LỜI MỞ ĐẦU

Mạng viễn thông với tài nguyên băng thông khan hiếm khi nhiều luồng dữ liệu cùng truy cập sẽ dẫn đến tình trạng tắc nghẽn nếu không có sự phân chia công bằng về mặt băng thông cho nhiều người cùng sử dụng.

Nhóm em chọn làm bài tập lớn với đề tài **"băng thông công bằng giữa các luồng"** trong hệ thống mạng thông tin.

CHƯƠNG I: ĐỀ TÀI THỰC HIỆN

BTL của nhóm yêu cầu tính tốc độ các luồng dữ liệu gửi qua mạng để các luồng chia sẽ băng thông kênh truyền dựa theo nguyên lý công bằng cực đại cực tiểu (max-min fairness) và dựng kịch bản mô phỏng bằng công cụ NS2

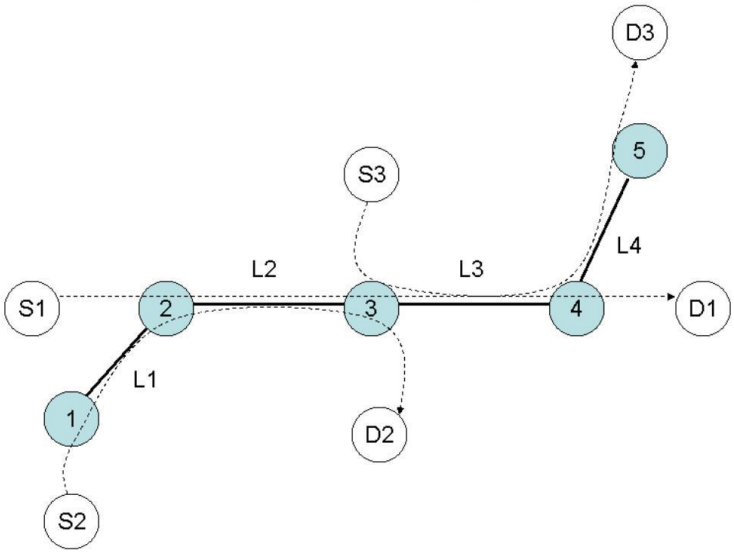
Các thành viên trong nhóm và nhiệm vụ của từng thành viên

- Trịnh Ngọc Cường: Tìm hiểu code và viết báo cáo.
- Nghiêm Lê Hoa: Tìm hiểu thuật toán Tính băng thông công bằng giữa các luồng viết code làm slide.
- Hoàng Trọng Minh: Viết báo cáo, mô phỏng code.

Thực hiện:

1. Yêu cầu:

Bài 4: Băng thông công bằng giữa các luồng



Cho một mạng gồm 5 nút như hình vẽ. Nút 1, 2, 3, 4, 5 là các hàng đợi đơn hoạt động theo nguyên tắc FIFO với độ lớn hàng đợi *K*=10 gói. Có 3 luồng dữ liệu được gửi qua mạng tương ứng là (*S1, D1*), (*S2, D2*) và (*S3, D3*). Trong đó *Si* là nguồn phát dữ liệu còn *Di* là đích.

Đường nối *L1* có dung lượng 1,5Mb/s trễ lan truyền 150ms; *L2* có dung lượng là 1Mb/s trễ lan truyền 100ms; đường *L3* có dung lượng 0,6Mb/s, trễ lan truyền 50ms; và L4 có dung lượng 0,5Mb/s, trễ lan truyền 100ms. Các nguồn *Si* đều phát gói với độ dài cố định là 125byte, khoảng thời gian giữa các gói tuân theo phân bố Poisson.

4.1. Giả thiết băng thông đối đa tổng cộng mà các luồng được chiếm trên một kênh truyền vật lý là bằng 95% dung lượng kênh truyền. Tính tốc độ λ_{S1}, λ_{S2}, λ_{S3} (kbit/s) để 3 luồng trên chia sẻ băng thông kênh truyền theo nguyên lý công bằng cực đại - cực tiểu (max – min fairness).

4.2. Dựng kịch bản mô phỏng mạng trên với tốc độ các luồng λ_{S1}, λ_{S2}, λ_{S3} đã được tính toán trong 4.1. Chạy mô phỏng trong 100s.

4.3. Vẽ đồ thị băng thông $r_i(t)$ mà các luồng (*S1, D1*), (*S2, D2*) và (*S3, D3*) sử dụng. Vẽ đồ thị tốc độ mất gói $e_i(t)$ (số gói mất/đơn vị thời gian) của 3 luồng (*S1, D1*), (*S2, D2*) và (*S3, D3*) tại nút 3.

4.4. Thay các luồng theo phân bố Poisson như trên bằng các luồng TCP. Lặp lại câu 4.3. Có nhận xét gì?

2. *Kết quả*

4.1: Tính tốc độ phát gói

Theo đề bài, ta có các tham số:

• Đường nối *L1* có dung lượng là C1= 1.5Mb/s trễ lan truyền 150ms •

Đường nối *L2* có dung lượng l à C 2 = 1Mb/s, trễ lan truyền 1 0 0ms

- Đường nối L3 có dung lượng là C3 = 0.6Mb/s, trễ lan truyền 50ms
- Đường nối L4 có dung lượng là C4 = 0.5Mb/s, trễ lan truyền 100ms

Nút 1, 2, 3, 4, 5 là các hàng đợi đơn hoạt động theo nguyên tắc FIFO với độ lớn hàng đợi *K*=10 gói.

Các nguồn Si phát gói với độ dài cố định 125byte, tuân theo phân bố Poisson.

Băng thông tối đa tổng cộng mà các luồng được chiếm trên một kênh truyền vật lý là bằng 95% dung lượng kênh truyền

$\lambda_1, \lambda_2, \lambda_3 = ?$

<u>Tính toán:</u>

- Các đường nối $_1 = (1, 2)$, $_2 = (2, 3)$, $_3 = (3, 4)$, $_4 = (4, 5)$

- Các luồng (S1, D1), (S2, D2), (S3, D3)

Luồng	(S1, D1)	(S2, D2)	(S3, D3)	Giải thích
Bước 1:	0	0	0	Khởi tạo, $=$ $\min_{\in(1,2,3,4)} \left(\frac{1.5-0}{1}, \frac{1-0}{2}, \frac{0.6-0}{2}, \frac{0.5-0}{1} \right) =$ $\min_{\in(1,2,3,4)} (1.5, 0.5, 0.3, 0.5) = 0.3$
Bước 2:	<u>0.3</u>	0.3	<u>0.3</u>	3 bão hòa (0.3 + 0.3 = 0.6 = $_3$) loại bỏ (S1, D1), (S3, D3), 3
Bước 3:	λ_1 $= 0.3$		λ_3 $= 0.3$	$\min_{\in(1,2)} \left(\frac{1.5-0.3}{1}, \frac{1-0.6}{1} \right)$ $= \min_{\in(1,2,3,4)} (1.2, 0.4)$ $= 0.4$
Bước 4:		<u>0.7</u>		2 bão hòa (0.3 + 0.7 = 1.0 = $_2$) loại bỏ (S2, D2), 2
Bước 5:		λ_2 $= 0.7$		Kết thúc thuật toán

Mặt khác, theo giả thiết băng thông tối đa tổng cộng mà các luồng được chiếm trên một kênh truyền vật lý là bằng 95% dung lượng kênh truyền nên tốc độ phát gói của các nguồn:

- $\lambda_1 = 0.3 * 0.95 = 0.285$ (Mb/s) = 285 (kbit/s)
- $\lambda_2 = 0.7 * 0.95 = 0.665$ (Mb/s) = 665 (kbit/s)
- $\lambda_3 = 0.3 * 0.95 = 0.285$ (Mb/s) = 285 (kbit/s)

⇨ Như vậy ta đã tính được các tham số λ_1, λ_2, λ_3 theo nguyên lý max-min fairness.

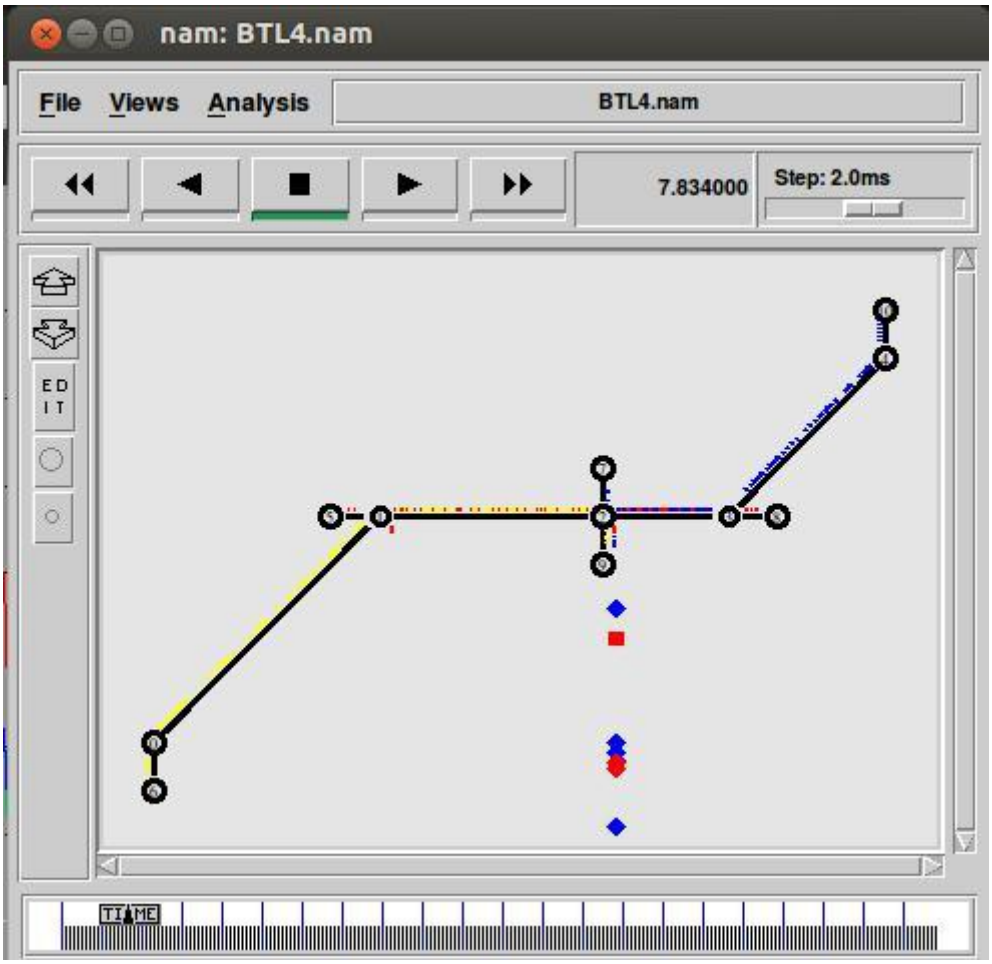
4.2: Dựng kịch bản mô phỏng trong 100s

Ta tính toán được các tốc độ phát gói như ở trên:

tốc độ đến trung bình gói/s

set lambda1 285.0
set lambda2 665.0
set lambda3 285.0

Kích thước gói 125 gói/s
set pksize 125.0



Hình 4.1: Kịch bản mô phỏng

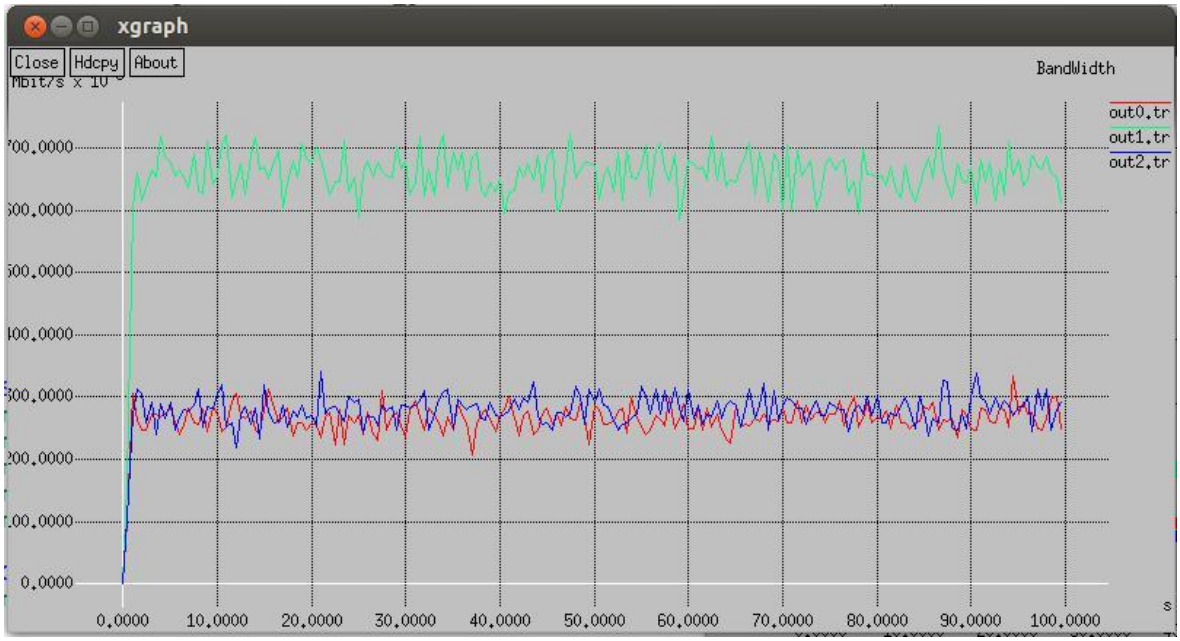
Các nút 0,1,2,3,4: là các nút n1, n2, n3, n4, n5

Các nút 5, 6, 7: là các nguồn 1, 2, 3 tương ứng

Các nút 8, 9, 10: là các đích 1, 2, 3 tương ứng.

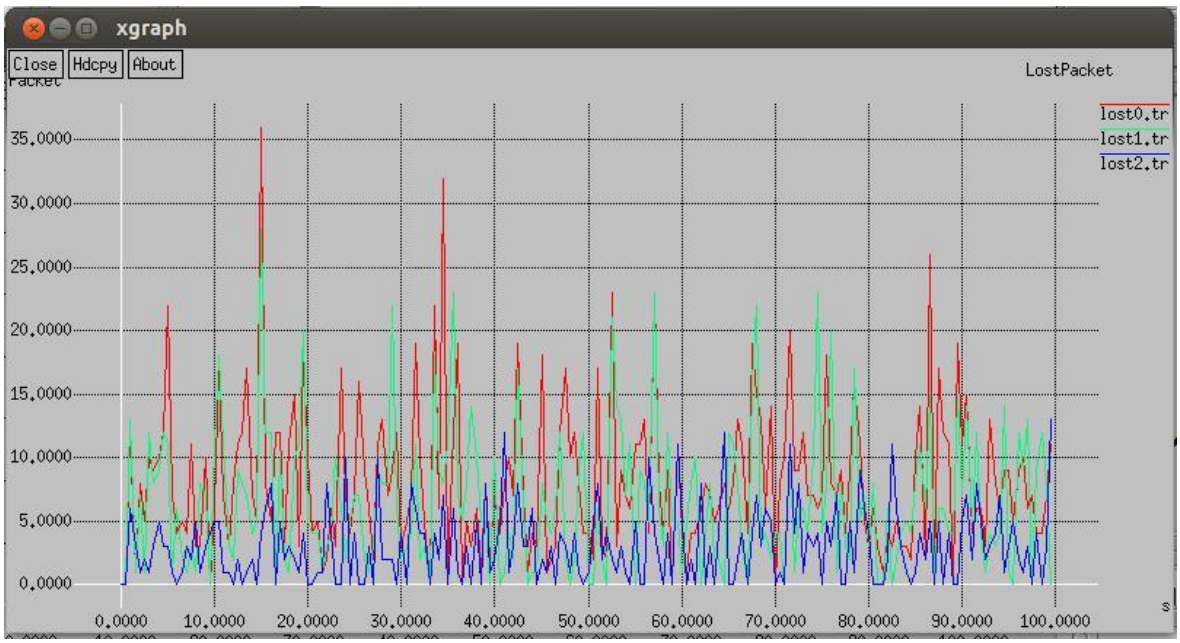
4.3: Vẽ đồ thị

- đồ thi băng thông của các luồng (S1, D1), (S2, D2), S3,D3)



Hình 4.2: đồ thị băng thông

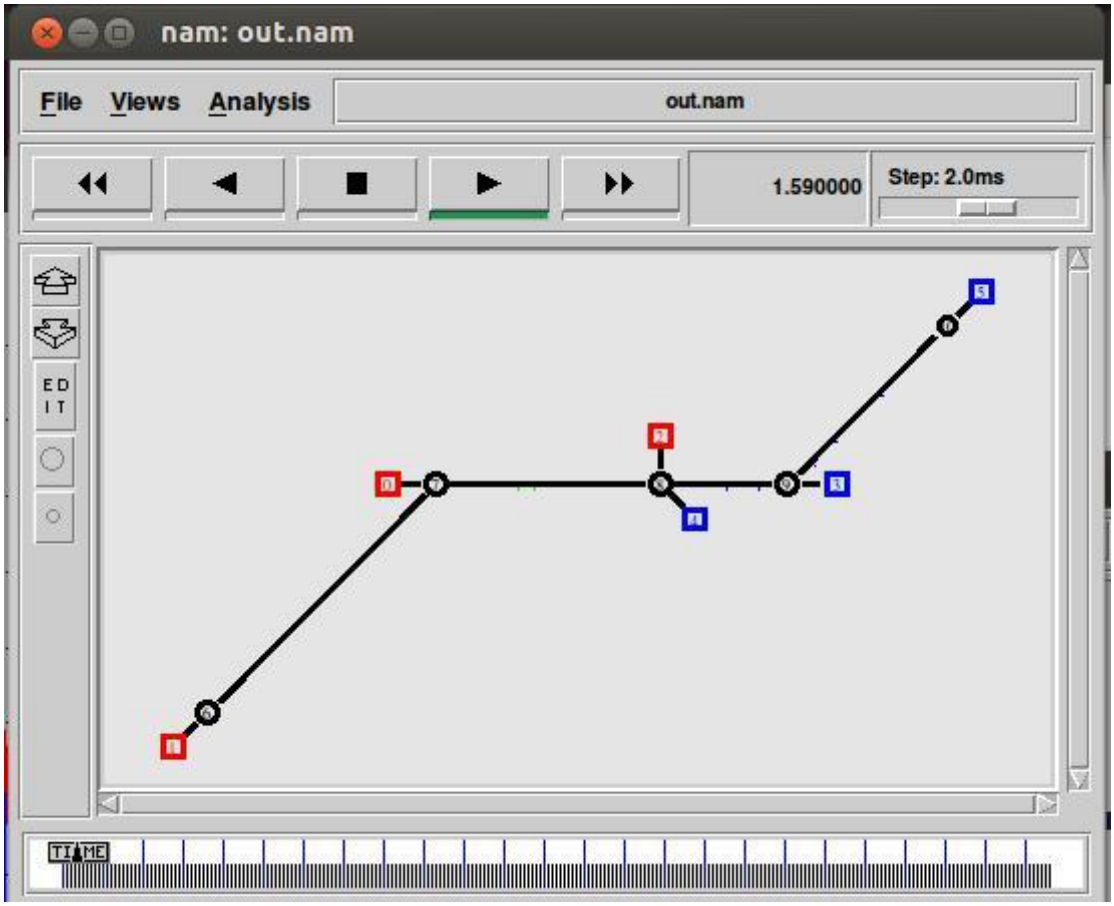
- đồ thị tốc độ mất gói của 3 luồng



Hình 4.3: đồ thị tốc độ mất gói

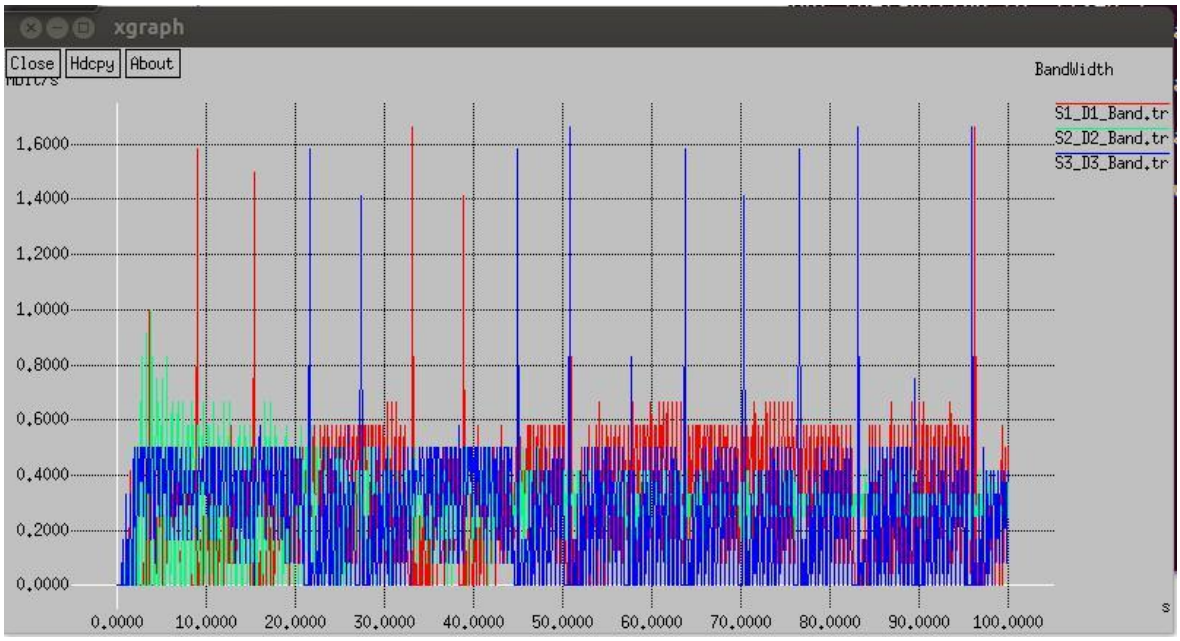
4.4: Thay nguồn trên bằng nguồn TCP

- Kịch bản mô phỏng



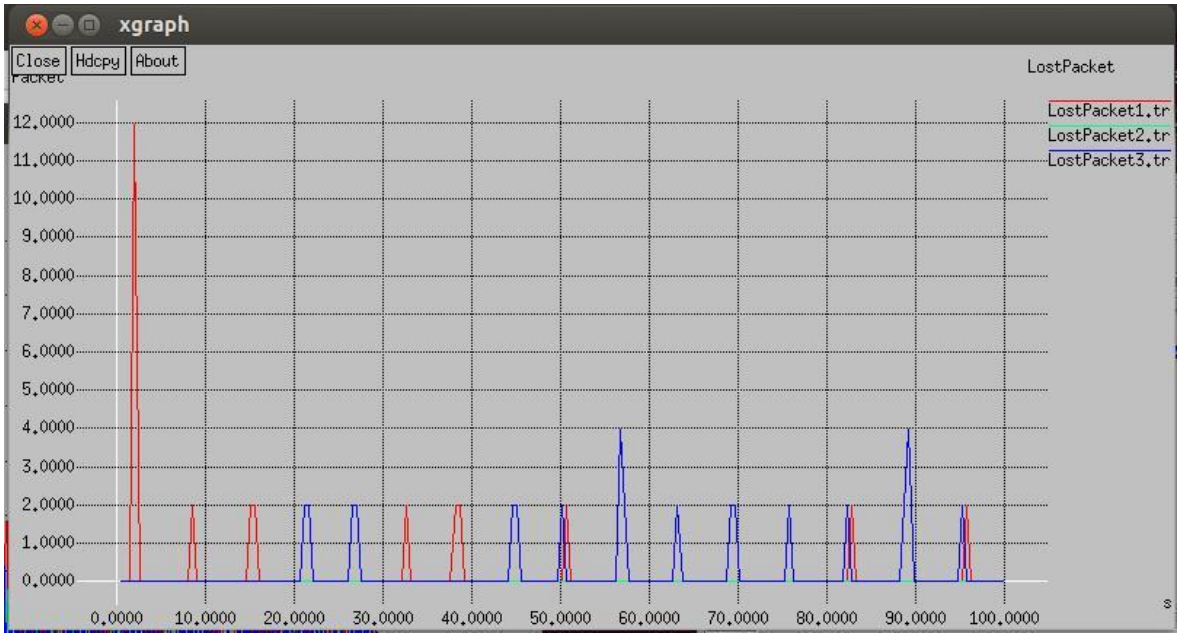
Hình 4.4: Kịch bản mô phỏng

- đồ thị băng thông



Hình 4.5: đồ thị băng thông

- **Tốc độ mất gói**



Hình 4.6: đồ thị mất gói

CHƯƠNG 2: KẾT LUẬN

Sau 3 lần chạy mô phỏng ta thu được kết quả của 3 lần đều giống nhau:

- Băng thông của 3 luồng khi mô phỏng hoàn toàn phù hợp với kết quả tính toán trên lý thuyết.
- Tốc độ mất gói trung bình của luồng (S3,D3) là thấp nhất rồi đến luồng 1 (S1,D1), luồng 2 (S2, D2)
- Giao thức TCP cho phép truyền dữ liệu một cách hiệu quả hơn UDP, ít mất gói hơn UDP do cơ chế có thông báo lỗi khi truyền.
-

TÀI LIỆU THAM KHẢO

- http://www.isi.edu/nsnam/ns/tutorial/index.html.
- http://nile.wpi.edu/NS/.
- http://www-sop.inria.fr/maestro/personnel/Eitan.Altman/COURS-NS/n3.pdf.
- http://www.isi.edu/nsnam/ns/tutorial/nsscript4.html.
- http://www.isi.edu/nsnam/ns/ns-build.html.

- http://www.isi.edu/nsnam/ns/ns-problems.html.
- http://www.svbkol.org/forum/showthread.php?t=11106.

PHỤ LỤC

Code 4.3:

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

#Define different colors for data flows
$ns color 1 Red
$ns color 2 Yellow
$ns color 3 Blue

#Set lambda value (packet/s)
set lambda1 285.0
set lambda2 665.0
set lambda3 285.0

#Set packet size
set pksize 125.0

#Time to send packet
set ArrivalTime1 [new RandomVariable/Exponential]
$ArrivalTime1 set avg_ [expr 1/$lambda1]
set ArrivalTime2 [new RandomVariable/Exponential]
$ArrivalTime2 set avg_ [expr 1/$lambda2]
set ArrivalTime3 [new RandomVariable/Exponential]
$ArrivalTime3 set avg_ [expr 1/$lambda3]

#Open the Trace file
set f0 [open out0.tr w]
set f1 [open out1.tr w]
set f2 [open out2.tr w]

set l0 [open lost0.tr w]
set l1 [open lost1.tr w]
set l2 [open lost2.tr w]

#Open the nam trace file
set nf [open BTL4.nam w]
$ns namtrace-all $nf

#Dinh nghia 1 thu tuc 'finish'
proc finish {} {
        global ns f0 f1 f2 nf
        $ns flush-trace
        #Close the output files
        close $f0
        close $f1
```

```
        close $f2
        close $nf
        #Execute nam on the trace file
        exec nam BTL4.nam &
        #Call xgraph to display the results

        exec xgraph out0.tr out1.tr out2.tr -geometry 800x400 -t "BandWidth" -x "s" -y "Mbit/s" &

        exec xgraph lost0.tr lost1.tr lost2.tr -geometry 800x400 -t "LostPacket" -x "s" -y "Packet" &

        exit 0

}

#Create 5 nodes
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

#Create 3 soures
set s1 [$ns node]
set s2 [$ns node]
set s3 [$ns node]

#Create 3 destinations
set d1 [$ns node]
set d2 [$ns node]
set d3 [$ns node]

#Create links between the nodes
$ns duplex-link $n1 $n2 1.5Mb 150ms DropTail
$ns duplex-link $n2 $n3 1Mb 100ms DropTail
$ns duplex-link $n3 $n4 0.6Mb 50ms DropTail
$ns duplex-link $n4 $n5 0.5Mb 100ms DropTail
$ns duplex-link $s2 $n1 1Mb 10ms DropTail
$ns duplex-link $s1 $n2 1Mb 10ms DropTail
$ns duplex-link $d2 $n3 1Mb 10ms DropTail
$ns duplex-link $s3 $n3 1Mb 10ms DropTail
$ns duplex-link $d1 $n4 1Mb 10ms DropTail
$ns duplex-link $d3 $n5 1Mb 10ms DropTail

#Set position of nodes
$ns duplex-link-op $s2 $n1 orient up
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $s1 $n2 orient right
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $s3 $n3 orient down
$ns duplex-link-op $d2 $n3 orient up
$ns duplex-link-op $n4 $n3 orient left
$ns duplex-link-op $d1 $n4 orient left
$ns duplex-link-op $n4 $n5 orient right-up
```

```
$ns duplex-link-op $d3 $n5 orient down

#Set position of queues

$ns duplex-link-op $n2 $n1 queuePos 1.5
$ns duplex-link-op $n3 $n2 queuePos 1.5
$ns duplex-link-op $n4 $n3 queuePos 1.5
$ns duplex-link-op $n5 $n4 queuePos 1.5

#Set queue size
$ns queue-limit $n1 $n2 10
$ns queue-limit $n2 $n3 10
$ns queue-limit $n3 $n4 10
$ns queue-limit $n4 $n5 10

#Create a UDP agent and attach it to node s1
set udp0 [new Agent/UDP]
$udp0 set class_ 1
$ns attach-agent $s1 $udp0

#Create a UDP agent and attach it to node s2
set udp1 [new Agent/UDP]
$udp1 set class_ 2
$ns attach-agent $s2 $udp1

#Create a UDP agent and attach it to node s3
set udp2 [new Agent/UDP]
$udp2 set class_ 3
$ns attach-agent $s3 $udp2

#Create a Sink agent (a traffic sink) and attach it to node d1, d2, d3
set sink0 [new Agent/LossMonitor]
$ns attach-agent $d1 $sink0

set sink1 [new Agent/LossMonitor]
$ns attach-agent $d2 $sink1

set sink2 [new Agent/LossMonitor]
$ns attach-agent $d3 $sink2

#Connect the traffic sources with the traffic sink
$ns connect $udp0 $sink0
$ns connect $udp1 $sink1
$ns connect $udp2 $sink2

#Send packet
proc sendpacket0 {} {
        global ns udp0 ArrivalTime1 pksize
        set time [$ns now]
        $ns at [expr $time + [$ArrivalTime1 value]] "sendpacket0"
        $udp0 send $pksize
}
```

```
proc sendpacket1 {} {
        global ns udp1 ArrivalTime2 pksize
        set time [$ns now]
        $ns at [expr $time + [$ArrivalTime2 value]] "sendpacket1"
        $udp1 send $pksize
}

proc sendpacket2 {} {
        global ns udp2 ArrivalTime3 pksize
        set time [$ns now]
        $ns at [expr $time + [$ArrivalTime3 value]] "sendpacket2"
        $udp2 send $pksize
}

proc recordbw {} {
        global sink0 sink1 sink2 f0 f1 f2
        #Get an instance of the simulator
        set ns [Simulator instance]
        #Set the time after which the procedure should be called again
  set time 0.5
        #How many bytes have been received by the traffic sinks?
  set bw0 [$sink0 set bytes_]
  set bw1 [$sink1 set bytes_]
  set bw2 [$sink2 set bytes_]
        #Get the current time
  set now [$ns now]
        #Calculate the bandwidth (in MBit/s) and write it to the files
  puts $f0 "$now [expr $bw0/$time*8/1000000]"
  puts $f1 "$now [expr $bw1/$time*8/1000000]"
  puts $f2 "$now [expr $bw2/$time*8/1000000]"
        #Reset the bytes_ values on the traffic sinks
  $sink0 set bytes_ 0
  $sink1 set bytes_ 0
  $sink2 set bytes_ 0
        #Re-schedule the procedure
  $ns at [expr $now+$time] "recordbw"
}

proc recordlost {} {
        global sink0 sink1 sink2 l0 l1 l2
        #Get an instance of the simulator
        set ns [Simulator instance]
        #Set the time after which the procedure should be called again
  set time 0.5
        #How many packet have been lost?
  set lost0 [$sink0 set nlost_]
  set lost1 [$sink1 set nlost_]
  set lost2 [$sink2 set nlost_]
        #Get the current time
  set now [$ns now]
        #Calculate number of packet lost
  puts $l0 "$now [expr $lost0]"
  puts $l1 "$now [expr $lost1]"
```

```
    puts $l2 "$now [expr $lost2]"
                #Reset the nlost_ values on the traffic sinks
    $sink0 set nlost_ 0
    $sink1 set nlost_ 0
    $sink2 set nlost_ 0
                #Re-schedule the procedure
    $ns at [expr $now+$time] "recordlost"
}

#Schedule events for the CBR agents
$ns at 0.0 "recordbw"
$ns at 0.0 "recordlost"
$ns at 0.5 "sendpacket0"
$ns at 0.5 "sendpacket1"
$ns at 0.5 "sendpacket2"

#Call the finish procedure
$ns at 100 "finish"

#Run the simulation
$ns run
```

Code 4.4:

Thay nguồn UDP bằng Nguồn TCP

```
#tao doi tuong mo phong
set ns [new Simulator]

#xac dinh cac loai mau khac nhau cho cac duong du lieu (for NAM)
$ns color 1 red
$ns color 2 green
$ns color 3 blue

#cac bien xac dinh toc do phat goi cua cac duong lien ket (goi/s)
set lambda1 285.0
set lambda2 665.0
set pksize  125.0



#mo cac trace files S?_D?_Band.tr luu du lieu de ve do thi bang thong va S?_D?_Lost cho do thi toc do mat goi (for XGRAPH)
set f1 [open S1_D1_Band.tr w]
set f2 [open S2_D2_Band.tr w]
set f3 [open S3_D3_Band.tr w]

set l1 [open LostPacket1.tr w]
set l2 [open LostPacket2.tr w]
set l3 [open LostPacket3.tr w]

set tf [open btl4.tr w]
$ns trace-all $tf
#tao trace file la cac file chua du lieu dau ra cua mo phong dung lenh open
#set tracefile1 [open out.tr w]
```

```
#tracefile1 la 1 con tro tro den file du lieu dau ra duoc goi "out.tr"
#mo file "out.tr" su dung cho viec viet (writing file) - w
#$ns trace-all $tracefile1
#trace-all la 1 phuong thuc mo phong (trace tat ca cac su kien (events) theo 1 dang thuc dinh truoc
#lenh trace-file voi thong so la ten cua file ma chung ta can theo doi (trace)

#tao NAM trace file cung y nghia nhu tren nhung voi muc dich hinh anh hien truoc mat de hinh dung
set namfile [open out.nam w]
#namfile la 1 con tro tro den file du lieu dau ra (cho NAM) duoc goi "out.nam"
$ns namtrace-all $namfile
#cau lenh noi rang "doi tuong mo phong thu lai toan bo tien trinh theo doi mo phong theo dinh dang dau vao NAM", no se lay te n ma su thoe doi
(trace) duoc viet vao sau do boi lenh "$ns flush-trace" (xem thu tuc 'finish' duoi day)

#xac dinh thu tuc 'finish'
proc finish {} {
# thu tuc finish k co doi so dau vao
   global ns namfile f1 f2 f3 tf l1 l2 l3
#global noi rang chung ta su dung cac bien duoc khai bao ben ngoai thu tuc va sau khi thu tuc ket thuc, gia tri cua cac bien nay se thay doi khi ra
ngoai
   $ns flush-trace
 #dong cac file dau ra
   close $f1
   close $f2
   close $f3
   close $tf
            close $l1
            close $l2
            close $l3
#phuong thuc mo phong "flush-trace" se xuat cac theo doi ra file tuong ung
exec nam out.nam &
close $namfile
exec awk -f s1_d1.awk btl4.tr
   #close $tracefile1

 #thuc thi XGRAPH de hien thi ket qua
   exec xgraph S1_D1_Band.tr S2_D2_Band.tr S3_D3_Band.tr -geometry 800x400 -t "BandWidth" -x "s" -y "Mbit/s" & exec
   xgraph LostPacket1.tr LostPacket2.tr LostPacket3.tr -geometry 800x400 -t "LostPacket" -x "s" -x "s" -y "Packet" &


#ham close dong cac file trace duoc xac dinh luc truoc

#ham exec thuc hien chuong trinh NAM cho viec quan sat, o day dung ten that cua file, chu KO dung pointer "namfile" cua no vi la ham thuc thi nen
phai thuc thi noi dung cua pointer, chu k phai pointer
   exit 0
#ham exit ket thuc application va tra lai so 0 la trang thai cua he thong, Zero mac dinh la clean exit (thoat va xoa)
}

#xac dinh 1 mang cac link (lien ket) va cac node (nut)
#cach xac dinh node
#tao 3 nut nguon
set s(1) [$ns node]
set s(2) [$ns node]
set s(3) [$ns node]
```

```
$s(1) shape "square"
$s(1) color "red"
$s(2) shape "square"
$s(2) color "red"
$s(3) shape "square"
$s(3) color "red"

#tao 3 nut dich
set d(1) [$ns node]
set d(2) [$ns node]
set d(3) [$ns node]

$d(1) shape "square"
$d(1) color "blue"
$d(2) shape "square"
$d(2) color "blue"
$d(3) shape "square"
$d(3) color "blue"

#tao 5 nut trung gian
set n(1) [$ns node]
set n(2) [$ns node]
set n(3) [$ns node]
set n(4) [$ns node]
set n(5) [$ns node]

#tao lien ket giua cac nut voi bang thong (Mbit/s) va tre truyen dan (ms)
$ns duplex-link $s(1) $n(2) 1Mb 10ms DropTail
$ns duplex-link $n(2) $n(3) 1Mb 100ms DropTail
$ns duplex-link $n(3) $n(4) 0.6Mb 50ms DropTail
$ns duplex-link $n(4) $d(1) 1Mb 10ms DropTail
$ns duplex-link $s(2) $n(1) 1Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 1.5Mb 150ms DropTail
$ns duplex-link $n(3) $d(2) 1Mb 10ms DropTail
$ns duplex-link $s(3) $n(3) 1Mb 10ms DropTail
$ns duplex-link $n(4) $n(5) 0.5Mb 100ms DropTail
$ns duplex-link $n(5) $d(3) 1Mb 10ms DropTail

#thiet lap vi tri cac nut tren (for NAM)
$ns duplex-link-op $s(1) $n(2) orient right
$ns duplex-link-op $n(2) $n(3) orient right
$ns duplex-link-op $n(3) $n(4) orient right
$ns duplex-link-op $n(4) $d(1) orient right
$ns duplex-link-op $s(2) $n(1) orient right-up
$ns duplex-link-op $n(1) $n(2) orient right-up
$ns duplex-link-op $n(3) $d(2) orient right-down
$ns duplex-link-op $s(3) $n(3) orient down
$ns duplex-link-op $n(4) $n(5) orient right-up
$ns duplex-link-op $n(5) $d(3) orient right-up

#thiet lap vi tri hang doi
$ns duplex-link-op $n(1) $n(2) queuePos 0.5
```

```
$ns duplex-link-op $n(2) $n(3) queuePos 0.5
$ns duplex-link-op $n(3) $n(4) queuePos 0.5
$ns duplex-link-op $n(4) $n(5) queuePos 0.5

#thiet lap kich thuoc hang doi
$ns queue-limit $n(1) $n(2) 10
$ns queue-limit $n(2) $n(3) 10
$ns queue-limit $n(3) $n(4) 10
$ns queue-limit $n(4) $n(5) 10

#tao TCP agent and attach it to node s1, s2, s3
set tcp1 [new Agent/TCP]
$tcp1 set fid_ 1
#$tcp1 set packetSize_ $pksize
$ns attach-agent $s(1) $tcp1

set tcp2 [new Agent/TCP]
$tcp2 set fid_ 2
#$tcp2 set packetSize_ $pksize
$ns attach-agent $s(2) $tcp2

set tcp3 [new Agent/TCP]
$tcp3 set fid_ 3
#$tcp3 set packetSize_ $pksize
$ns attach-agent $s(3) $tcp3

#tao 1 sink agent va noi lien no voi cac node d1, d2, d3
set sink1 [new Agent/TCPSink]
$ns attach-agent $d(1) $sink1

set sink2 [new Agent/TCPSink]
$ns attach-agent $d(2) $sink2

set sink3 [new Agent/TCPSink]
$ns attach-agent $d(3) $sink3

#connect the traffic sources with the traffic sinks
$ns connect $tcp1 $sink1
$ns connect $tcp2 $sink2
$ns connect $tcp3 $sink3

proc sendpacket1 {} {
   global ns tcp1 ArrivalTime1 pksize
   set now [$ns now]
   $ns at [expr $now + [$ArrivalTime1 value]] "sendpacket1"
   $tcp1 send $pksize
}

proc sendpacket2 {} {
   global ns tcp2 ArrivalTime2 pksize
   set now [$ns now]
   $ns at [expr $now + [$ArrivalTime2 value]] "sendpacket2"
   $tcp2 send $pksize
```

```
}

proc sendpacket3 {} {
    global ns tcp3 ArrivalTime1 pksize
    set now [$ns now]
    $ns at [expr $now + [$ArrivalTime1 value]] "sendpacket3"
    $tcp3 send $pksize
}

#thoi gian de phat di 1 goi
set ArrivalTime1 [new RandomVariable/Exponential]
$ArrivalTime1 set avg_ [expr 1/$lambda1]

set ArrivalTime2 [new RandomVariable/Exponential]
$ArrivalTime2 set avg_ [expr 1/$lambda2]

#cac ham sau loc du lieu va luu vao trace file de ve do thi bang thong va do thi toc do mat goi
proc record_bw {} {
    global sink1 sink2 sink3 f1 f2 f3
    #get an instance of the simulator
    set ns [Simulator instance]
    #set the time after which the procedure should be called again
    set time 0.1
    #how many bytes have been received by the traffic sinks
    #bytes_ = number of received bytes = so byte nhan duoc o dich
    set bw1 [$sink1 set bytes_]
    set bw2 [$sink2 set bytes_]
    set bw3 [$sink3 set bytes_]
    #get the current time
    set now [$ns now]
    #calculate the bandwidth (in MBit/s) and write it to the files
    puts $f1 "$now [expr $bw1/$time * 8/1000000]"
    puts $f2 "$now [expr $bw2/$time * 8/1000000]"
    puts $f3 "$now [expr $bw3/$time * 8/1000000]"
    #reset the byte_ values on the traffic sinks
    $sink1 set bytes_ 0
    $sink2 set bytes_ 0
    $sink3 set bytes_ 0
    #re-schedule the procedure
    $ns at [expr $now + $time] "record_bw"
}

#thuc hien chay mo phong trong 100s
#lap tien trinh cac su kien
$ns at 0.0 "record_bw"
$ns at 0.1 "sendpacket1"
$ns at 0.1 "sendpacket2"
$ns at 0.1 "sendpacket3"
#goi thu tuc 'finish' ket thuc chuong trinh
$ns at 100 "finish"

#chay mo phong
$ns run
```