.0 155.0"
$ns at 0.0 "$n(17) setdest 660.0 918.0 10.0"
$ns at 0.0 "$n(18) setdest 285.0 540.0 105.0"
$ns at 0.0 "$n(19) setdest 860.0 658.0 150.0"
$ns at 0.0 "$n(20) setdest 845.0 940.0 105.0"
$ns at 0.0 "$n(11) setdest 760.0 398.0 180.0"
$ns at 0.0 "$n(22) setdest 945.0 220.0 105.0"
$ns at 0.0 "$n(23) setdest 60.0 918.0 110.0"
$ns at 0.0 "$n(24) setdest 545.0 640.0 105.0"
$ns at 0.0 "$n(25) setdest 460.0 818.0 10.0"
$ns at 0.0 "$n(26) setdest 545.0 340.0 155.0"
$ns at 0.0 "$n(27) setdest 860.0 318.0 10.0"
$ns at 0.0 "$n(28) setdest 685.0 340.0 105.0"
$ns at 0.0 "$n(29) setdest 860.0 658.0 150.0"
$ns at 7.0 "$n(9) setdest 545.0 40.0 105.0"
$ns at 7.0 "$n(8) setdest 760.0 18.0 10.0"
$ns at 7.0 "$n(7) setdest 445.0 40.0 155.0"
$ns at 7.0 "$n(6) setdest 660.0 18.0 10.0"
$ns at 7.0 "$n(5) setdest 285.0 140.0 105.0"
$ns at 7.0 "$n(4) setdest 860.0 158.0 150.0"
$ns at 7.0 "$n(19) setdest 745.0 940.0 105.0"
$ns at 7.0 "$n(18) setdest 660.0 398.0 180.0"
$ns at 7.0 "$n(17) setdest 445.0 220.0 105.0"
$ns at 7.0 "$n(16) setdest 60.0 218.0 110.0"
$ns at 7.0 "$n(15) setdest 345.0 40.0 105.0"
$ns at 7.0 "$n(14) setdest 960.0 18.0 10.0"
$ns at 7.0 "$n(13) setdest 445.0 740.0 155.0"
$ns at 7.0 "$n(12) setdest 660.0 918.0 10.0"
$ns at 7.0 "$n(11) setdest 285.0 540.0 105.0"
$ns at 7.0 "$n(10) setdest 860.0 658.0 150.0"
$ns at 7.0 "$n(29) setdest 845.0 940.0 105.0"
$ns at 7.0 "$n(18) setdest 760.0 398.0 180.0"
$ns at 7.0 "$n(27) setdest 945.0 220.0 105.0"
$ns at 7.0 "$n(26) setdest 60.0 918.0 110.0"
$ns at 7.0 "$n(25) setdest 545.0 640.0 105.0"
$ns at 7.0 "$n(24) setdest 460.0 818.0 10.0"
$ns at 7.0 "$n(23) setdest 545.0 340.0 155.0"
$ns at 7.0 "$n(22) setdest 860.0 318.0 10.0"
$ns at 7.0 "$n(21) setdest 685.0 340.0 105.0"
$ns at 7.0 "$n(20) setdest 860.0 658.0 150.0"
#############################################################################
```

```
#############################################################################
$ns at 6.0 "$n(1) setdest 625.0 625.0 185.0"
$ns at 5.0 "$n(0) setdest 450.0 450.0 180.0"
$ns at 12.0 "$n(1) setdest 180.0 185.0 185.0"
$ns at 6.0 "$n(2) setdest 430.0 430.0 185.0"
$ns at 12.0 "$n(2) setdest 390.0 390.0 185.0"
$ns at 6.0 "$n(3) setdest 450.0 450.0 185.0"
$ns at 12.0 "$n(3) setdest 230.0 265.0 185.0"
#############################################################################
#connecting traffic agent
#n(0) source n(1) sink0
#############################################################################
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set sink0 [new Agent/LossMonitor]
$ns attach-agent $n(1) $sink0
$ns connect $udp0 $sink0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set rate_ 600kb
$cbr0 set interval_ 0.05
$cbr0 attach-agent $udp0
#############################################################################
#n(2) source n(4) sink1
#############################################################################
set udp1 [new Agent/UDP]
$ns attach-agent $n(2) $udp1
set sink1 [new Agent/LossMonitor]
$ns attach-agent $n(4) $sink1
$ns connect $udp1 $sink1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 500
$cbr1 set rate_ 600kb
$cbr1 set interval_ 0.05
$cbr1 attach-agent $udp1
#############################################################################
#n(5) source n(6) sink
#############################################################################
set udp2 [new Agent/UDP]
$ns attach-agent $n(5) $udp2
set sink2 [new Agent/LossMonitor]
$ns attach-agent $n(6) $sink2
$ns connect $udp2 $sink2
set cbr2 [new Application/Traffic/CBR]
$cbr2 set packetSize_ 500
$cbr2 set rate_ 600kb
```

```
$cbr2 set interval_ 0.05
$cbr2 attach-agent $udp2
################################################################################
#
#n(3) source n(6) sink
################################################################################
#
set udp3 [new Agent/UDP]
$ns attach-agent $n(3) $udp3
set sink3 [new Agent/LossMonitor]
$ns attach-agent $n(6) $sink3
$ns connect $udp3 $sink3
set cbr3 [new Application/Traffic/CBR]
$cbr3 set packetSize_ 500
$cbr3 set rate_ 600kb
$cbr3 set interval_ 0.05
$cbr3 attach-agent $udp3
################################################################################
#finish procedure
################################################################################
proc finish {} {
	global ns f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 f10 f11 tracefd namtrace
	   #Close the output files
	close $f0
	   close $f1
	   close $f2
	   close $f3
	   close $f4
	close $f5
	close $f6
	close $f7
	close $f8
	close $f9
	close $f10
	close $f11
	   $ns flush-trace
	close $tracefd
	   close $namtrace
	   exec nam sim12.nam &
	#Call xgraph to display the results
	exec xgraph out02.tr out12.tr out22.tr out32.tr -geometry 800x400 -y bandwidth -t
30_node_AODV_throughput -x time &
	exec xgraph lost02.tr lost12.tr lost22.tr lost32.tr -geometry 800x400 -y packet-loss -t
30_node_AODV_packet_loss -x time &
	exec xgraph delay02.tr delay12.tr delay22.tr delay32.tr -geometry 800x400 -y routing-delay
-t 30_node_AODV_packet_delay -x time &
```

```
	exit 0
}
#######################################################################################
#node position setting
#######################################################################################

#node position -static (fixed ) topology
$n(0) set X_ 850.0
$n(0) set Y_ 50.0
$n(0) set Z_ 0.0

$n(1) set X_ 1200.0
$n(1) set Y_ 200.0
$n(1) set Z_ 0.0

$n(2) set X_ 1300.0
$n(2) set Y_ 300.0
$n(2) set Z_ 0.0

$n(3) set X_ 1150.0
$n(3) set Y_ 300.0
$n(3) set Z_ 0.0

$n(4) set X_ 1400.0
$n(4) set Y_ 400.0
$n(4) set Z_ 0.0

$n(5) set X_ 1800.0
$n(5) set Y_ 300.0
$n(5) set Z_ 0.0

$n(6) set X_ 200.0
$n(6) set Y_ 150.0
$n(6) set Z_ 0.0

$n(7) set X_ 60.0
$n(7) set Y_ 130.0
$n(7) set Z_ 0.0

$n(8) set X_ 210.0
$n(8) set Y_ 850.0
$n(8) set Z_ 0.0

$n(9) set X_ 700.0
$n(9) set Y_ 920.0
$n(9) set Z_ 0.0
```

```
$n(10) set X_ 1250.0
$n(10) set Y_ 800.0
$n(10) set Z_ 0.0

$n(11) set X_ 1300.0
$n(11) set Y_ 1400.0
$n(11) set Z_ 0.0

$n(12) set X_ 1600.0
$n(12) set Y_ 1350.0
$n(12) set Z_ 0.0

$n(13) set X_ 200.0
$n(13) set Y_ 250.0
$n(13) set Z_ 0.0

$n(14) set X_ 250.0
$n(14) set Y_ 50.0
$n(14) set Z_ 0.0

$n(15) set X_ 120.0
$n(15) set Y_ 1200.0
$n(15) set Z_ 0.0

$n(16) set X_ 300.0
$n(16) set Y_ 1350.0
$n(16) set Z_ 0.0

$n(17) set X_ 1150.0
$n(17) set Y_ 350.0
$n(17) set Z_ 0.0

$n(18) set X_ 400.0
$n(18) set Y_ 300.0
$n(18) set Z_ 0.0

$n(19) set X_ 100.0
$n(19) set Y_ 320.0
$n(19) set Z_ 0.0

$n(20) set X_ 200.0
$n(20) set Y_ 150.0
$n(20) set Z_ 0.0

$n(21) set X_ 1150.0
```

```
$n(21) set Y_ 90.0
$n(21) set Z_ 0.0

$n(22) set X_ 200.0
$n(22) set Y_ 1120.0
$n(22) set Z_ 0.0

$n(23) set X_ 300.0
$n(23) set Y_ 1130.0
$n(23) set Z_ 0.0

$n(24) set X_ 1315.0
$n(24) set Y_ 300.0
$n(24) set Z_ 0.0

$n(25) set X_ 400.0
$n(25) set Y_ 1400.0
$n(25) set Z_ 0.0

$n(26) set X_ 10.0
$n(26) set Y_ 30.0
$n(26) set Z_ 0.0

$n(27) set X_ 20.0
$n(27) set Y_ 1750.0
$n(27) set Z_ 0.0

$n(28) set X_ 1970.0
$n(28) set Y_ 230.0
$n(28) set Z_ 0.0

$n(29) set X_ 1840.0
$n(29) set Y_ 420.0
$n(29) set Z_ 0.0
#############################################################################
# Initialize Flags
#############################################################################
set holdtime0 0
set holdseq0 0
set holdtime1 0
set holdseq1 0
set holdtime2 0
set holdseq2 0
set holdtime3 0
set holdseq3 0
set holdrate0 0
```

```
set holdrate1 0
set holdrate2 0
set holdrate3 0
#################################################################################
#record procedure
#################################################################################

proc record {} {
	global sink0 sink1 sink2 sink3 f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 f10 f11 holdtime0 holdseq0
holdtime1 holdseq1 holdtime2 holdseq2 holdtime3 holdseq3 holdrate0 holdrate1 holdrate2
holdrate3
	set ns [Simulator instance]
	#Set the time after which the procedure should be called again
	set time 0.5
	   set now [$ns now]
	   set bw0 [$sink0 set bytes_]
	   set bw1 [$sink1 set bytes_]
	set bw2 [$sink2 set bytes_]
	set bw3 [$sink3 set bytes_]

	   set bw4 [$sink0 set nlost_]
	set bw5 [$sink1 set nlost_]
	set bw6 [$sink2 set nlost_]
	set bw7 [$sink3 set nlost_]

	   set bw8 [$sink0 set lastPktTime_]
	set bw9 [$sink0 set npkts_]

	set bw10 [$sink1 set lastPktTime_]
	set bw11 [$sink1 set npkts_]

	set bw12 [$sink2 set lastPktTime_]
	set bw13 [$sink2 set npkts_]

	set bw14 [$sink3 set lastPktTime_]
	set bw15 [$sink3 set npkts_]

	puts $f0 "$now [expr (($bw0+$holdrate0)*8)/(2*$time*1000000)]"
	puts $f1 "$now [expr (($bw1+$holdrate1)*8)/(2*$time*1000000)]"
	puts $f2 "$now [expr (($bw2+$holdrate2)*8)/(2*$time*1000000)]"
	puts $f3 "$now [expr (($bw3+$holdrate3)*8)/(2*$time*1000000)]"

	# Record Packet Loss Rate in File
	puts $f4 "$now [expr $bw4/$time]"
	puts $f5 "$now [expr $bw5/$time]"
	puts $f6 "$now [expr $bw6/$time]"
	puts $f7 "$now [expr $bw7/$time]"
```

```
# Record Packet Delay in File

if { $bw9 > $holdseq0 } {
	puts $f8 "$now [expr ($bw8 - $holdtime0)/($bw9 - $holdseq0)]"
} else {
	puts $f8 "$now [expr ($bw9 - $holdseq0)]"
}



if { $bw11 > $holdseq1 } {

	puts $f9 "$now [expr ($bw10 - $holdtime1)/($bw11 - $holdseq1)]"

} else {

	puts $f9 "$now [expr ($bw11 - $holdseq1)]"

}



if { $bw13 > $holdseq2 } {

	puts $f10 "$now [expr ($bw12 - $holdtime2)/($bw13 - $holdseq2)]"

} else {

	puts $f10 "$now [expr ($bw13 - $holdseq2)]"

}



if { $bw15 > $holdseq3 } {

	puts $f11 "$now [expr ($bw14 - $holdtime3)/($bw15 - $holdseq3)]"

} else {

	puts $f11 "$now [expr ($bw15 - $holdseq3)]"

}
```

```
        $sink0 set bytes_ 0
    $sink1 set bytes_ 0
    $sink2 set bytes_ 0
    $sink3 set bytes_ 0

        $sink0 set nlost_ 0
    $sink1 set nlost_ 0
    $sink2 set nlost_ 0
    $sink3 set nlost_ 0

    set holdtime0 $bw8
    set holdseq0 $bw9

        set holdrate0 $bw0
    set holdrate1 $bw1
    set holdrate2 $bw2
    set holdrate3 $bw3


        #Re-schedule the procedure
    $ns at [expr $now+$time] "record"
}

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

   $ns at 60.0 "$n($i) reset";

}


$ns at 0.0 "record"
$ns at 10.0 "$cbr0 start"
$ns at 10.0 "$cbr1 start"
$ns at 10.0 "$cbr2 start"
$ns at 10.0 "$cbr3 start"
$ns at 50.0 "$cbr0 stop"
$ns at 50.0 "$cbr1 stop"
$ns at 50.0 "$cbr2 stop"
$ns at 50.0 "$cbr3 stop"
$ns at 60.0 "finish"

puts "Starting Simulation..."

#Run the simulation
$ns run
```

Sample code for AODV routing protocol in NS2 :

This is code for AODV link failure condition checking.

```
void
AODV::handle_link_failure(nsaddr_t id) {
aodv_rt_entry *rt, *rtn;
Packet *rerr = Packet::alloc();
struct hdr_aodv_error *re = HDR_AODV_ERROR(rerr);
re->DestCount = 0;
for(rt = rtable.head(); rt; rt = rtn) {
// for each rt entry
rtn = rt->rt_link.le_next;
if ((rt->rt_hops != INFINITY2) && (rt->rt_nexthop == id) ) {
assert (rt->rt_flags == RTF_UP);
assert((rt->rt_seqno%2) == 0);
rt->rt_seqno++;
re->unreachable_dst[re->DestCount] = rt->rt_dst;
re->unreachable_dst_seqno[re->DestCount] = rt->rt_seqno;
#ifdef DEBUG
fprintf(stderr, "%s(%f): %d\t(%d\t%u\t%d)\n", __FUNCTION__, CURRENT_TIME,
index, re->unreachable_dst[re->DestCount],
re->unreachable_dst_seqno[re->DestCount], rt->rt_nexthop);
#endif // DEBUG
re->DestCount += 1;
rt_down(rt);
}
```

d. Analysis of congestion control (TCP and UDP).

```
set ns [new Simulator]

set nf [open out.nam w]
$ns namtrace-all $nf

set tf [open out.tr w]
set windowVsTime [open win w]
```

```
set param [open parameters w]
$ns trace-all $tf

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

#Create bottleneck and dest nodes
set n2 [$ns node]
set n3 [$ns node]

#Create links between these nodes
$ns duplex-link $n2 $n3 0.7Mb 20ms DropTail

set NumbSrc 5
set Duration 10

#Source nodes
for {set j 1} {$j<=$NumbSrc} { incr j } {
set S($j) [$ns node]
}

# Create a random generator for starting the ftp and for bottleneck link delays
set rng [new RNG]
$rng seed 0

# paratmers for random variables for delays
set RVdly [new RandomVariable/Uniform]
$RVdly set min_ 1
$RVdly set max_ 5
$RVdly use-rng $rng

# parameters for random variables for begenning of ftp connections
set RVstart [new RandomVariable/Uniform]
$RVstart set min_ 0
$RVstart set max_ 7
$RVstart use-rng $rng

#We define two random parameters for each connections
for {set i 1} {$i<=$NumbSrc} { incr i } {
```

```
set startT($i)  [expr [$RVstart value]]
set dly($i) [expr [$RVdly value]]
puts $param "dly($i) $dly($i) ms"
puts $param "startT($i)  $startT($i) sec"
}

#Links between source and bottleneck
for {set j 1} {$j<=$NumbSrc} { incr j } {
$ns duplex-link $S($j) $n2 10Mb $dly($j)ms DropTail
$ns queue-limit $S($j) $n2 100
}

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

#TCP Sources
for {set j 1} {$j<=$NumbSrc} { incr j } {
set tcp_src($j) [new Agent/TCP/Reno]
}

#TCP Destinations
for {set j 1} {$j<=$NumbSrc} { incr j } {
set tcp_snk($j) [new Agent/TCPSink]
}

#Connections
for {set j 1} {$j<=$NumbSrc} { incr j } {
$ns attach-agent $S($j) $tcp_src($j)
$ns attach-agent $n3 $tcp_snk($j)
$ns connect $tcp_src($j) $tcp_snk($j)
}

#FTP sources
for {set j 1} {$j<=$NumbSrc} { incr j } {
set ftp($j) [$tcp_src($j) attach-source FTP]
}

#Parametrisation of TCP sources
for {set j 1} {$j<=$NumbSrc} { incr j } {
$tcp_src($j) set packetSize_ 552
}

#Schedule events for the FTP agents:
```

```
for {set i 1} {$i<=$NumbSrc} { incr i } {
$ns at $startT($i) "$ftp($i) start"
$ns at $Duration "$ftp($i) stop"
}

proc plotWindow {tcpSource file k} {
global ns
set time 0.03
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$k $now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file $k" }

# The procedure will now be called for all tcp sources
for {set j 1} {$j<=$NumbSrc} { incr j } {
$ns at 0.1 "plotWindow $tcp_src($j) $windowVsTime $j"
}

$ns at [expr $Duration] "finish"
$ns run
```