Assignment-10

Name-Gourab Nanda
Roll No-120CS0195

1.

```
#==================================
#     Simulation parameters setup
#==================================
set val(stop)   90.0                ;# time of simulation end

#==================================
#        Initialization
#==================================
#Create a ns simulator
set ns [new Simulator -multicast on]

#Open the NS trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

$ns color 1 purple
$ns color 2 bisque

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

#==================================
#        Nodes Definition
#==================================
#Create 9 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]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]

#==================================
#        Links Definition
#==================================
#Createlinks between nodes
$ns duplex-link $n0 $n1 5.0Mb 2ms DropTail
$ns queue-limit $n0 $n1 50
$ns duplex-link $n1 $n2 5.0Mb 1ms DropTail
$ns queue-limit $n1 $n2 50
$ns duplex-link $n2 $n3 5.0Mb 1ms DropTail
```

```
$ns queue-limit $n2 $n3 50
$ns duplex-link $n0 $n4 5.0Mb 2ms DropTail
$ns queue-limit $n0 $n4 50
$ns duplex-link $n3 $n4 5.0Mb 2ms DropTail
$ns queue-limit $n3 $n4 50
$ns duplex-link $n3 $n5 5.0Mb 2ms DropTail
$ns queue-limit $n3 $n5 50
$ns duplex-link $n5 $n6 5.0Mb 2ms DropTail
$ns queue-limit $n5 $n6 50
$ns duplex-link $n6 $n8 5.0Mb 2ms DropTail
$ns queue-limit $n6 $n8 50
$ns duplex-link $n8 $n7 5.0Mb 2ms DropTail
$ns queue-limit $n8 $n7 50
$ns duplex-link $n7 $n5 5.0Mb 2ms DropTail
$ns queue-limit $n7 $n5 50

#Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient up
$ns duplex-link-op $n1 $n2 orient right
$ns duplex-link-op $n2 $n3 orient right-down
$ns duplex-link-op $n0 $n4 orient right-down
$ns duplex-link-op $n3 $n4 orient left-down
$ns duplex-link-op $n3 $n5 orient right
$ns duplex-link-op $n5 $n6 orient right-up
$ns duplex-link-op $n6 $n8 orient right-down
$ns duplex-link-op $n8 $n7 orient left-down
$ns duplex-link-op $n7 $n5 orient left-up

set mproto DM
set mrthandle [$ns mrtproto $mproto {}]
set group0 [Node allocaddr]
#=================================
#       Agents Definition
#=================================
set tcp [new Agent/TCP]
$tcp set class_ 1
$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $n8 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
$udp0 set dst_addr_ $group0
$udp0 set dst_port_ 0
#=================================
#       Applications Definition
#=================================
#Setup a FTP Application over TCP connection
set ftp [new Application/FTP]
```

```
$ftp attach-agent $tcp
$ns at 0.1 "$ftp start"
$ns at 5.0 "$ftp stop"

#Setup a CBR Application over UDP connection
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
$cbr0 set packetSize_ 1000
$cbr0 set rate_ 1.0Mb
$cbr0 set random_ null
$ns at 0.2 "$cbr0 start"
$ns at 4.0 "$cbr0 stop"

set rcvr [new Agent/LossMonitor]
$ns attach-agent $n6 $rcvr
$ns attach-agent $n7 $rcvr
$ns attach-agent $n8 $rcvr
$ns at 0.3 "$n6 join-group $rcvr $group0"
$ns at 0.3 "$n7 join-group $rcvr $group0"
$ns at 0.3 "$n8 join-group $rcvr $group0"
$ns at 3.8 "$n6 leave-group $rcvr $group0"
$ns at 3.8 "$n7 leave-group $rcvr $group0"
$ns at 3.8 "$n8 leave-group $rcvr $group0"

#===================================
#        Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exit 0
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

awk---

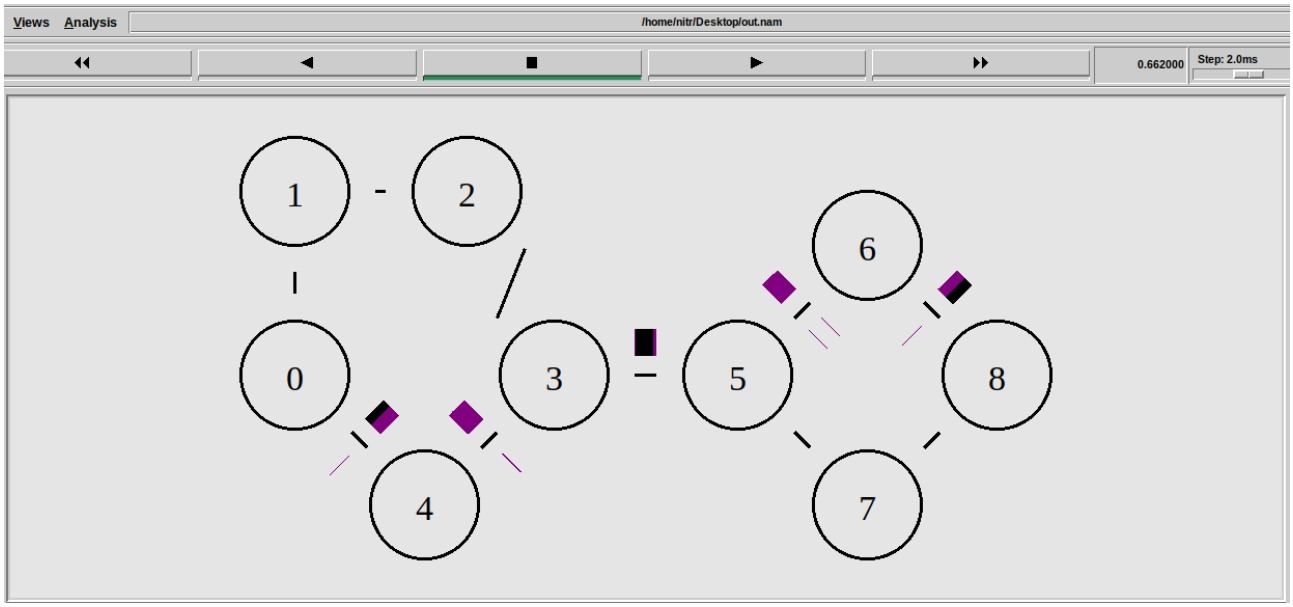
```
BEGIN{
startTime=1.0;
maxdelay=0;
maxdelay1=0;
interval=0.1;
prev=0
```

```
recv=0;
recv1=0;
}
{
event=$1;
time=$2;
from_node=$3;
to_node=$4;
pkt_type=$5;
pkt_size=$6;
flgs=$7;
f_id=$8;
src_addr=$9;
dest_addr=$10;
seq_n0=$11;
pkt_id=$12;
if (sendTime[pkt_id] == 0 && (event == "+" || event == "s") ) {
        if (time < startTime) {
                startTime = time
                }
        sendTime[pkt_id] = time
        this_flow = flow_t
        }

# Update total received packets' size and store packets arrival time
if (event == "r" && to_node == "8") {
                if(pkt_type=="tcp")
                recv1++;
                if(pkt_type=="cbr")
                recv++;
                if (time > stopTime) {
                        stopTime = time
                        }
                recvdSize += pkt_size
                recvTime[pkt_id] = time
                if(pkt_type=="cbr"&&(maxdelay<recvTime[pkt_id]-sendTime[pkt_id]))
                maxdelay=recvTime[pkt_id]-sendTime[pkt_id]
                if(pkt_type=="tcp"&&(maxdelay1<recvTime[pkt_id]-sendTime[pkt_id]))
                maxdelay1=recvTime[pkt_id]-sendTime[pkt_id]
        }

}
END{
        printf("max delay of tcp = %.2f\n",maxdelay1);
        printf("max delay of udp = %.2f\n",maxdelay);
        printf("no of  packet received at 8(UDP)= %.2f\n",recv);
        printf("no of  packet received at 8(TCP)= %.2f\n",recv1);
}
```

```
nit@nit-HP-Compaq-Elite-8300-SFF:~$ awk -f lab10q1.awk out.tr
max delay of tcp = 0.12
max delay of udp = 0.10
no of  packet received at 8(UDP)= 252.00
no of  packet received at 8(TCP)= 194.00
```



2.

TCP and UDP throughput

awk--

```
BEGIN {
recv=0;
gotime = 1;
time = 0;
time_interval=1;
}
#body
{
	event = $1
	    time = $2
	    node_id = $3
	    level = $4
	    pktType = $7
	        packet_size = $8;
```

```
 if(time>gotime) {

  print gotime, (packet_size * recv * 8.0)/1000; #packet size * ... gives results in kbps
  gotime+= time_interval;
  recv=0;
  }

#========Calculate throughput==========

if (( event == "r") && ( pktType == "tcp" ) && ( level=="AGT" ))
{
 recv++;
}

} #body


END {
;
}


BEGIN {
recv=0;
gotime = 1;
time = 0;
time_interval=1;
}
#body
{
       event = $1
            time = $2
            node_id = $3
            level = $4
            pktType = $7
                packet_size = $8

 if(time>gotime) {

  print gotime, (packet_size * recv * 8.0)/1000; #packet size * ... gives results in kbps
  gotime+= time_interval;
  recv=0;
  }

#========Calculate throughput==========

if (( event == "r") && ( pktType == "cbr" ) && ( level=="AGT" ))
{
 recv++;
}

} #body
```

```
END {
;
}
```