Graded Lab 1 WSN

Submitted by: Rohan Verma 1510110508 (Note: On my computer, $EXP is named as $CBR2 and starts at correct time.)

Part A (Determine the number of packets dropped. Plot throughput vs time)

Scripts ns2(gr.tcl)

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>

#===================================
#     Simulation parameters setup
#===================================
set val(stop)   5.0                         ;# time of simulation end

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

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

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

#===================================
#        Nodes Definition
#===================================
#Create 5 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

#===================================
# Map colors to ints
#===================================

$ns color 10 blue
$ns color 11 red
$ns color 12 green

#===================================
#        Links Definition
#===================================
```

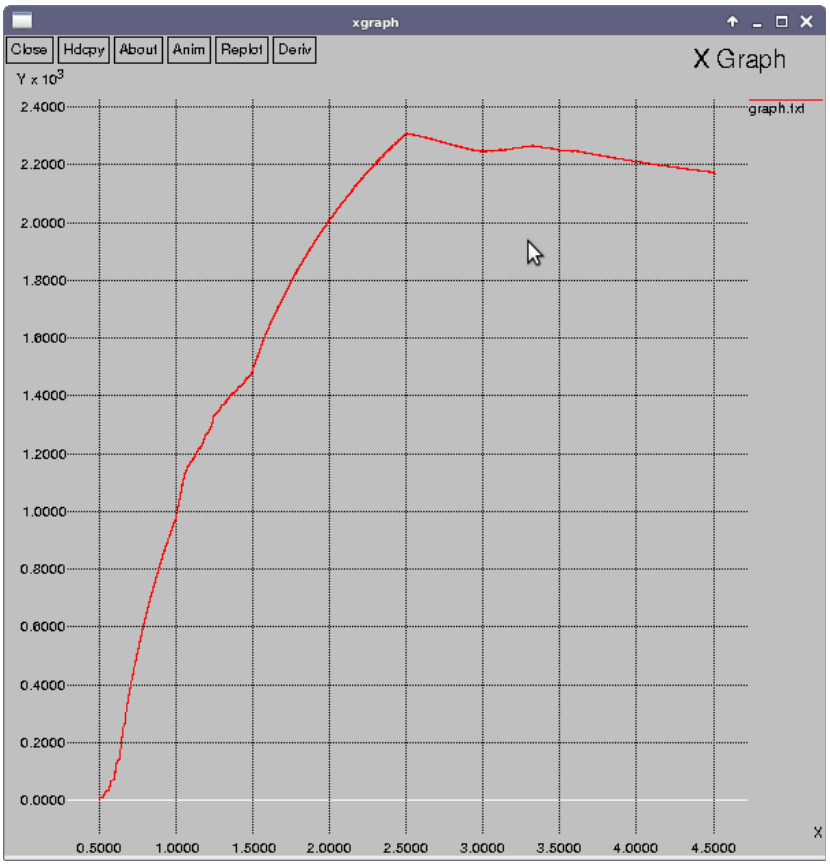
```
#Createlinks between nodes
$ns duplex-link $n0 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n0 $n3 50
$ns duplex-link $n1 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n1 $n3 50
$ns duplex-link $n2 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n2 $n3 50
$ns duplex-link $n3 $n4 0.6Mb 10ms DropTail
$ns queue-limit $n3 $n4 10

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

#===================================
#        Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink3 [new Agent/TCPSink]
$ns attach-agent $n4 $sink3
$ns connect $tcp0 $sink3
$tcp0 set packetSize_ 150
$tcp0 set class_ 10

#Setup a UDP connection
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set null5 [new Agent/Null]
$ns attach-agent $n4 $null5
$ns connect $udp1 $null5
$udp1 set packetSize_ 200
$udp1 set class_ 11

#Setup a UDP connection
set udp2 [new Agent/UDP]
$ns attach-agent $n2 $udp2
set null4 [new Agent/Null]
$ns attach-agent $n4 $null4
$ns connect $udp2 $null4
$udp2 set packetSize_ 1500
$udp2 set class_ 12


#===================================
#        Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 0.5 "$ftp0 start"
$ns at 3.5 "$ftp0 stop"

#Setup a CBR Application over UDP connection
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
```

```
$cbr1 set packetSize_ 200
$cbr1 set rate_ 0.5Mb

$ns at 1.0 "$cbr1 start"
$ns at 2.5 "$cbr1 stop"

#Setup a CBR Application over UDP connection
set cbr2 [new Application/Traffic/CBR]
$cbr2 attach-agent $udp2
$cbr2 set packetSize_ 200
$cbr2 set burst_time_ 2s
$cbr2 set idle_time_ 1s
$cbr2 set rate_ 0.5Mb

$ns at 1.5 "$cbr2 start"
$ns at 4.5 "$cbr2 stop"


#===================================
#        Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exec python analysis.py
    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
```

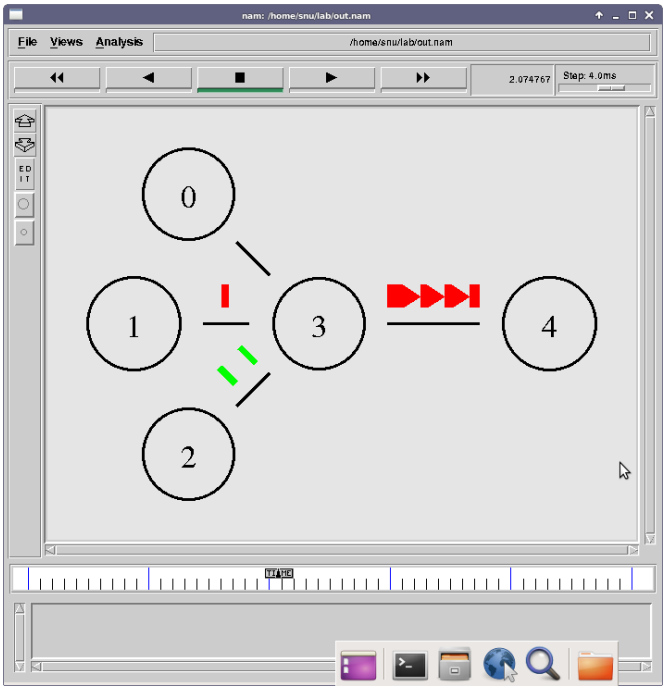
Script python (analysis.py)

```
count = 0
packets = 0
g = open('graph.txt', 'w')
d = open('drops.txt', 'w')
with open('out.tr') as f:
    lines = f.readlines()
for line in lines:
    k = line.split()
    if line[0] == 'd':
        count+=1
    if k[4] != 'ack':
        packets += 1

    g.write(str(k[1]) + " " +  str(packets*1.0/float(k[1])) + "\n")

g.close()
d.write("# of packets dropped:" + str(count))
d.close()
```

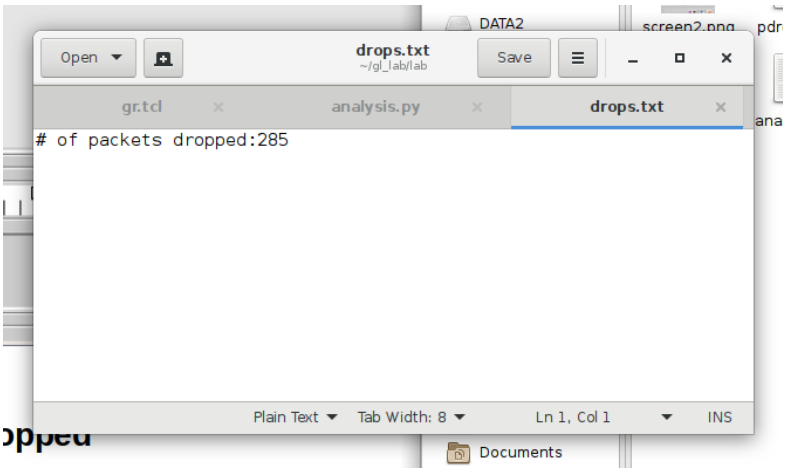
Xgraph output



Nam output



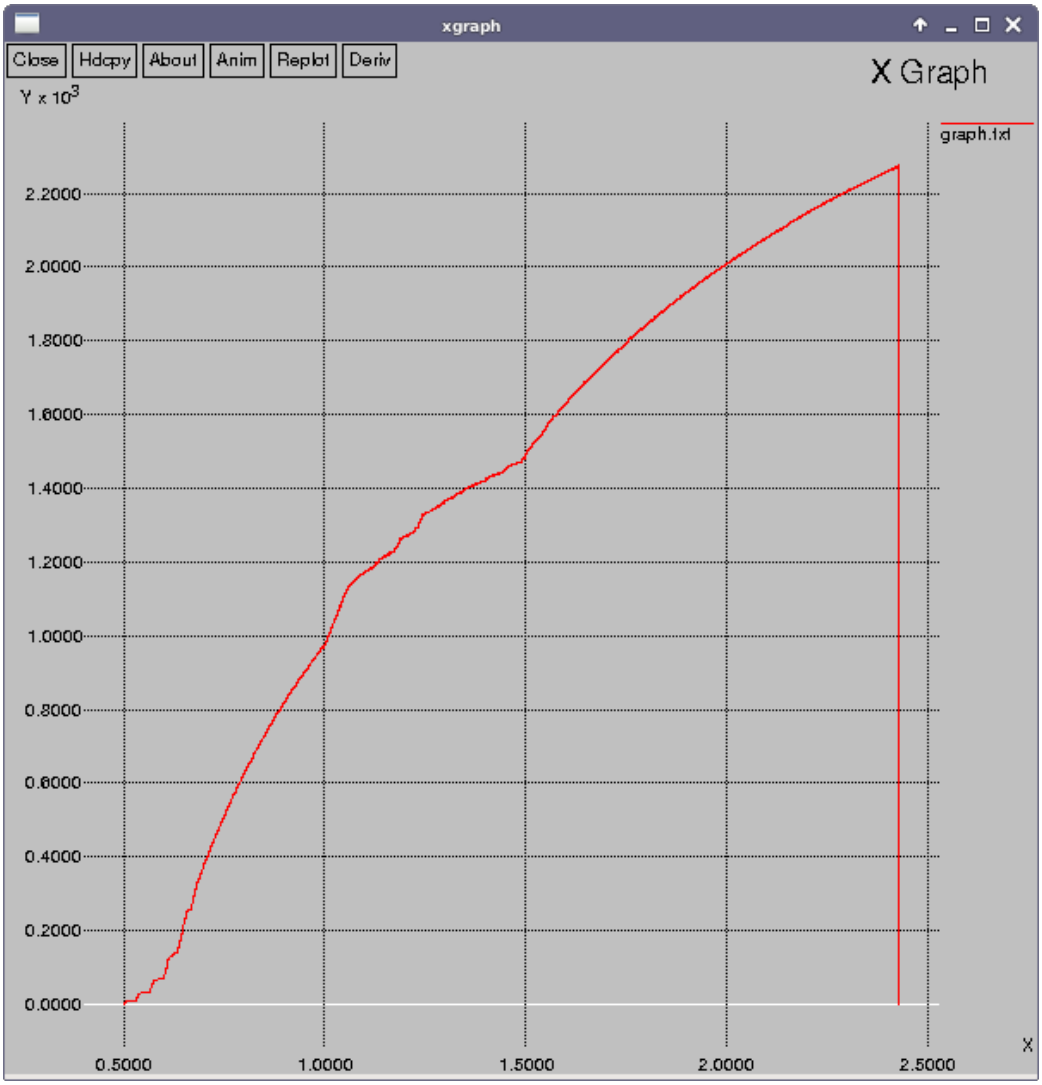
Number of packet dropped



PART B (Introduce an error model (packet error rate of 10%) at backbone link (N3-N4) and plot throughput vs time.)

```
# Only modified part of gr.tcl
#=================================
# Error Model Definition
#=================================
set em [new ErrorModel]
$em unit pkt
$em set rate_ 0.1
$em ranvar [new RandomVariable/Uniform]
$em drop-target [new Agent/Null]

#=================================
#        Links Definition
#=================================
#Createlinks between nodes
$ns duplex-link $n0 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n0 $n3 50
$ns duplex-link $n1 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n1 $n3 50
$ns duplex-link $n2 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n2 $n3 50
$ns duplex-link $n3 $n4 0.6Mb 10ms DropTail
$ns queue-limit $n3 $n4 10

#Attach Lossmodel to link
$ns link-lossmodel $em $n3 $n4

#Give node position (for NAM)
$ns duplex-link-op $n0 $n3 orient right-down
$ns duplex-link-op $n1 $n3 orient right
```

```
$ns duplex-link-op $n2 $n3 orient right-up
$ns duplex-link-op $n3 $n4 orient right

#===================================
#        Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam out.nam &
    exec python analysis.py &
    exec xgraph graph.txt &
    exit 0
}
```

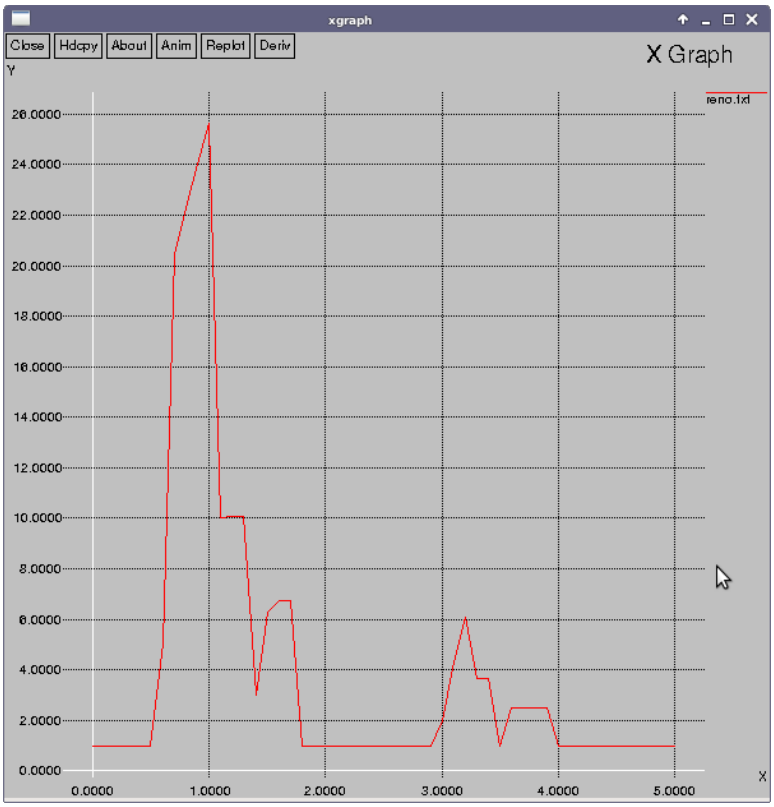


PART C (Plot congestion window for N0 (source) and N4 (destination)- TCP Reno and TCP Tahoe. Compare both graphs)

```
#=================================
# Obtain CWND from TCP agent
#=================================

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

$ns at 0.0 "plotWindow $tcp0 $cwndgraph"
```

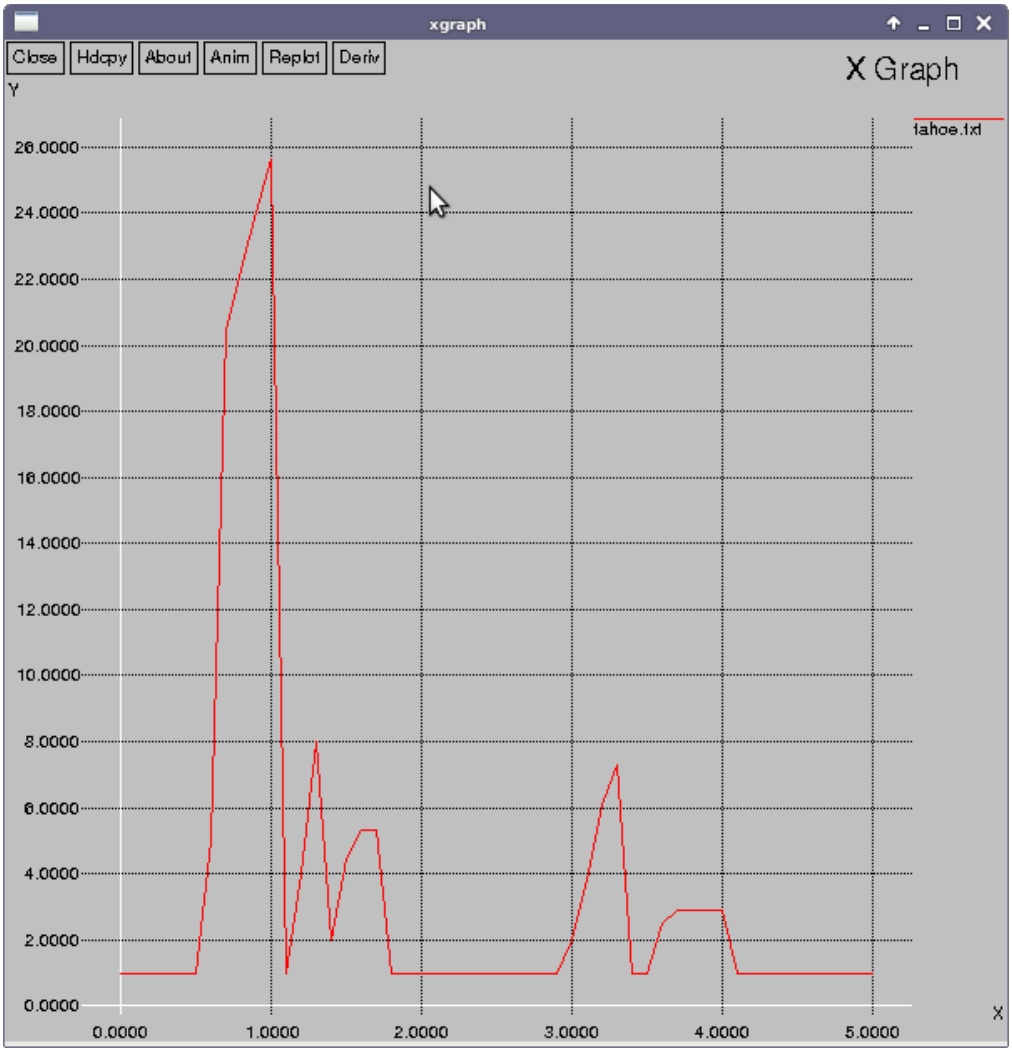
For checking TCP Reno:

```
set cwndgraph [open  "reno.txt"  w]

#Setup a TCP connection
set tcp0 [new Agent/TCP/Reno]
$ns attach-agent $n0 $tcp0
```



For checking TCP Tahoe:

```
set cwndgraph [open  "tahoe.txt"  w]
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
```



PART D (Add N5, N6 and N7 to the above network and simulate an Ethernet LAN with N4, N5, N6 and N7 using CSMA/CD as MAC protocol. Take bandwidth = 0.5 Mbps, delay = 40 ms.)

```
#===================================
#        Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n0 $n3 50
$ns duplex-link $n1 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n1 $n3 50
$ns duplex-link $n2 $n3 2.0Mb 5ms DropTail
$ns queue-limit $n2 $n3 50
$ns duplex-link $n3 $n4 0.6Mb 10ms DropTail
$ns queue-limit $n3 $n4 10

#Attach Lossmodel to link
$ns link-lossmodel $em $n3 $n4

# Make LAN
$ns newLan "$n4 $n5 $n6 $n7" 0.5Mb 40ms LL Queue/DropTail MAC/Csma/Cd

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