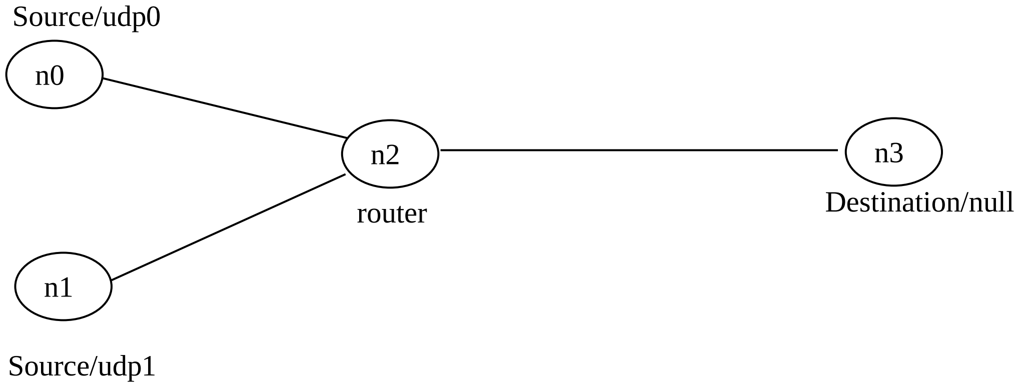
Lab Experiment 1 :

Implement three nodes point – to – point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.

Topology-



Code –

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

#Open trace file
set nt [open lab1.tr w]
$ns trace-all $nt

#Open namtrace file
set nf [open lab1.nam w]
$ns namtrace-all $nf

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

#Assign color to the packet
$ns color 1 Blue
$ns color 2 Red

#label nodes
$n0 label "Source/udp0"
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/null"

#create links, specify the type, nodes, bandwidth, delay and ARQ algorithm for it
$ns duplex-link $n0 $n2 10Mb 300ms DropTail
$ns duplex-link $n1 $n2 10Mb 300ms DropTail
```

```
$ns duplex-link $n2 $n3 100Kb 300ms DropTail

#set queue size between the nodes
$ns queue-limit $n0 $n2 10
$ns queue-limit $n1 $n2 10
$ns queue-limit $n2 $n3 5

#create and attach UDP agent to n0, n1 and Null agent to n3
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1


set null3 [new Agent/Null]
$ns attach-agent $n3 $null3

#attach Application cbr to udp
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0

set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1

#set udp0 packet to red color and udp1 packet to blue color
$udp0 set class_ 1
$udp1 set class_ 2

#connect the agents
$ns connect $udp0 $null3
$ns connect $udp1 $null3

#set packet size and interval for cbr1
$cbr1 set packetSize_ 500Mb
$cbr1 set interval_ 0.005

#finish procedure
proc finish { } {
        global ns nf nt
        $ns flush-trace
        exec nam lab1.nam &
        close $nt
        close $nf
        exit 0
}

$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
$ns run
```

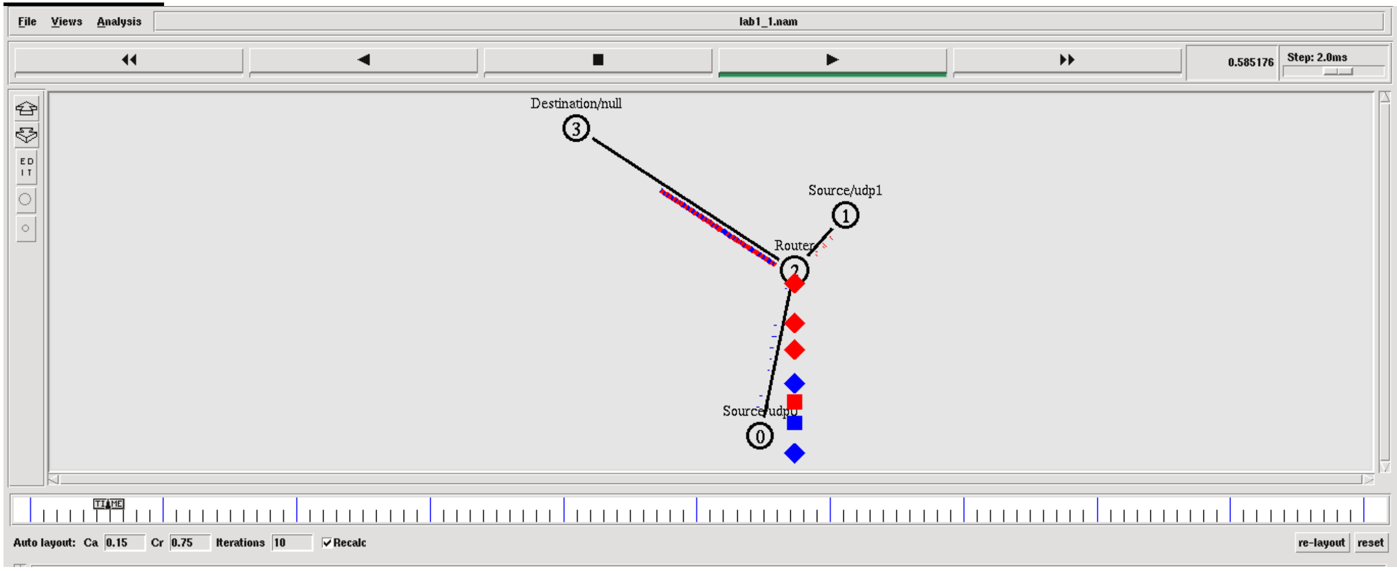
Awk file-

```
BEGIN{
        count=0;
}
{
        if($1=="r")
        count++
}
END{
        printf("Number of packets dropped is = %d\n",count);
}
```

Output-

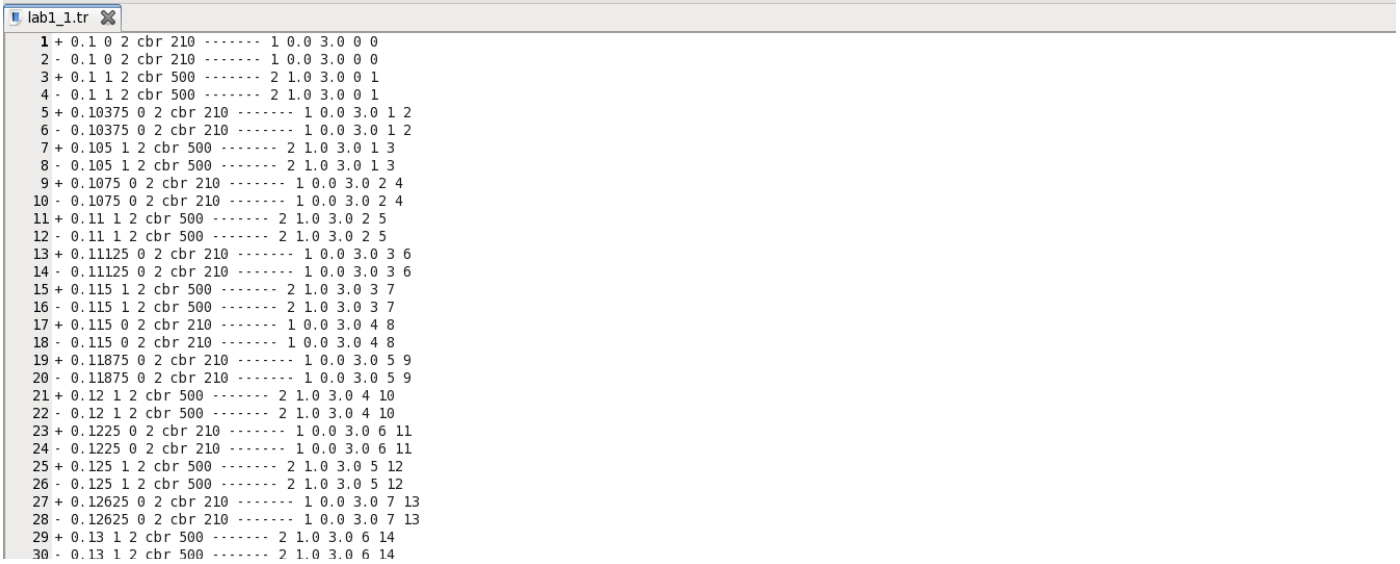
```
$awk  -f  numDrop.awk  lab1.tr
Number of packets dropped is = 714
```

Simulation-



Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑Recalc

Trace File-

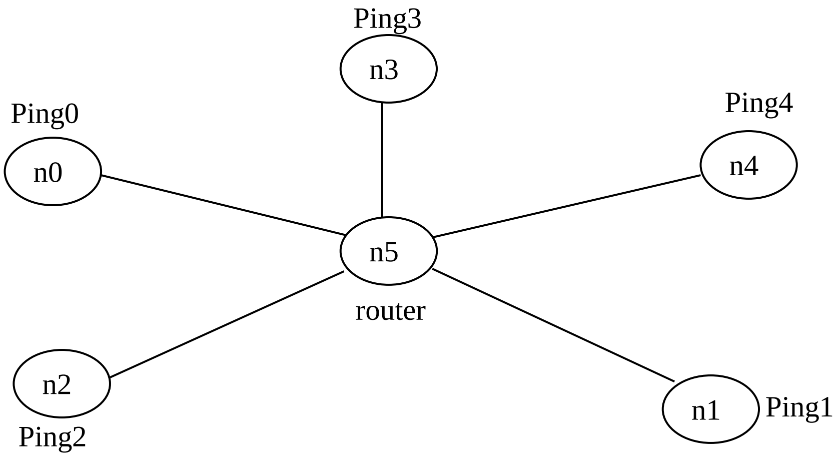


```
+ 0.1 0 2 cbr 210 ------- 1 0.0 3.0 0 0
- 0.1 0 2 cbr 210 ------- 1 0.0 3.0 0 0
+ 0.1 1 2 cbr 500 ------- 2 1.0 3.0 0 1
- 0.1 1 2 cbr 500 ------- 2 1.0 3.0 0 1
+ 0.10375 0 2 cbr 210 ------- 1 0.0 3.0 1 2
- 0.10375 0 2 cbr 210 ------- 1 0.0 3.0 1 2
+ 0.105 1 2 cbr 500 ------- 2 1.0 3.0 1 3
- 0.105 1 2 cbr 500 ------- 2 1.0 3.0 1 3
+ 0.1075 0 2 cbr 210 ------- 1 0.0 3.0 2 4
- 0.1075 0 2 cbr 210 ------- 1 0.0 3.0 2 4
+ 0.11 1 2 cbr 500 ------- 2 1.0 3.0 2 5
- 0.11 1 2 cbr 500 ------- 2 1.0 3.0 2 5
+ 0.11125 0 2 cbr 210 ------- 1 0.0 3.0 3 6
- 0.11125 0 2 cbr 210 ------- 1 0.0 3.0 3 6
+ 0.115 1 2 cbr 500 ------- 2 1.0 3.0 3 7
- 0.115 1 2 cbr 500 ------- 2 1.0 3.0 3 7
+ 0.115 0 2 cbr 210 ------- 1 0.0 3.0 4 8
- 0.115 0 2 cbr 210 ------- 1 0.0 3.0 4 8
+ 0.11875 0 2 cbr 210 ------- 1 0.0 3.0 5 9
- 0.11875 0 2 cbr 210 ------- 1 0.0 3.0 5 9
+ 0.12 1 2 cbr 500 ------- 2 1.0 3.0 4 10
- 0.12 1 2 cbr 500 ------- 2 1.0 3.0 4 10
+ 0.1225 0 2 cbr 210 ------- 1 0.0 3.0 6 11
- 0.1225 0 2 cbr 210 ------- 1 0.0 3.0 6 11
+ 0.125 1 2 cbr 500 ------- 2 1.0 3.0 5 12
- 0.125 1 2 cbr 500 ------- 2 1.0 3.0 5 12
+ 0.12625 0 2 cbr 210 ------- 1 0.0 3.0 7 13
- 0.12625 0 2 cbr 210 ------- 1 0.0 3.0 7 13
+ 0.13 1 2 cbr 500 ------- 2 1.0 3.0 6 14
- 0.13 1 2 cbr 500 ------- 2 1.0 3.0 6 14
```

Lab Experiment 2 :

Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

Topology-



Code-

```
#create Simulator object
set ns [new Simulator]

#open trace file
set nt [open prac2.tr w]
$ns trace-all $nt

#open namtrace file
set nf [open prac2.nam w]
$ns namtrace-all $nf


#create 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]

#label nodes
$n0 label "ping0"
$n1 label "ping1"
$n2 label "ping2"
$n3 label "ping3"
$n4 label "ping4"
$n5 label "router"
```

```
#create links, specify the type, nodes, bandwidth, delay and ARQ algorithm for it
$ns duplex-link $n0 $n5 1Mb 10ms DropTail
$ns duplex-link $n1 $n5 1Mb 10ms DropTail
$ns duplex-link $n2 $n5 1Mb 10ms DropTail
$ns duplex-link $n3 $n5 1Mb 10ms DropTail
$ns duplex-link $n4 $n5 1Mb 10ms DropTail

#set queue length
$ns queue-limit $n0 $n5 5
$ns queue-limit $n1 $n5 5
$ns queue-limit $n2 $n5 2
$ns queue-limit $n3 $n5 5
$ns queue-limit $n4 $n5 2

$ns color 2 Red
$ns color 3 Blue
$ns color 4 Green
$ns color 5 Yellow
```

#define ‘recv’ function for class Agent/Ping

```
Agent/Ping instproc recv {from rtt} {
        $self instvar node_
        puts "node [$node_ id] received ping answer from $from with round-trip time $rtt ms"
}

#create ping agent and attach them to node
set p0 [new Agent/Ping]
$ns attach-agent $n0 $p0
$p0 set class_ 1

set p1 [new Agent/Ping]
$ns attach-agent $n1 $p1
$p1 set class_ 2

set p2 [new Agent/Ping]
$ns attach-agent $n2 $p2
$p2 set class_ 3

set p3 [new Agent/Ping]
$ns attach-agent $n3 $p3
$p3 set class_ 4

set p4 [new Agent/Ping]
$ns attach-agent $n4 $p4
$p4 set class_ 5
#connect 2 agents
$ns connect $p2 $p4
$ns connect $p3 $p4

proc sendPingPacket { } {
        global ns p2 p3
        set intervalTime 0.001
        set now [$ns now]
        $ns at [expr $now + $intervalTime] "$p2 send"
        $ns at [expr $now + $intervalTime] "$p3 send"
        $ns at [expr $now + $intervalTime] "sendPingPacket"
}

proc finish { } {
        global ns nt nf
        $ns flush-trace

        close $nt
        close $nf
        exec nam prac2.nam &
        exit 0
}

$ns at 0.1 "sendPingPacket"
$ns at 2.0 "finish"
$ns run
```

Awk file-

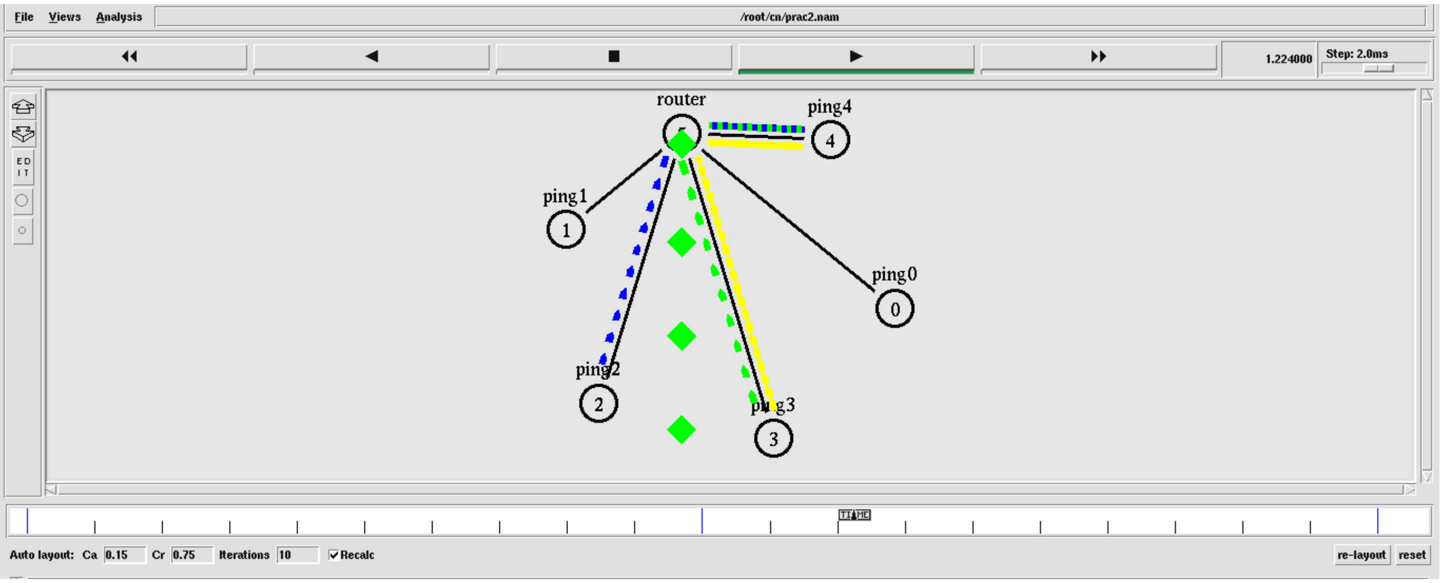
```
BEGIN{
        count=0;
}
{
        if($1=="r")
        count++
}
END{
        printf("Number of packets dropped is = %d\n",count);
}
```

Output-

```
$ns lab2.tcl
node 3 received ping answer from 4 with round-trip time 66.3 ms
node 3 received ping answer from 4 with round-trip time 66.8 ms
node 3 received ping answer from 4 with round-trip time 66.3 ms
node 3 received ping answer from 4 with round-trip time 66.9 ms
node 3 received ping answer from 4 with round-trip time 66.4 ms
node 3 received ping answer from 4 with round-trip time 66.9 ms
node 3 received ping answer from 4 with round-trip time 66.4 ms
node 3 received ping answer from 4 with round-trip time 66.9 ms
node 3 received ping answer from 4 with round-trip time 66.4 ms
node 3 received ping answer from 4 with round-trip time 66.9 ms
node 3 received ping answer from 4 with round-trip time 66.4 ms
node 3 received ping answer from 4 with round-trip time 67.0 ms
node 3 received ping answer from 4 with round-trip time 66.5 ms
node 3 received ping answer from 4 with round-trip time 67.0 ms
node 3 received ping answer from 4 with round-trip time 66.5 ms
node 3 received ping answer from 4 with round-trip time 67.0 ms
node 3 received ping answer from 4 with round-trip time 66.5 ms
```

```
$awk  -f  numDrop.awk  prac2.tr
Number of packets dropped is = 41
```

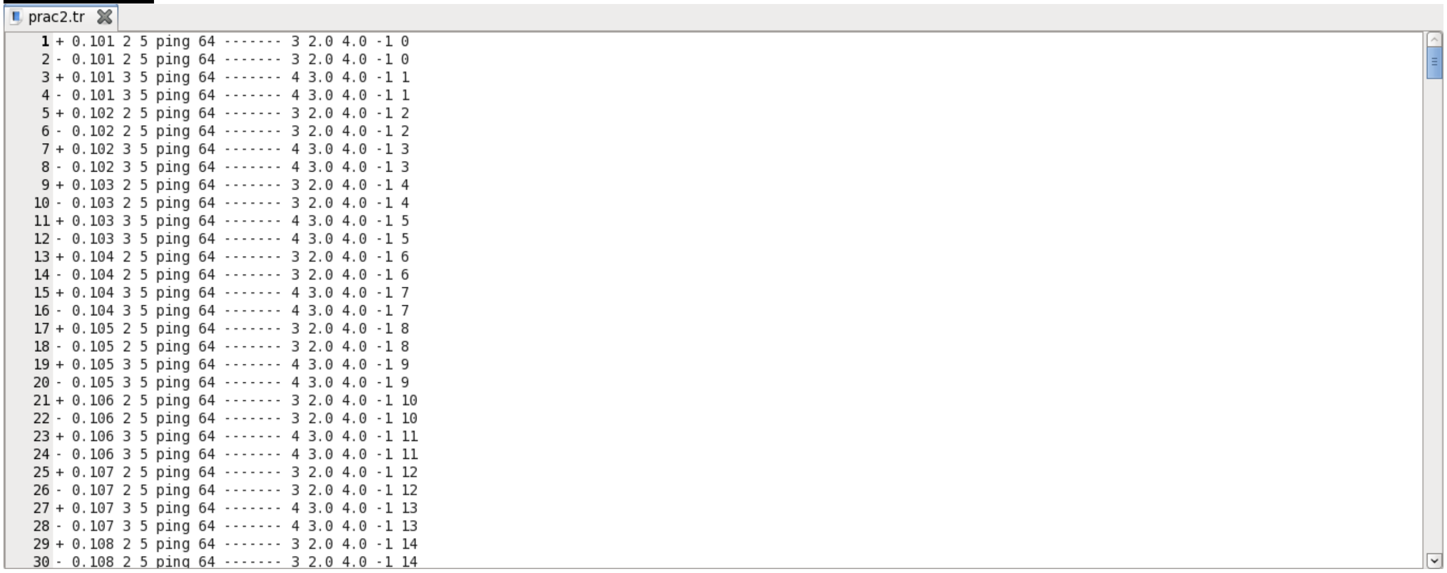
Simulation-



1.224000 Step: 2.0ms

Auto layout: Ca 0.15 Cr 0.75 Iterations 10 ☑Recalc

Trace File-

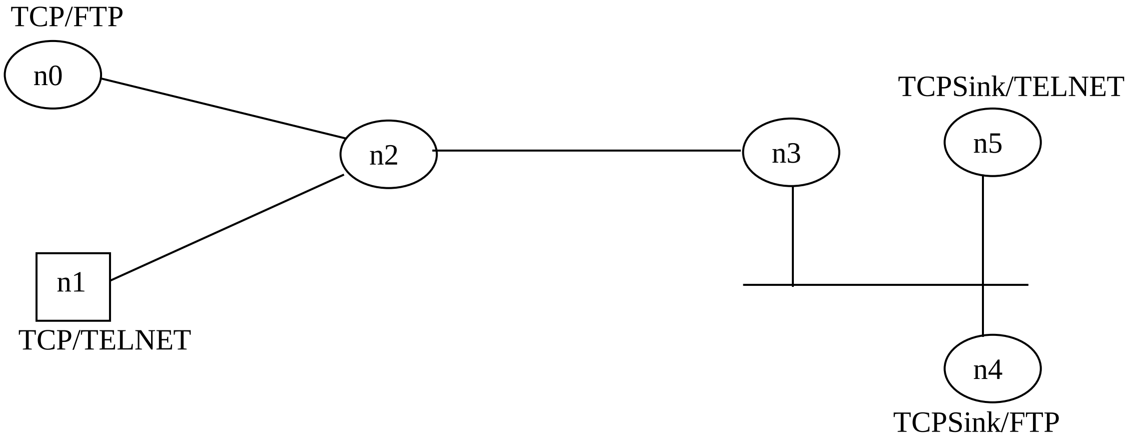


```
+ 0.101 2 5 ping 64 ------- 3 2.0 4.0 -1 0
- 0.101 2 5 ping 64 ------- 3 2.0 4.0 -1 0
+ 0.101 3 5 ping 64 ------- 4 3.0 4.0 -1 1
- 0.101 3 5 ping 64 ------- 4 3.0 4.0 -1 1
+ 0.102 2 5 ping 64 ------- 3 2.0 4.0 -1 2
- 0.102 2 5 ping 64 ------- 3 2.0 4.0 -1 2
+ 0.102 3 5 ping 64 ------- 4 3.0 4.0 -1 3
- 0.102 3 5 ping 64 ------- 4 3.0 4.0 -1 3
+ 0.103 2 5 ping 64 ------- 3 2.0 4.0 -1 4
- 0.103 2 5 ping 64 ------- 3 2.0 4.0 -1 4
+ 0.103 3 5 ping 64 ------- 4 3.0 4.0 -1 5
- 0.103 3 5 ping 64 ------- 4 3.0 4.0 -1 5
+ 0.104 2 5 ping 64 ------- 3 2.0 4.0 -1 6
- 0.104 2 5 ping 64 ------- 3 2.0 4.0 -1 6
+ 0.104 3 5 ping 64 ------- 4 3.0 4.0 -1 7
- 0.104 3 5 ping 64 ------- 4 3.0 4.0 -1 7
+ 0.105 2 5 ping 64 ------- 3 2.0 4.0 -1 8
- 0.105 2 5 ping 64 ------- 3 2.0 4.0 -1 8
+ 0.105 3 5 ping 64 ------- 4 3.0 4.0 -1 9
- 0.105 3 5 ping 64 ------- 4 3.0 4.0 -1 9
+ 0.106 2 5 ping 64 ------- 3 2.0 4.0 -1 10
- 0.106 2 5 ping 64 ------- 3 2.0 4.0 -1 10
+ 0.106 3 5 ping 64 ------- 4 3.0 4.0 -1 11
- 0.106 3 5 ping 64 ------- 4 3.0 4.0 -1 11
+ 0.107 2 5 ping 64 ------- 3 2.0 4.0 -1 12
- 0.107 2 5 ping 64 ------- 3 2.0 4.0 -1 12
+ 0.107 3 5 ping 64 ------- 4 3.0 4.0 -1 13
- 0.107 3 5 ping 64 ------- 4 3.0 4.0 -1 13
+ 0.108 2 5 ping 64 ------- 3 2.0 4.0 -1 14
- 0.108 2 5 ping 64 ------- 3 2.0 4.0 -1 14
```

Lab Experiment 3:

Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

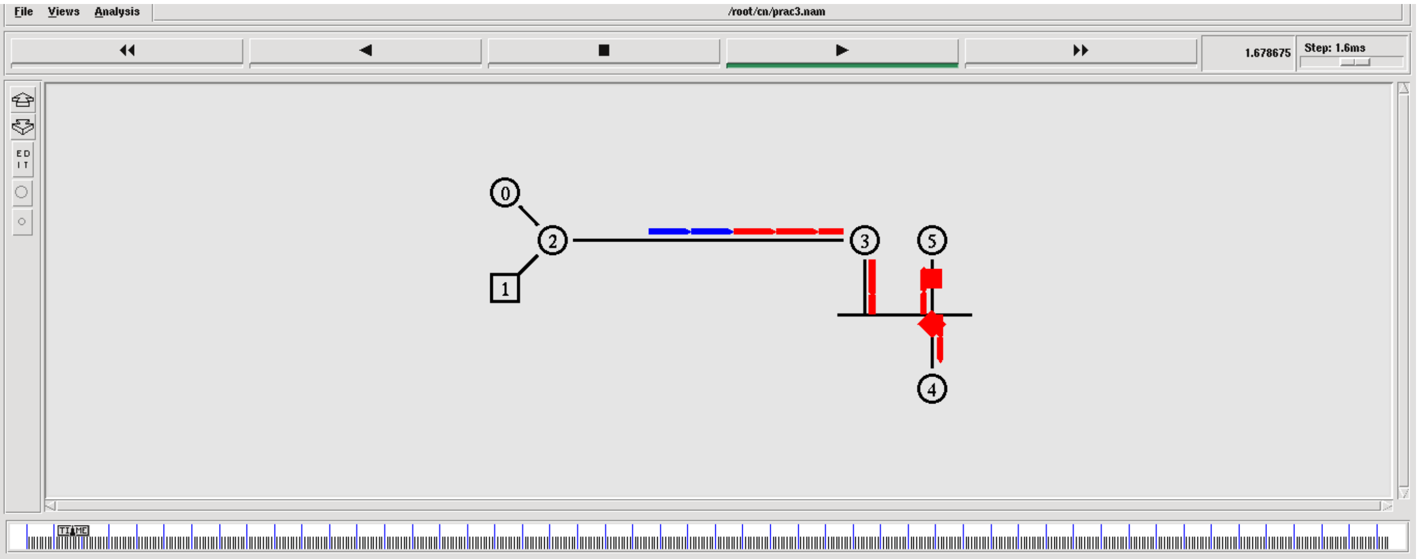
Topology-



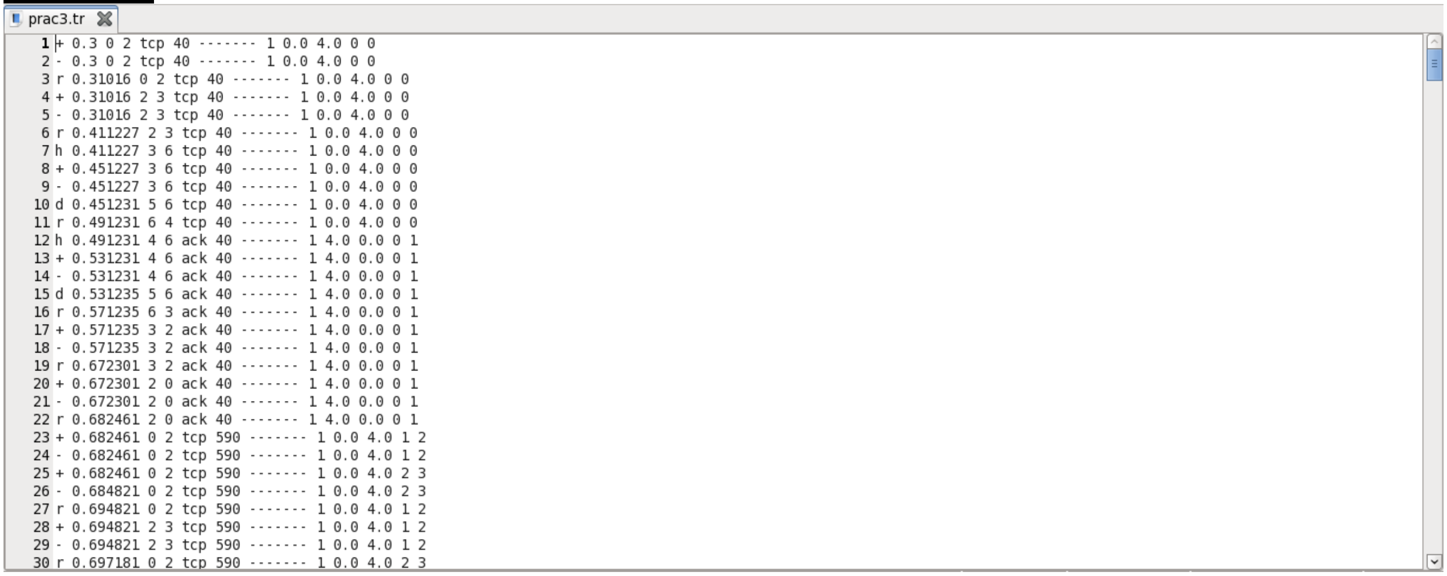
Code-

```
#set ns Simulator
set ns [new Simulator]

#define color for data flow
$ns color 1 Blue
$ns color 2 Red

#open trace file
set tracefile1 [open lab3.tr w]
set winfile [open winfile w]
$ns trace-all $tracefile1

#open namtrace file
set namfile [open lab3.nam w]
$ns namtrace-all $namfile

#define finish procedure
proc finish { } {
        global ns tracefile1 namfile
        $ns flush-trace
        close $tracefile1
        close $namfile
        exec nam lab3.nam &
        exit 0
}

#create 6 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 shape box
#create link  between 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/802_3]

#give node position
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns simplex-link-op $n3 $n2 orient left
$ns simplex-link-op $n2 $n3 orient right

#set queue size of link(n2-n3)
$ns queue-limit $n2 $n3 20

#setup tcp connection
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set packetSize_ 552

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

#setup a TCP1 connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n5 $sink1
$ns connect $tcp1 $sink1
$tcp1 set fid_ 2
$tcp1 set packetSize_ 552

set telnet0 [new Application/Telnet]
$telnet0 attach-agent $tcp1

#title congestion window1
set outfile1 [open congestion1.xg w]
puts $outfile1 "TitleText: Congestion Window-- Source _tcp"
puts $outfile1 "xUnitText: Simulation Time(Secs)"
puts $outfile1 "yUnitText: Congestion WindowSize"

#title congestion window2
set outfile2 [open congestion2.xg w]
puts $outfile2 "TitleText: Congestion Window-- Source _tcp1"
puts $outfile2 "xUnitText: Simulation Time(Secs)"
puts $outfile2 "yUnitText: Congestion WindowSize"

proc plotWindow {tcpSource outfile} {
        global ns
        set time 0.1
        set now [$ns now]
        set cwnd [$tcpSource set cwnd_]
        puts $outfile "$now $cwnd"
        $ns at [expr $now+$time] "plotWindow $tcpSource $outfile"
}

$ns at 0.1 "plotWindow $tcp $winfile"
$ns at 0.0 "plotWindow $tcp $outfile1"
$ns at 0.1 "plotWindow $tcp1 $outfile2"
$ns at 0.3 "$ftp start"
$ns at 0.5 "$telnet0 start"
$ns at 49.0 "$ftp stop"

$ns at 49.1 "$telnet0 stop"
$ns at 50.0 "finish"

$ns run
```

Simulation-



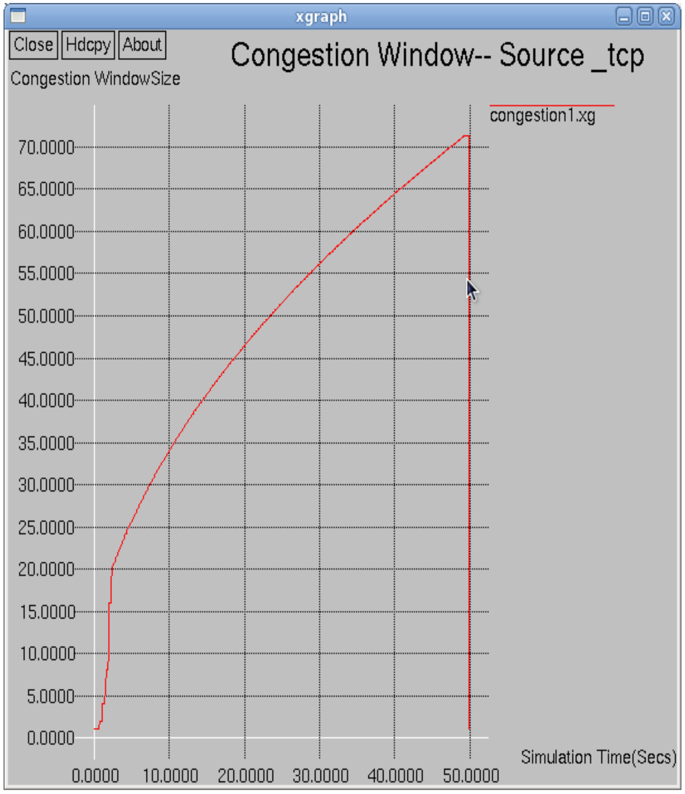
Trace File-



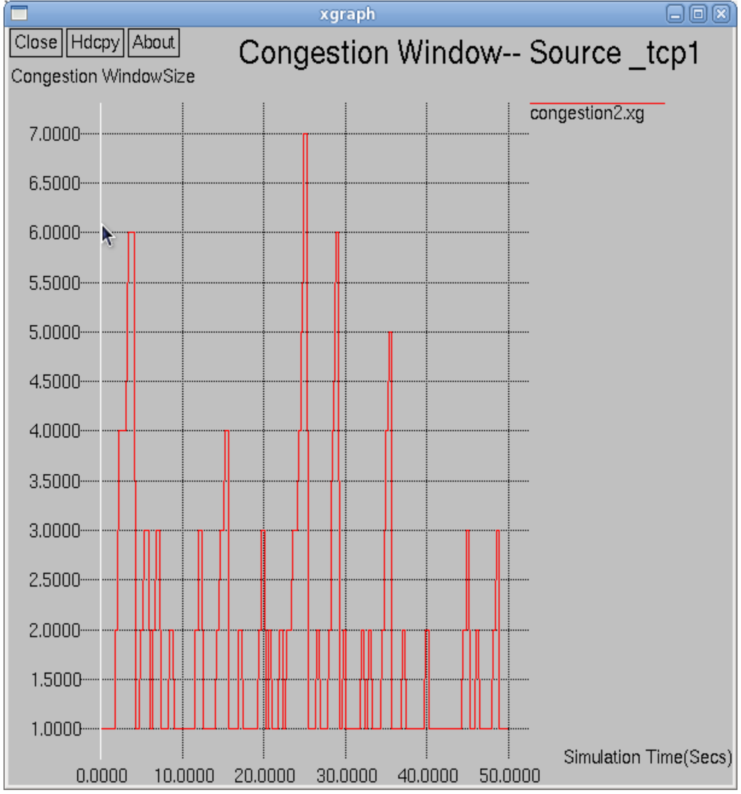
```
+ 0.3 0 2 tcp 40 ------- 1 0.0 4.0 0 0
- 0.3 0 2 tcp 40 ------- 1 0.0 4.0 0 0
r 0.31016 0 2 tcp 40 ------- 1 0.0 4.0 0 0
+ 0.31016 2 3 tcp 40 ------- 1 0.0 4.0 0 0
- 0.31016 2 3 tcp 40 ------- 1 0.0 4.0 0 0
r 0.411227 2 3 tcp 40 ------- 1 0.0 4.0 0 0
h 0.411227 3 6 tcp 40 ------- 1 0.0 4.0 0 0
+ 0.451227 3 6 tcp 40 ------- 1 0.0 4.0 0 0
- 0.451227 3 6 tcp 40 ------- 1 0.0 4.0 0 0
d 0.451231 5 6 tcp 40 ------- 1 0.0 4.0 0 0
r 0.491231 6 4 tcp 40 ------- 1 0.0 4.0 0 0
h 0.491231 4 6 ack 40 ------- 1 4.0 0.0 0 1
+ 0.531231 4 6 ack 40 ------- 1 4.0 0.0 0 1
- 0.531231 4 6 ack 40 ------- 1 4.0 0.0 0 1
d 0.531235 5 6 ack 40 ------- 1 4.0 0.0 0 1
r 0.571235 6 3 ack 40 ------- 1 4.0 0.0 0 1
+ 0.571235 3 2 ack 40 ------- 1 4.0 0.0 0 1
- 0.571235 3 2 ack 40 ------- 1 4.0 0.0 0 1
r 0.672301 3 2 ack 40 ------- 1 4.0 0.0 0 1
+ 0.672301 2 0 ack 40 ------- 1 4.0 0.0 0 1
- 0.672301 2 0 ack 40 ------- 1 4.0 0.0 0 1
r 0.682461 2 0 ack 40 ------- 1 4.0 0.0 0 1
+ 0.682461 0 2 tcp 590 ------- 1 0.0 4.0 1 2
- 0.682461 0 2 tcp 590 ------- 1 0.0 4.0 1 2
+ 0.682461 0 2 tcp 590 ------- 1 0.0 4.0 2 3
- 0.684821 0 2 tcp 590 ------- 1 0.0 4.0 2 3
r 0.694821 0 2 tcp 590 ------- 1 0.0 4.0 1 2
+ 0.694821 2 3 tcp 590 ------- 1 0.0 4.0 1 2
- 0.694821 2 3 tcp 590 ------- 1 0.0 4.0 1 2
r 0.697181 0 2 tcp 590 ------- 1 0.0 4.0 2 3
```

Congestion graph-

$ xgraph congestion1.xg



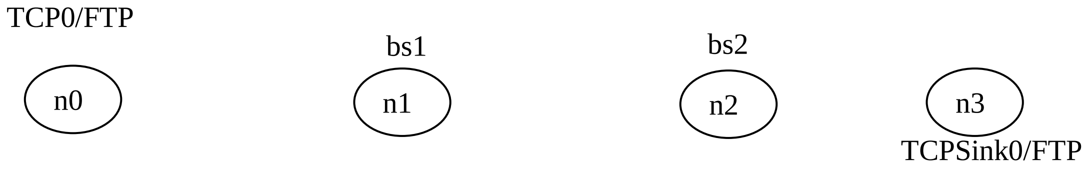
$ xgraph congestion2.xg



Lab Experiment 4 :

Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

Topology-



Code-

```
#create Simulator class
set ns [new Simulator]

#open trace file
set nt [open lab42.tr w]
$ns trace-all $nt

#create Topography object
set topo [new Topography]
#define grid size
$topo load_flatgrid 1000 1000

#open namtrace file
set nf [open lab42.nam w]
$ns namtrace-all-wireless $nf 1000 1000

#specify node configuration
$ns node-config -adhocRouting DSDV \
                -llType LL \
                -macType Mac/802_11 \
                -ifqType Queue/DropTail \
                -ifqLen 20 \
                -phyType Phy/WirelessPhy \
                -channelType Channel/WirelessChannel \
                -propType Propagation/TwoRayGround \
                -antType Antenna/OmniAntenna \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON

#create General Operation Director(god) object that stores total number of mobile nodes.
create-god 4
#create nodes and label them
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

$n0 label "tcp0"
$n1 label "sink0"
$n2 label "bs1"
$n3 label "bs2"

#give initial x, y, z coordinates to nodes
$n0 set X_ 110
$n0 set Y_ 500
$n0 set Z_ 0

$n1 set X_ 600
$n1 set Y_ 500
$n1 set Z_ 0

$n2 set X_ 300
$n2 set Y_ 500
$n2 set Z_ 0

$n3 set X_ 450
$n3 set Y_ 500
$n3 set Z_ 0

#attach agent and application to nodes and connect them
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0

set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1

$ns connect $tcp0 $sink1

#schedule the event
$ns at 0.5 "$ftp0 start"

#set up destination for mobile nodes. They move to <x><y> coordinates at <s>m/s.
$ns at 0.3 "$n0 setdest 110 500 10"
$ns at 0.3 "$n1 setdest 600 500 20"
$ns at 0.3 "$n2 setdest 300 500 30"
$ns at 0.3 "$n3 setdest 450 500 30"

$ns at 10.0 "$n0 setdest 100 550 5"
$ns at 10.0 "$n1 setdest 630 450 5"

$ns at 70.0 "$n0 setdest 170 680 5"
$ns at 70.0 "$n1 setdest 580 380 5"
```

```
$ns at 120.0 "$n0 setdest 140 720 5"

$ns at 135.0 "$n0 setdest 110 600 5"

$ns at 140.0 "$n1 setdest 600 550 5"

$ns at 155.0 "$n0 setdest 89 500 5"

$ns at 190.0 "$n0 setdest 100 440 5"

$ns at 210.0 "$n1 setdest 700 600 5"

$ns at 240.0 "$n1 setdest 650 500 5"

proc finish { } {
        global ns nt nf
        $ns flush-trace
        exec nam lab42.nam &
        close $nt
        close $nf
        exit 0
}

$ns at 400 "finish"
$ns run
```

Awk file-

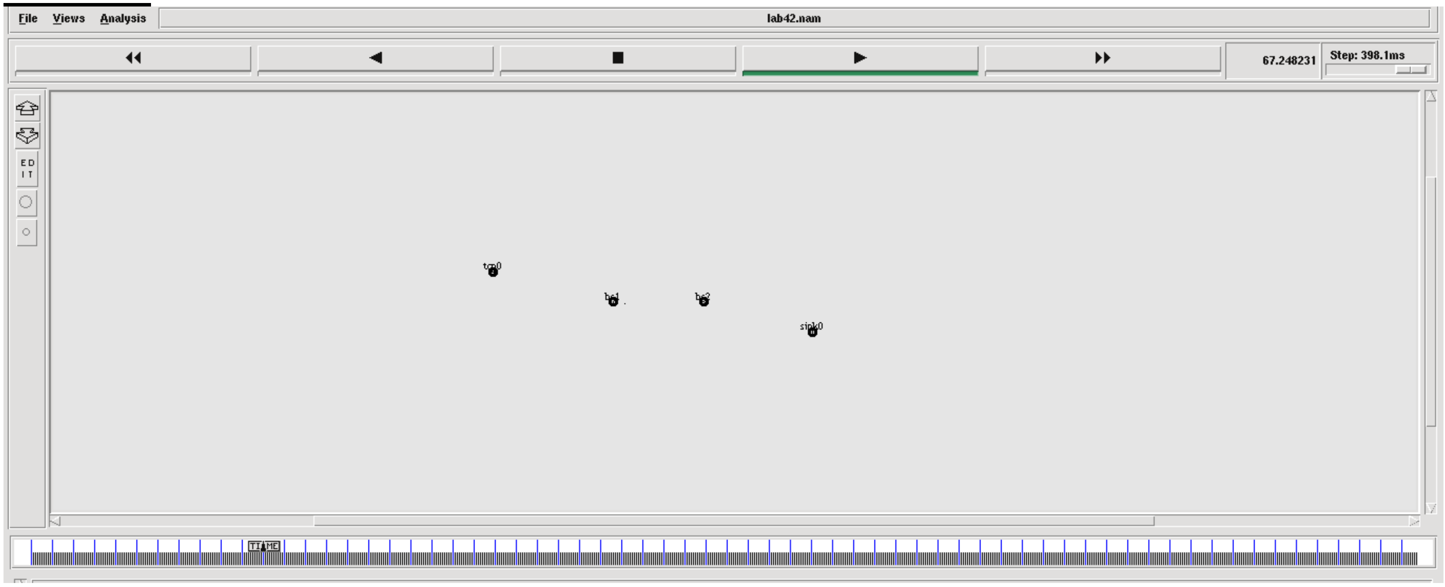
```
BEGIN{
        PktsSent=0;
        PktsRcvd=0;
        PktsAtRTR=0;
}

{
        if(($1=="s")&&($4=="RTR")&&($7=="tcp"))
        {
                PktsAtRTR++;
        }
        if(($1=="s")&&($4=="AGT")&&($7=="tcp"))
        {
                PktsSent++;
        }
        if(($1=="r")&&($4=="AGT")&&($7=="tcp"))
        {
                PktsRcvd++;
        }
}

END{
        print " Number of Packets Sent :" PktsSent
```

```
                print " Number of Packets Received :" PktsRcvd
                print " Pacjet Delivery Ratio :" PktsRcvd/PktsSent*100
                print " Routing Load :" PktsAtRTR/PktsRcvd
        }
```

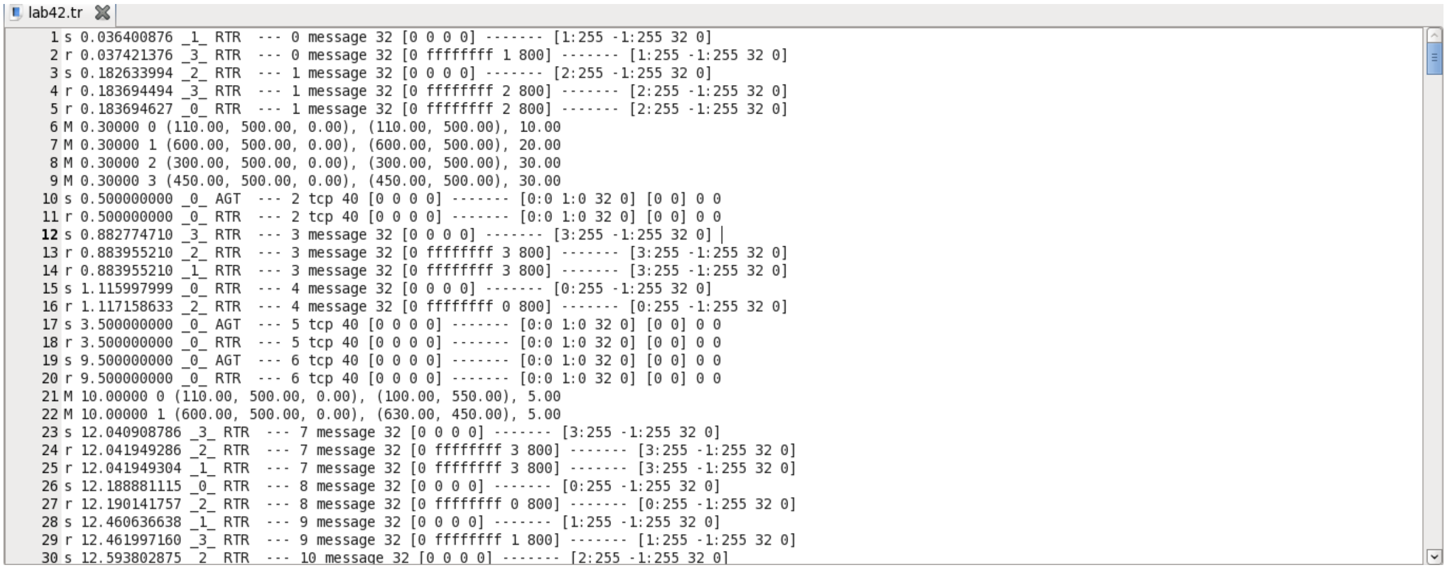
Output-

```
$awk -f count.awk lab42.tr
 Number of Packets Sent :6819
 Number of Packets Received :6685
 Pacjet Delivery Ratio :98.0349
 Routing Load :1.02004
```

Simulator-



Trace File-

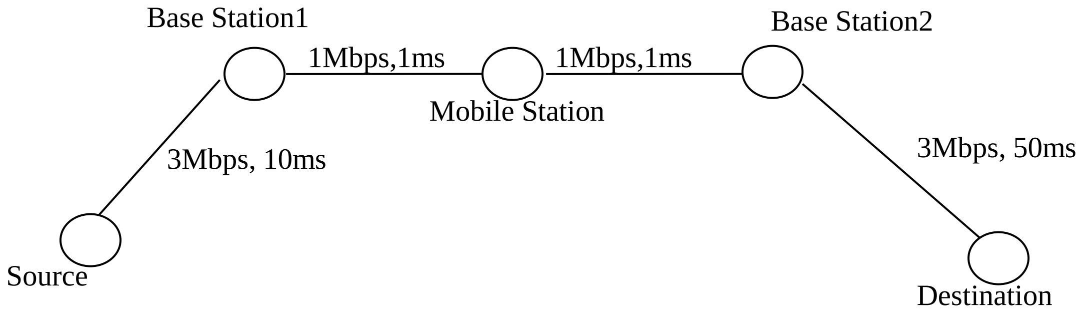


```
s 0.036400876 _1_ RTR  --- 0 message 32 [0 0 0 0] ------- [1:255 -1:255 32 0]
r 0.037421376 _3_ RTR  --- 0 message 32 [0 ffffffff 1 800] ------- [1:255 -1:255 32 0]
s 0.182633994 _2_ RTR  --- 1 message 32 [0 0 0 0] ------- [2:255 -1:255 32 0]
r 0.183694494 _3_ RTR  --- 1 message 32 [0 ffffffff 2 800] ------- [2:255 -1:255 32 0]
r 0.183694627 _0_ RTR  --- 1 message 32 [0 ffffffff 2 800] ------- [2:255 -1:255 32 0]
M 0.30000 0 (110.00, 500.00, 0.00), (110.00, 500.00), 10.00
M 0.30000 1 (600.00, 500.00, 0.00), (600.00, 500.00), 20.00
M 0.30000 2 (300.00, 500.00, 0.00), (300.00, 500.00), 30.00
M 0.30000 3 (450.00, 500.00, 0.00), (450.00, 500.00), 30.00
s 0.500000000 _0_ AGT  --- 2 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
r 0.500000000 _0_ RTR  --- 2 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
s 0.882774710 _3_ RTR  --- 3 message 32 [0 0 0 0] ------- [3:255 -1:255 32 0]
r 0.883955210 _2_ RTR  --- 3 message 32 [0 ffffffff 3 800] ------- [3:255 -1:255 32 0]
r 0.883955210 _1_ RTR  --- 3 message 32 [0 ffffffff 3 800] ------- [3:255 -1:255 32 0]
s 1.115997999 _0_ RTR  --- 4 message 32 [0 0 0 0] ------- [0:255 -1:255 32 0]
r 1.117158633 _2_ RTR  --- 4 message 32 [0 ffffffff 0 800] ------- [0:255 -1:255 32 0]
s 3.500000000 _0_ AGT  --- 5 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
r 3.500000000 _0_ RTR  --- 5 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
s 9.500000000 _0_ AGT  --- 6 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
r 9.500000000 _0_ RTR  --- 6 tcp 40 [0 0 0 0] ------- [0:0 1:0 32 0] [0 0] 0 0
M 10.00000 0 (110.00, 500.00, 0.00), (100.00, 550.00), 5.00
M 10.00000 1 (600.00, 500.00, 0.00), (630.00, 450.00), 5.00
s 12.040908786 _3_ RTR  --- 7 message 32 [0 0 0 0] ------- [3:255 -1:255 32 0]
r 12.041949286 _2_ RTR  --- 7 message 32 [0 ffffffff 3 800] ------- [3:255 -1:255 32 0]
r 12.041949304 _1_ RTR  --- 7 message 32 [0 ffffffff 3 800] ------- [3:255 -1:255 32 0]
s 12.188881115 _0_ RTR  --- 8 message 32 [0 0 0 0] ------- [0:255 -1:255 32 0]
r 12.190141757 _2_ RTR  --- 8 message 32 [0 ffffffff 0 800] ------- [0:255 -1:255 32 0]
s 12.460636638 _1_ RTR  --- 9 message 32 [0 0 0 0] ------- [1:255 -1:255 32 0]
r 12.461997160 _3_ RTR  --- 9 message 32 [0 ffffffff 1 800] ------- [1:255 -1:255 32 0]
s 12.593802875 _2_ RTR  --- 10 message 32 [0 0 0 0] ------- [2:255 -1:255 32 0]
```

Lab Experiment 5 :

Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

Topology-

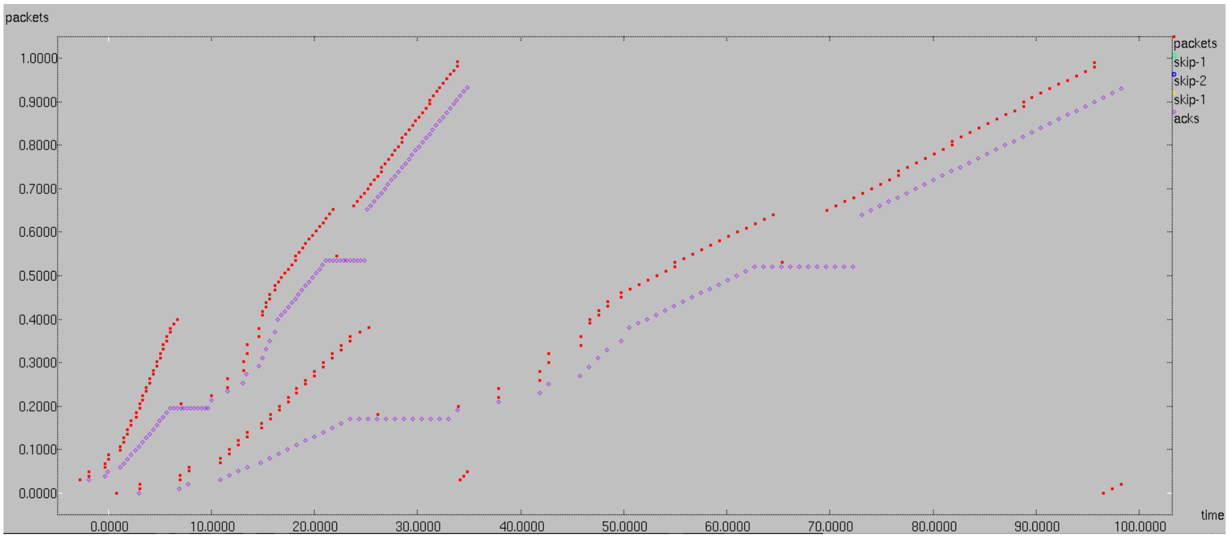


Code –

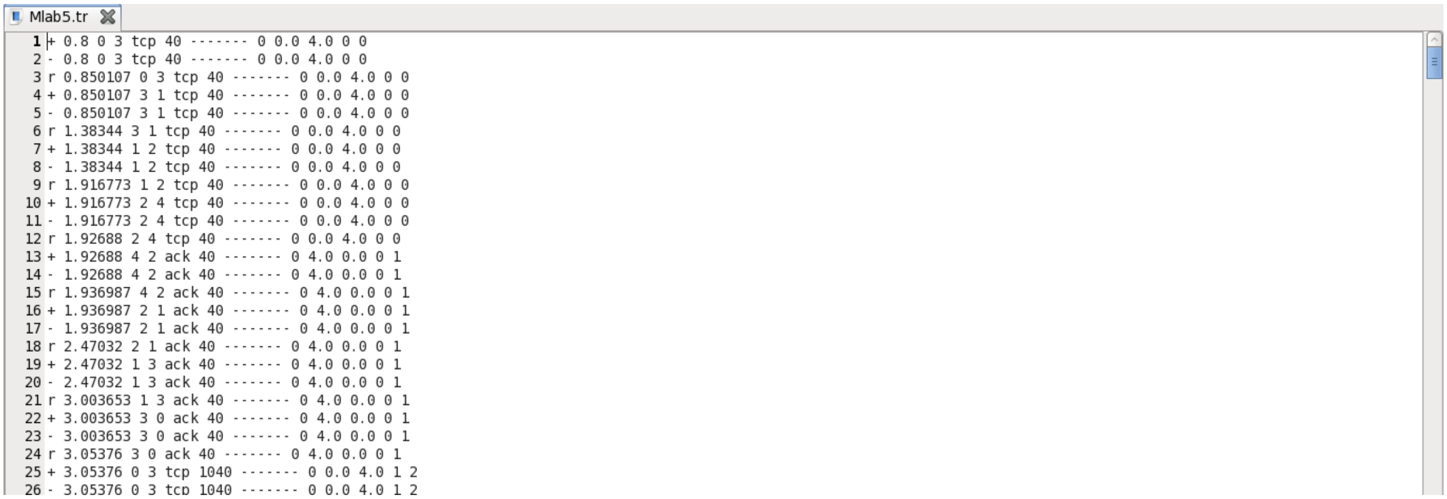
```
#General parameters
set stop 100
set type gsm

#AQM parameters
set minth 30
set maxth 0
set adaptive 1

#traffic generation
set flows 0
set window 30

#plotting statistics
set opt(wrap) 100
set opt(srcTrace) is
set opt(dstTrace) bs2

#default downlink bandwidth in bps
set bwDL(gsm) 9600
#default propogation delay in sec
set propDL(gsm) .500

set ns [new Simulator]

set tf [open Mlab5.tr w]
$ns trace-all $tf

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]
```

```
proc cell_topo {} {
  global ns nodes
        $ns duplex-link $nodes(lp) $nodes(bs1) 3Mbps 10ms DropTail
        $ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
        $ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
        $ns duplex-link $nodes(bs2) $nodes(is) 3Mbps 50ms DropTail
        puts "GSM Cell Topology"
        }
proc set_link_params {t} {

global ns nodes bwDL propDL
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

$ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

$ns queue-limit $nodes(bs1) $nodes(ms) 10
$ns queue-limit $nodes(bs2) $nodes(ms) 10
}

#RED and TCP parameters
Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth
Agent/TCP set window_ $window

#create topology
switch $type {
        gsm -
        umts {cell_topo}
}

set_link_params $type
$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]

# set up TCP connection
if {$flows == 0} {
        set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
        set ftp1 [[set tcp1] attach-app FTP]
         $ns at 0.8 "[set ftp1] start"
         }

 proc stop {} {
        global nodes opt tf
        set wrap $opt(wrap)
        set sid [$nodes($opt(srcTrace)) id]
        set did [$nodes($opt(dstTrace)) id]
        set a "Mlab5.tr"

        set GETRC "../bin/getrc"
        set RAW2XG "../bin/raw2xg"

        exec $GETRC -s $sid -d $did -f 0 Mlab5.tr | \
        $RAW2XG -s 0.01 -m $wrap -r > plot.xgr

        exec $GETRC -s $did -d $sid -f 0 Mlab5.tr | \
        $RAW2XG -a -s 0.01 -m $wrap >> plot.xgr

        exec xgraph -x time -y packets plot.xgr &
        exit 0
        }
         $ns at $stop "stop"
        $ns run
```

Graph-



Trace File-

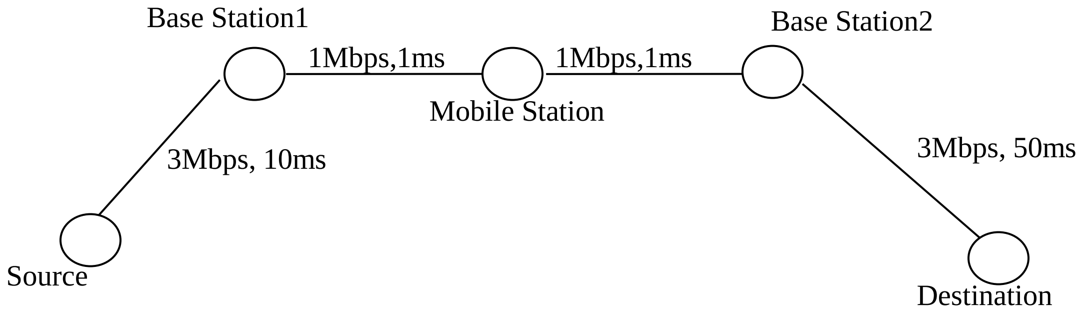


```
+ 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ------- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
r 1.38344 3 1 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.38344 1 2 tcp 40 ------- 0 0.0 4.0 0 0
- 1.38344 1 2 tcp 40 ------- 0 0.0 4.0 0 0
r 1.916773 1 2 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.916773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
- 1.916773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
r 1.92688 2 4 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.92688 4 2 ack 40 ------- 0 4.0 0.0 0 1
- 1.92688 4 2 ack 40 ------- 0 4.0 0.0 0 1
r 1.936987 4 2 ack 40 ------- 0 4.0 0.0 0 1
+ 1.936987 2 1 ack 40 ------- 0 4.0 0.0 0 1
- 1.936987 2 1 ack 40 ------- 0 4.0 0.0 0 1
r 2.47032 2 1 ack 40 ------- 0 4.0 0.0 0 1
+ 2.47032 1 3 ack 40 ------- 0 4.0 0.0 0 1
- 2.47032 1 3 ack 40 ------- 0 4.0 0.0 0 1
r 3.003653 1 3 ack 40 ------- 0 4.0 0.0 0 1
+ 3.003653 3 0 ack 40 ------- 0 4.0 0.0 0 1
- 3.003653 3 0 ack 40 ------- 0 4.0 0.0 0 1
r 3.05376 3 0 ack 40 ------- 0 4.0 0.0 0 1
+ 3.05376 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
- 3.05376 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
```

Lab Experiment 6 :

Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.

Topology-

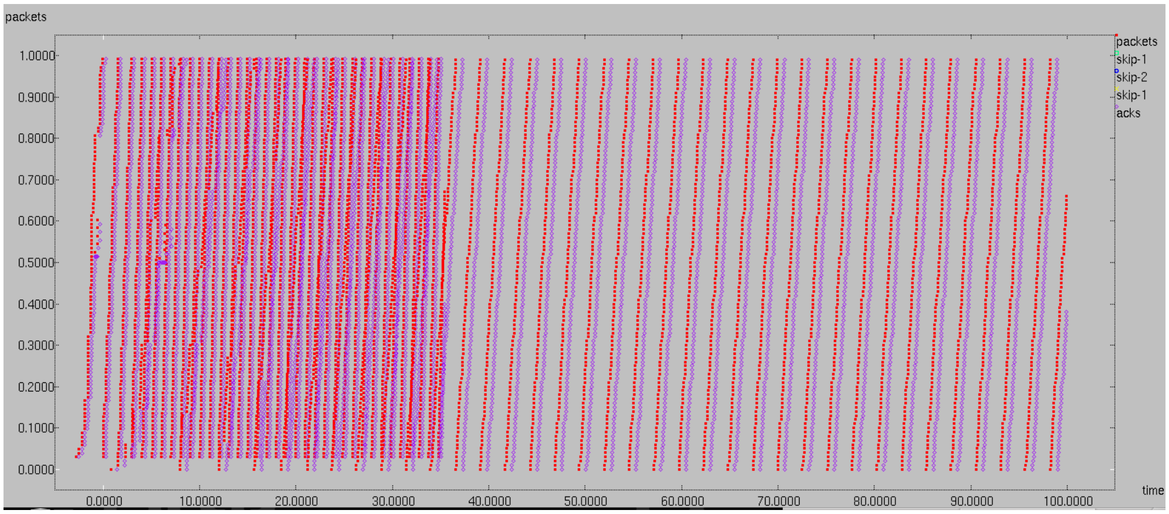


Code –

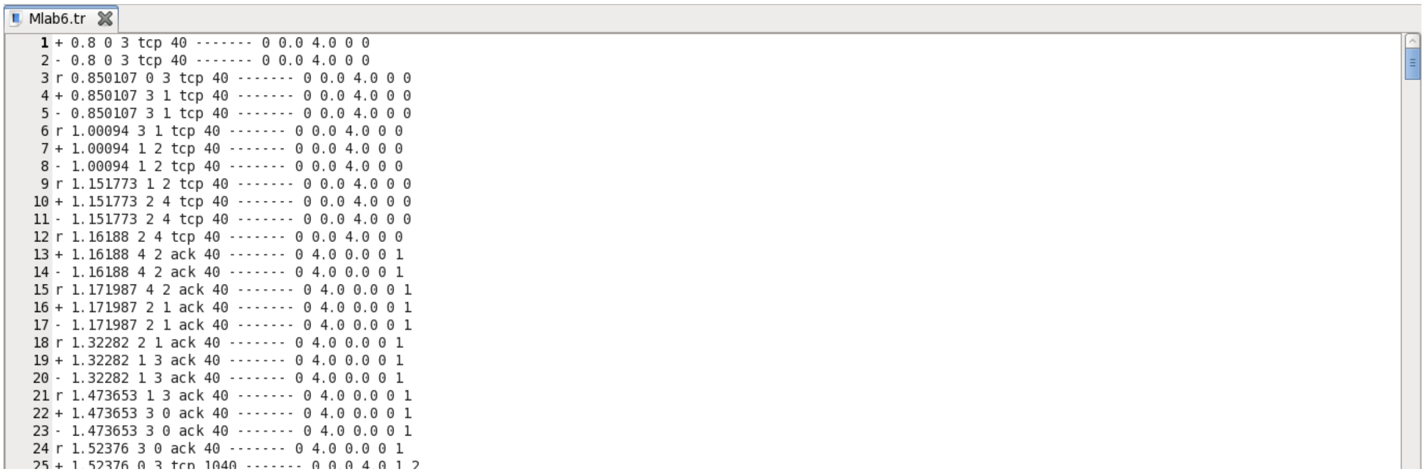
```
#General parameters
set stop 100
set type umts

#AQM parameters
set minth 30
set maxth 0
set adaptive 1

#traffic generation
set flows 0
set window 30

#plotting statistics
set opt(wrap) 100
set opt(srcTrace) is
set opt(dstTrace) bs2

#default downlink bandwidth in bps
set bwDL(umts) 38400
#default propogation delay in sec
set propDL(umts) .150


set ns [new Simulator]

set tf [open Mlab6.tr w]
$ns trace-all $tf

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]
```

```
proc cell_topo {} {
    global ns nodes
            $ns duplex-link $nodes(lp) $nodes(bs1) 3Mbps 10ms DropTail
            $ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
            $ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
            $ns duplex-link $nodes(bs2) $nodes(is) 3Mbps 50ms DropTail
            puts "umts Cell Topology"
            }
proc set_link_param {t} {

global ns nodes bwDL propDL
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

$ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

$ns queue-limit $nodes(bs1) $nodes(ms) 20
$ns queue-limit $nodes(bs2) $nodes(ms) 20
}

#set RED and TCP parameters
Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth
Agent/TCP set window_ $window

#create topology
switch $type {
            umts {cell_topo}
}

set_link_param $type
$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]

#set up TCP connection
if {$flows == 0 } {
            set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
            set ftp1 [[set tcp1] attach-app FTP]
              $ns at 0.8 "[set ftp1] start"
              }

  proc stop {} {
            global nodes opt tf
            set wrap $opt(wrap)
            set sid [$nodes($opt(srcTrace)) id]
            set did [$nodes($opt(dstTrace)) id]

            set a "Mlab6.tr"
            set GETRC "../bin/getrc"
            set RAW2XG "../bin/raw2xg"

            exec $GETRC -s $sid -d $did -f 0 Mlab6.tr | \
            $RAW2XG -s 0.01 -m $wrap -r > plot6.xgr

            exec $GETRC -s $did -d $sid -f 0 Mlab6.tr | \
            $RAW2XG -a -s 0.01 -m $wrap >> plot6.xgr

            exec xgraph -x time -y packets plot6.xgr &
            exit 0
            }
             $ns at $stop "stop"
            $ns run
```

Graph-



Trace File-



```
+ 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
- 0.8 0 3 tcp 40 ------- 0 0.0 4.0 0 0
r 0.850107 0 3 tcp 40 ------- 0 0.0 4.0 0 0
+ 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
- 0.850107 3 1 tcp 40 ------- 0 0.0 4.0 0 0
r 1.00094 3 1 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.00094 1 2 tcp 40 ------- 0 0.0 4.0 0 0
- 1.00094 1 2 tcp 40 ------- 0 0.0 4.0 0 0
r 1.151773 1 2 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.151773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
- 1.151773 2 4 tcp 40 ------- 0 0.0 4.0 0 0
r 1.16188 2 4 tcp 40 ------- 0 0.0 4.0 0 0
+ 1.16188 4 2 ack 40 ------- 0 4.0 0.0 0 1
- 1.16188 4 2 ack 40 ------- 0 4.0 0.0 0 1
r 1.171987 4 2 ack 40 ------- 0 4.0 0.0 0 1
+ 1.171987 2 1 ack 40 ------- 0 4.0 0.0 0 1
- 1.171987 2 1 ack 40 ------- 0 4.0 0.0 0 1
r 1.32282 2 1 ack 40 ------- 0 4.0 0.0 0 1
+ 1.32282 1 3 ack 40 ------- 0 4.0 0.0 0 1
- 1.32282 1 3 ack 40 ------- 0 4.0 0.0 0 1
r 1.473653 1 3 ack 40 ------- 0 4.0 0.0 0 1
+ 1.473653 3 0 ack 40 ------- 0 4.0 0.0 0 1
- 1.473653 3 0 ack 40 ------- 0 4.0 0.0 0 1
r 1.52376 3 0 ack 40 ------- 0 4.0 0.0 0 1
+ 1.52376 0 3 tcp 1040 ------- 0 0.0 4.0 1 2
```

Lab Program 7 :

Write a program for error detecting code using CRC-CCITT (16- bits).

Code –

```
import java.util.Scanner;

public class CRC {

    public static int n;

    public static void main(String[] args)
    {
        Scanner sc=new Scanner(System.in);

        CRC crc=new CRC();

        String copy,rec,code,zero="0000000000000000";

        System.out.println("enter the dataword to be sent");
        code=sc.nextLine();

        n=code.length();

        copy=code;
        code+=zero;
        code=crc.divide(code);

        System.out.println("dataword="+copy);

        copy=copy.substring(0,n)+code.substring(n);

        System.out.print("CRC=");
        System.out.println(code.substring(n));

        System.out.println("transmitted frame is="+copy);

        System.out.println("enter received data:");
        rec=sc.nextLine();

        if(zero.equals(crc.divide(rec).substring(n)))
        System.out.println("correct bits received");
        else
            System.out.println("received frame contains one or more error");

        sc.close();
    }

    public String divide(String s)
    {
        String div="10001000000100001";
        int i,j;
        char x;

        for(i=0;i<n;i++)
        {
            x=s.charAt(i);

            for(j=0;j<17;j++)
            {
                if(x=='1')
                {
                    if(s.charAt(i+j)!=div.charAt(j))
                        s=s.substring(0,i+j)+"1"+s.substring(i+j+1);
                    else
                        s=s.substring(0,i+j)+"0"+s.substring(i+j+1);
                }
            }
        }
        return s;
    }
}
```

Output 1 –

```
enter the dataword to be sent
1100
dataword=1100
CRC=1100000110001100
transmitted frame is=11001100000110001100
enter received data:
11001100000010001100
received frame contains one or more error
```

Output 2 –

```
enter the dataword to be sent
1100
dataword=1100
CRC=1100000110001100
transmitted frame is=11001100000110001100
enter received data:
11001100000110001100
correct bits received
```

Output 3 –

```
enter the dataword to be sent
1101
dataword=1101
CRC=1101000110101101
transmitted frame is=11011101000110101101
enter received data:
11011001000110110010
received frame contains one or more error
```

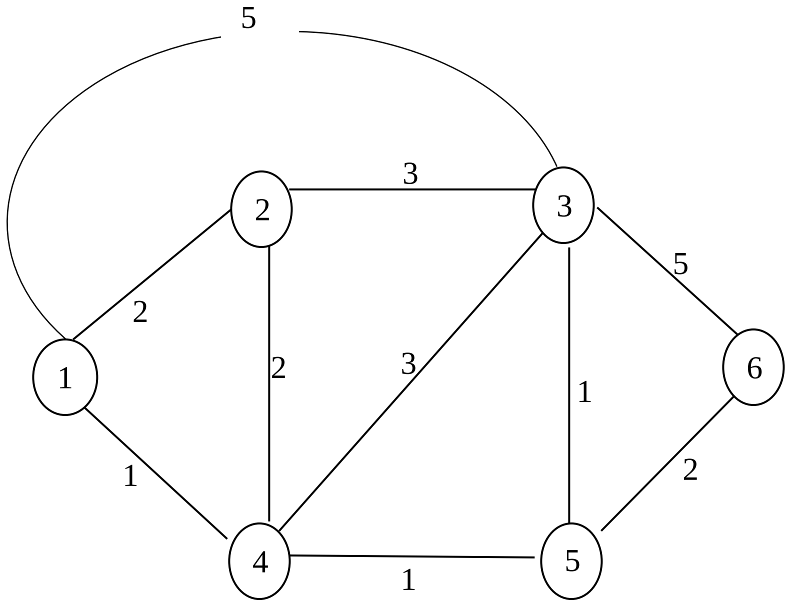
Lab Program 8 :

Write a program to find the shortest path between vertices using bellman-ford algorithm.

Code –

```
import java.util.Scanner;

public class bellmanford
{

    public int distance[];
    public int numb_vert;
    public static final int MAX_VALUE=999;

    public bellmanford(int numb_vert)
    {
        this.numb_vert = numb_vert;
        distance = new int[numb_vert+1];
    }

    public void BellmanfordpEvaluation(int source,int adj_matrix[][])
    {

        for(int node=1;node<=numb_vert;node++)
            distance[node]=MAX_VALUE;
        distance[source]=0;

        for(int node=1;node<=numb_vert-1;node++)
        {
            for(int src_node=1;src_node<=numb_vert;src_node++)
            {
                for(int dest_node=1;dest_node<=numb_vert;dest_node++)
                {
                    if(adj_matrix[src_node][dest_node]!=MAX_VALUE)
                    {
                        if(distance[dest_node] > distance[src_node] +
                                    adj_matrix[src_node][dest_node])

                            distance[dest_node] = distance[src_node] +
                                    adj_matrix[src_node][dest_node];
                    }
                }
            }
        }

        for(int src_node=1;src_node<=numb_vert;src_node++)
        {
            for(int dest_node=1;dest_node<=numb_vert;dest_node++)
            {
                if(adj_matrix[src_node][dest_node]!=MAX_VALUE)
                {
                    if(distance[dest_node] > distance[src_node] +
                                    adj_matrix[src_node][dest_node])
                    {
                        System.out.println("The graph contains negative edge cycle");

                    }
                }
            }
        }

        System.out.println("Routing Table for Router " + source+" is");

        System.out.println("Destination        Distance\t");
        for(int vertex=1;vertex<=numb_vert;vertex++)
            System.out.println(+vertex+"\t\t\t"+distance[vertex]);

    }

    public static void main(String args[])
    {

        int numb_vert=0;
        int source;
        Scanner scan = new Scanner(System.in);

        System.out.println("Enter the number of vertices");
        numb_vert = scan.nextInt();

        int adj_matrix[][] = new int[numb_vert+1][numb_vert+1];
        System.out.println("Enter the adjacency matrix");
        for(int src_node=1;src_node<=numb_vert;src_node++)
            for(int dest_node=1;dest_node<=numb_vert;dest_node++)
            {
                adj_matrix[src_node][dest_node] = scan.nextInt();
                if(src_node==dest_node)
                {
                    adj_matrix[src_node][dest_node]=0;
                    continue;
                }

                if(adj_matrix[src_node][dest_node]==0)
                    adj_matrix[src_node][dest_node]=MAX_VALUE;
            }

        for(int i=1;i<=numb_vert;i++)
        {
            bellmanford bellmanford = new bellmanford(numb_vert);
            bellmanford.BellmanfordpEvaluation(i,adj_matrix);
        }
        scan.close();
    }
}
```

Output 1 –

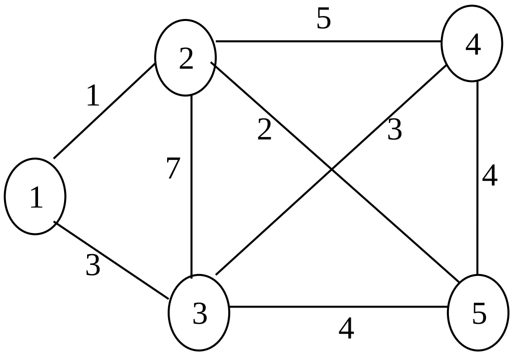


```
Enter the number of vertices
6
Enter the adjacency matrix
0     2     5 1     999 999
2     0     3 2      999 999
5     3     1 3     1     5
1     2     3 0     1     999
999 999 1 1     0     2
999 999 5 999 2     0
Routing Table for Router 1 is
Destination         Distance
1                              0
2                              2
3                              3
4                              1
5                              2
6                              4
Routing Table for Router 2 is
Destination         Distance
1                              2
2                              0
3                              3
4                              2
5                              3
6                              5
Routing Table for Router 3 is
Destination         Distance
1                              3
2                              3
3                              0
4                              2
5                              1
6                              3
Routing Table for Router 4 is
Destination         Distance
1                              1
2                              2
3                              2
4                              0
5                              1
6                              3
Routing Table for Router 5 is
Destination         Distance
1                              2
2                              3
3                              1
4                              1
5                              0
6                              2
Routing Table for Router 6 is
```

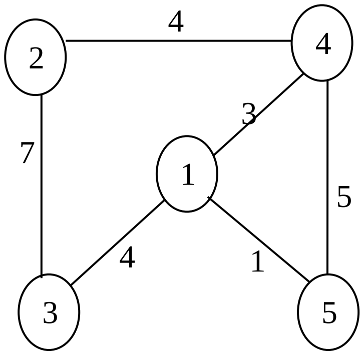
```
Destination       Distance
1                        4
2                        5
3                        3
4                        3
5                        2
6                        0
```

Output 2 –

```
Enter the number of vertices
5
Enter the adjacency matrix
0    1    3   999 999
1    0    7   5    2
3    7    0   3    4
999 5    3   0    4
999 2    4   4    0
Routing Table for Router 1 is
Destination       Distance
1                        0
2                        1
3                        3
4                        6
5                        3
Routing Table for Router 2 is
Destination       Distance
1                        1
2                        0
3                        4
4                        5
5                        2
Routing Table for Router 3 is
Destination       Distance
1                        3
2                        4
3                        0
4                        3
5                        4
Routing Table for Router 4 is
Destination       Distance
1                        6
2                        5
3                        3
4                        0
5                        4
Routing Table for Router 5 is
Destination       Distance
1                        3
2                        2
3                        4
4                        4
5                        0
```



Output 3 –



```
Enter the number of vertices
5
Enter the adjacency matrix
0     999 3     1     4
999 0     4     999 7
3     4     0     5     999
1     999 5     0     999
4     7     999 999 0
Routing Table for Router 1 is
Destination        Distance
1                          0
2                          7
3                          3
4                          1
5                          4
Routing Table for Router 2 is
Destination        Distance
1                          7
2                          0
3                          4
4                          8
5                          7
Routing Table for Router 3 is
Destination        Distance
1                          3
2                          4
3                          0
4                          4
5                          7
Routing Table for Router 4 is
Destination        Distance
1                          1
2                          8
3                          4
4                          0
5                          5
Routing Table for Router 5 is
Destination        Distance
1                          4
2                          7
3                          7
4                          5
5                          0
```

Lab Program 9 :

Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

Code –

```
import java.util.*;
import java.net.*;
import java.io.*;

public class tcpclient
{

        public static void main(String args[])
        {

            try
            {
                Scanner ser=new Scanner(System.in);
                Socket s=new Socket("localhost",998);

                DataInputStream dis=new DataInputStream(s.getInputStream());
                DataOutputStream dos=new DataOutputStream (s.getOutputStream());

                dos.writeUTF("connected to 127.0.0.1 \n");

                System.out.println(dis.readUTF());

                System.out.println("\n enter the full path of the the file to be displayed");
                String path=ser.nextLine();

                dos.writeUTF(path);
                System.out.println(new String (dis.readUTF()));

                dis.close();
                dos.close();

                s.close();
                ser.close();
            }
            catch(IOException e)
            {
                System.out.println("IO: "+e.getMessage());
            }

        }

}
```

```
import java.util.*;
import java.net.*;
import java.io.*;
public class tcpserver
{
        public static void main (String args[])
        {
                try
                {
                        ServerSocket s=new ServerSocket(998);

                        System.out.println("server ready \n waiting for connection \n");

                        Socket s1=s.accept();

                        DataOutputStream dos=new DataOutputStream(s1.getOutputStream());
                        DataInputStream dis=new DataInputStream(s1.getInputStream());

                        System.out.println(dis.readUTF());

                        dos.writeUTF("connected to server \n");

                        String path=dis.readUTF();
                        System.out.println("\n request received \n processing.......");
                        try
                        {
                                File myfile=new File(path);
                                Scanner scr=new Scanner(myfile);
                                String st=scr.nextLine();
                                st="\n the context of file is \n "+st;
                                while(scr.hasNextLine())
                                {
                                        st=st + "\n" + scr.nextLine();
                                }
                                dos.writeUTF(st);
                                dos.close();
                                s1.close();
                                scr.close();
                        }
                        catch(FileNotFoundException e)
                        {
                                System.out.println("\n,,,error...\n file not found");
                                dos.writeUTF("...error \n file not found");
                        }
                }
                catch(IOException e)
                {
                        System.out.println("IO: "+e.getMessage());
                }
                finally
                {
                        System.out.println("\n connection terminated");
                }
            }
    }
```

Output –

Client Side

```
$ javac tcpclient.java
$java tcpclient
connected to server


 enter the full path of the the file to be displayed
/root/cn/udp_tcp/text

 the context of file is
 Hi
how are you
tcp/ip program
112233
```

Server Side

```
$javac tcpserver.java
$ java tcpserver
server ready
 waiting for connection

connected to 127.0.0.1


 request received
 processing.......

 connection terminated
```

Lab Program 10(a):

Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.

Code –

```
import java.net.*;
import java.io.*;

public class UDPClient
{
        public static void main(String args[])
        {
                DatagramSocket aSocket=null;
                int clientPort=998;

                try
                {
                        aSocket=new DatagramSocket(clientPort);
                        byte[] buf=new byte[1000];

                        DatagramPacket data=new DatagramPacket(buf,buf.length);
                        System.out.println("Waiting for server\n");

                        aSocket.receive(data);
                        byte[] msg=new byte[1000];
                        msg=data.getData();
                        System.out.println("\n msg:"+(new String(msg,0,data.getLength())));

                }
                catch(SocketException e)
                {
                        System.out.println("Socket:" +e.getMessage());
                }
                catch(IOException e)
                {
                        System.out.println("IO:" +e.getMessage());
                }
                finally
                {
                        if(aSocket!=null)
                                aSocket.close();
                }

        }

}
```

```
import java.net.*;
import java.util.*;
import java.io.*;

public class UDPServer {

        public static void main(String args[])
        {

            DatagramSocket aSocket = null;
            Scanner scn=new Scanner(System.in);
            int serverPort =999;

            System.out.println("Server Ready\n Waiting for connection....\n");

            try
            {

                        aSocket=new DatagramSocket(serverPort);

                       byte[] buffer=new byte[1000];

                       System.out.println("\nEnter message to be sent:");
                      String str=scn.nextLine();

                       buffer=str.getBytes();

                       DatagramPacket data = new DatagramPacket(buffer,buffer.length,
                                                               InetAddress.getLocalHost(),998);
                       aSocket.send(data);

            }
            catch(SocketException e)
            {
                       System.out.println("Socket:"+e.getMessage());
            }
            catch(IOException e)
            {
               System.out.println("Io:"+e.getMessage());
            }
            finally
            {
                       System.out.println("\nMessage sent\nConnection terminated");

                        if(aSocket!=null)
                          aSocket.close();

                        scn.close();
            }

        }
}
```

Output –

Client Side

```
$ javac UDPClient.java
$# java UDPClient
Waiting for server


 msg:hello, this is server
```

Server Side

```
$javac UDPServer.java
$ java UDPServer
Server Ready
 Waiting for connection....


Enter message to be sent:
hello, this is server

Message sent
Connection terminated
```

Lab Program 11:

Write a program for simple RSA algorithm to encrypt and decrypt the data.

Code –

```
import java.math.BigInteger;
import java.util.Random;

public class rsalab
{
        private BigInteger p;
        private BigInteger q;
        private BigInteger n;
        private BigInteger phi;
        private BigInteger e,d;
        private int bitlength=256;

        private Random r;
        long p1;
        public rsalab()
        {
                r=new Random();

                p=BigInteger.probablePrime(bitlength, r);
                q=BigInteger.probablePrime(bitlength, r);
                n=p.multiply(q);
                phi=p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
                e=BigInteger.probablePrime(bitlength/2, r);

                while(phi.gcd(e).compareTo(BigInteger.ONE)>0 && e.compareTo(phi)<0)
                {
                        e.add(BigInteger.ONE);
                }

                d=e.modInverse(phi);

        }

        public rsalab(BigInteger e,BigInteger d,BigInteger n)
        {
                this.e=e;
                this.d=d;
                this.n=n;
        }

        public byte[] encrypt(byte[] message)
        {
                return (new BigInteger(message)).modPow(e,n).toByteArray();
        }
```

```
    public byte[] decrypt(byte[] message)
    {
        return (new BigInteger(message)).modPow(d,n).toByteArray();
    }
}


import java.lang.*;
import java.math.BigInteger;
import java.util.Random;
import java.io.*;

public class rsal {

    public static void main(String[] args) throws IOException
    {

        rsalab rsa=new rsalab();

        DataInputStream in=new DataInputStream(System.in);
        String teststring;
        System.out.println("Enter the plain text:");
        teststring = in.~~readLine~~();

        bts s1=new bts();
        System.out.println("Encrypting string: " +teststring);
        System.out.println("String in bytes:"+s1.bytesToString(teststring.getBytes()));

        bts s2=new bts();
        byte[] encrypted=rsa.encrypt(teststring.getBytes());

        System.out.println("Encrypted string :"+s2.bytesToString(encrypted));

        bts s3=new bts();
        byte[] decrypted=rsa.decrypt(encrypted);
        System.out.println("Decrypted string in bytes :"+s3.bytesToString(decrypted));
        System.out.println("Decrypted string :" + new String(decrypted));

    }

}


class bts
{
        public String bytesToString(byte[] encrypted)
        {
            String test="";
            for(byte b:encrypted)
            {
                test+=Byte.toString(b);
            }
```

```
                                        return test;

                                }
        }
```

Output 1–

```
Enter the plain text:
Hello World
Encrypting string: Hello World
String in bytes:721011081081113287111114108100
Encrypted string :45-2467-21-4376-10110519-45-12653101-11-9795-122111108-43-8077-1169-40-1172-85714-
5930-21-25-117-112-20-36-9110768-11395-20-5336-77-125147457-85-4748107-33-5578-87-1819-111-72-63-
705785179011914
Decrypted string in bytes :721011081081113287111114108100
Decrypted string :Hello World
```

Output 2–

```
Enter the plain text:
This is a sample
Encrypting string: This is a sample
String in bytes:841041051153210511532973211597109112108101
Encrypted string :7-38-64-487597-725231-45-87-6981-29-17-73-34127-101108-1289-126-769143-126-56-22-
21-27-7819120852868-91-81-47-105-7937-75-48-10681-6651-43-74-126-28-10468-853610941-38-58-127-126-
10910936-63347-69127
Decrypted string in bytes :841041051153210511532973211597109112108101
Decrypted string :This is a sample
```

Output 3–

```
Enter the plain text:
rsa algorithm
Encrypting string: rsa algorithm
String in bytes:114115973297108103111114105116104109
Encrypted string :3-56-1172220151939-1055135-16-4771-43127-58-2160117-3011961-46-323011771-125-
5612-175326-89480-23-102-111-94-239089983410156-12-113-128-50-9787-32-49-12033110-113-75-1611-23-
12671-86-852-62-70
Decrypted string in bytes :114115973297108103111114105116104109
Decrypted string :rsa algorithm
```

Lab Program 12:

Write a program for congestion control using leaky bucket algorithm.

Code –

```
import java.util.Scanner;

public class bucket {

        public static void main(String[] args)
        {

                Scanner sc=new Scanner(System.in);

                int bucket=0;
                int op_rate,i,n,bsize;

                System.out.println("Enter the number of packets");
                n=sc.nextInt();

                System.out.println("Enter the output rate of the bucket");
                op_rate=sc.nextInt();

                System.out.println("Enter the bucket size");
                bsize=sc.nextInt();

                System.out.println("Enter the arriving packets(size)");
                int pkt[]=new int[n];
                for(i=0;i<n;i++)
                {
                        pkt[i]=sc.nextInt();
                }

                System.out.println("\nSec\tpsize\tBucket\tAccept/Reject\tpkt_send");
                System.out.println("----------------------------------------------------");
                for(i=0;i<n;i++)
                {
                        System.out.print(i+1+"\t"+pkt[i]+"\t");
                        if(bucket+pkt[i]<=bsize)
                        {
                                bucket+=pkt[i];
                                System.out.print(bucket+"\tAccept\t\t"+min(bucket,op_rate)+"\n" +"");
                                bucket=sub(bucket,op_rate);
                        }
                        else
                        {
                                int reject=(bucket+pkt[i]-bsize);
                                bucket=bsize;
                                System.out.print(bucket+"\tReject "+reject+"\t"+min(bucket,op_rate)+"\n");

                                bucket=sub(bucket,op_rate);
```

```
					}
				}
				while(bucket!=0)
				{
					System.out.print((++i)+"\t0\t"+bucket+"\tAccept\t\t"+min(bucket,op_rate)+"\t");
					bucket=sub(bucket,op_rate);
				}
			}

			static int min(int a,int b)
			{
				return ((a<b)?a:b);
			}

			static int sub(int a,int b)
			{
				return (a-b)>0?(a-b):0;
			}
		}
```

Output 1–

```
Enter the number of packets
4
Enter the output rate of the bucket
7
Enter the bucket size
8
Enter the arriving packets(size)
6 8 9 5

Sec     psize   Bucket  Accept/Reject   pkt_send
---------------------------------------------------
1       6       6       Accept          6
2       8       8       Accept          7
3       9       8       Reject 2        7
4       5       6       Accept          6
```

Output 2–

```
Enter the number of packets
4
Enter the output rate of the bucket
6
Enter the bucket size
8
Enter the arriving packets(size)
4 5 6 10
```

```
Sec     psize   Bucket  Accept/Reject   pkt_send
----------------------------------------------------
1       4       4       Accept          4
2       5       5       Accept          5
3       6       6       Accept          6
4       10      8       Reject 2        6
5       0       2       Accept          2
```

Output 3–

```
Enter the number of packets
5
Enter the output rate of the bucket
5
Enter the bucket size
5
Enter the arriving packets(size)
4 6 3 7 5

Sec     psize   Bucket  Accept/Reject   pkt_send
----------------------------------------------------
1       4       4       Accept          4
2       6       5       Reject 1        5
3       3       3       Accept          3
4       7       5       Reject 2        5
5       5       5       Accept          5
```