TP2 Follow up

Vidal Sayarath

May 1, 2024

1 Visualization of the queue

The following figure 1 represents our modelisation of the network. The blue color corresponds to the route 0-1-2-3-4-5 and the red to the other route.

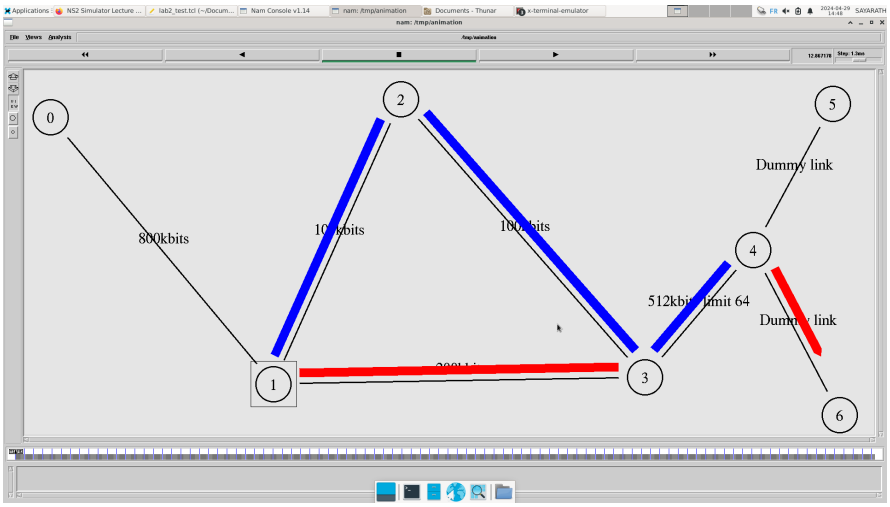

Figure 1: Queue system in NAM

2 Questions

2.1 Question (2) Varying b

To find bmax, we can run the code given underneath and simulate the system for different values of b. We set the probability of loss between the links 1-2 and 2-3 to p=0.1. The input rate b has higher step for lower values of b and the step is decreased significantly around where the delay R begins to exponentially increase, so around 260 (as seen in figure 2) This means there are more simulation points around 260 to 270.

Looking at 2, the network becomes unstable and congested around $b_{max} = 268$.

2.2 Question (3) Varying p

We now set $b = 100kbits$ and vary the loss probability p from 0 to 1. We wish to study the packet loss ratio as a function of p where the packet loss ratio is calculated as :

$$\text{loss ratio} = \frac{\text{Nb of sent packets} - \text{Nb of received packets}}{\text{Nb of sent packets}}$$

This gives the following figure 3

Looking at 3 shows that for a loss ratio of 5 percent which corresponds to 0.05 on our graphic, that $p_0 \approx 0.1$

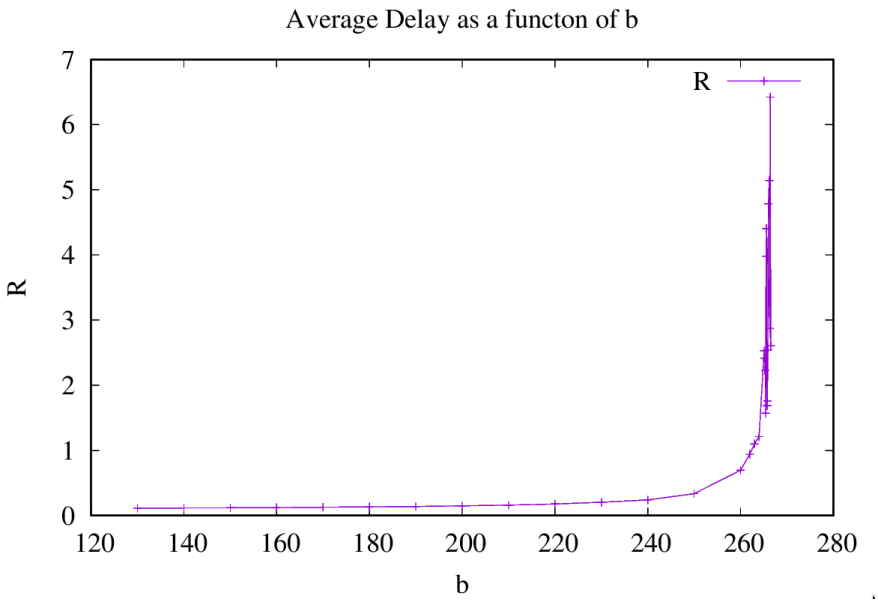

Figure 2: Plot of B vs R (delay)

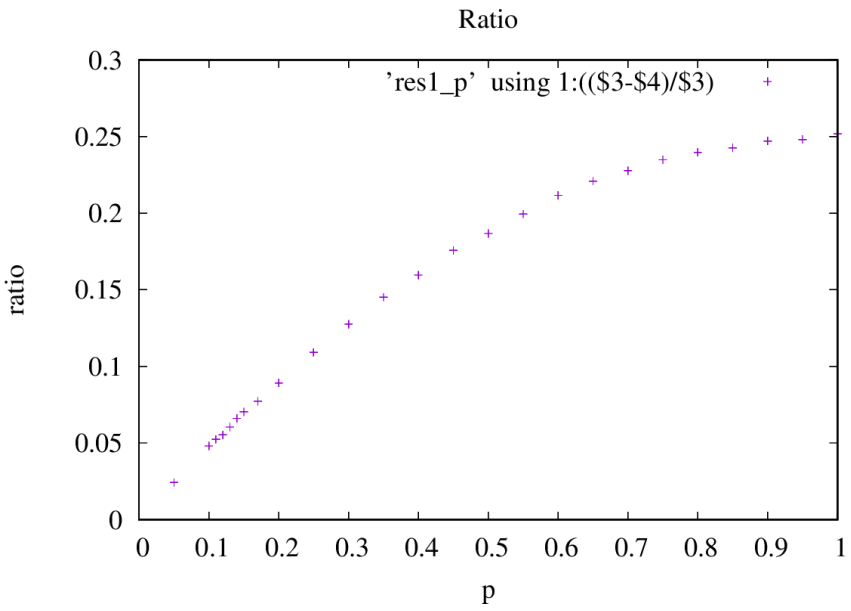

Figure 3: Plot loss ratio as function of p with b=100

It is coherent that the loss ratio of 5 percent is lower than the dropping ratio p_0. The loss ratio represents the proportion of packets lost out of the total packets sent, which can be lost due to factors such as a full buffer. The dropping ratio p_0 is the probability of a packet being dropped at a specific point in the network. The overall loss ratio is influenced by the loss between the links 1-2 and 2-3.

3 Question (4)

The bottleneck of the network is the first 'weak' link. Here the bottleneck of the link is intuitively either the link 1-2 or the link 1-3. This is because the link 0-1 has a high bandwidth, the links 4-5 and 4-6 as well. The link 2-3 cannot be the bottleneck since the link 1-2 has the same bandwidth but

is before. The link 3-4 cannot be the bottleneck since it also has a higher bandwidth than the other links.

We can now calculate which link is the bottleneck. Considering that the probabilties are $\frac{1}{4}$ and $\frac{3}{4}$ for the links 1-2 and 2-3 respectively. We have :

$$\frac{1}{4}b_{max} = 100\text{kbits}$$

and

$$\frac{3}{4}b_{max} = 200\text{kbits}$$

If we solve the first equation, we get $b_{max} = 400$ and if we solve the second equation, we get $b_{max} = \frac{800}{3} \approx 266.66$.

So the bottleneck of the system is the lower of the two. Hence $b_{max} \approx 266.66$ This value is coherent with our graph, since if we look at 2, the system becomes unstable around this value.

4 Question (5)

We have the loss ratio given by the formula :

$$t_{pp} = \frac{\text{Total input rate - Total output rate}}{\text{Total input rate}}$$

where the total input rate is given by b by definition.
In order to calculate the total output rate, we need to take into account that :

- There is $\frac{1}{4}$ probability of taking the route 0-1-2-3. With probability of losing a packet p on the link 1-2 and on the link 2-3. So the output rate from this route is $(1-p)^2 * \frac{b}{4}$
- There is $\frac{3}{4}$ probability of taking the route 0-1-3 with no losses, so the output rate from this route is $b * \frac{3}{4}$

With this, we can write output_rate $= (1-p)^2 \cdot \frac{b}{4} + b \cdot \frac{3}{4}$
Hence :

$$\text{Total output rate} = \left(\frac{b}{4} \cdot (1-p)^2\right) + b \cdot \frac{3}{4} \tag{1}$$

$$t_{PP} = \frac{b - \left[\left(\frac{b}{4} \cdot (1-p)^2\right) + b \cdot \frac{3}{4}\right]}{b} \tag{2}$$

$$t_{PP} = \frac{b - \left(\frac{b}{4} \cdot (1-p)^2\right) - b \cdot \frac{3}{4}}{b} \tag{3}$$

$$t_{PP} = \left(1 - \left(\frac{1}{4} \cdot (1-p)^2\right) - \frac{3}{4}\right) \tag{4}$$

$$t_{PP} = \frac{1}{4} \cdot \left[1 - (1-p)^2\right] \tag{5}$$

Solving for p with $t_{pp} = 0.05$ gives :

$$\frac{1}{4} \cdot \left[1 - (1-p)^2\right] = 0.05 \tag{6}$$

$$1 - (1-p)^2 = 0.2 \tag{7}$$

$$(1-p)^2 = 0.8 \tag{8}$$

$$1 - p = \sqrt{0.8} \tag{9}$$

$$\sqrt{0.8} \approx 0.8944 \tag{10}$$

$$p = 1 - 0.8944 \tag{11}$$

$$p \approx 0.1056 \tag{12}$$

5 Question 6

Using the Simulation and by reading the graphics roughly, we can compare our analytical results and simulation results with the following table :

	Simulation	Analytical	Error
$b_{\max}$	268	≈ 266.67	≈ 1.33
p_0	0.1	≈ 0.1056	≈ 0.0056

Table 1: Comparison of simulation and analytical results with error estimates.

Our results are close enough to the analytical results, confirming that our simulation is working as intended. If we want closer results to the analytical, we need to vary b and p with smaller steps and visualize closer the parts which interest us.

6 QUestion (7)

To model the studied network, the M/M/1 queueing model is the better choice due to the following reasons:

- **Single Server and FIFO Queue**: The M/M/1 model features one server and follows a FIFO queueing, which correponds to the network we are studying, where each link acts as a single server.
- **Exponential Inter-arrival and Service Times**: The model assumes exponential inter-arrival and service times, which is consistent with the studied network, including a Poisson distribution of packets.
- **Analytical Simplicity**: The M/M/1 model offers straightforward analytical solutions for metrics such as average delay, making it easier to model and analyze the network.

However, since we have two different routes, we need to define two different M/M/1 models and add the delays together. The parameters for both M/M/1 models are the arrival rates $\lambda = b$ and the service rates $\mu_1 = 100 \text{and} \mu_2 = 200$. We will compute the delay as :

$$R_{\text{combined delay}} = R_{\text{First MM1}} + R_{\text{second MM1}}$$

which gives the plot 4:

NOTE : The values for the delay were multiplied by 10 for it to have the same delays as the simulation, im unsure why it is the case.

In conclusion, the two M/M/1 models are good initial approximation for understanding the behavior of the network since the two delays follow the same pattern closely, but we still need to explain the factor 10 for the delay.

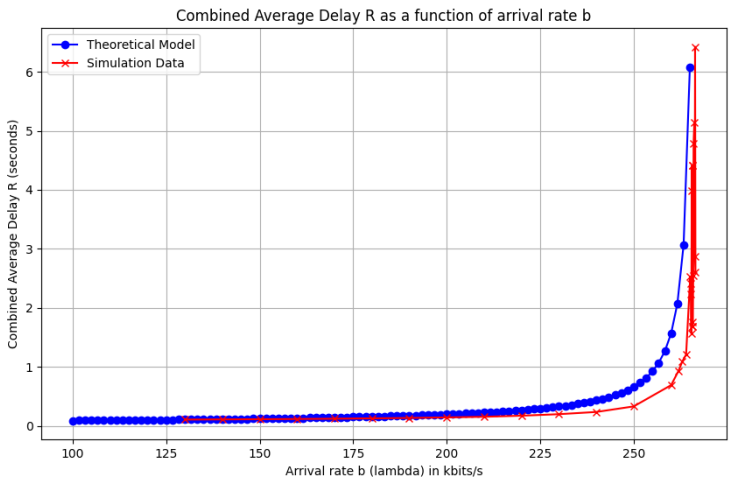

Figure 4: Comparison of two M/M/1 queues and simulation

7 Question (8)

We can modify the simulation code just by changing the queue limit to five. Meaning we set :

```
# Configure here the size of all link queues
$ns queue-limit $n(0) $n(1) 5
$ns queue-limit $n(1) $n(2) 5
$ns queue-limit $n(1) $n(3) 5
$ns queue-limit $n(3) $n(4) 5
$ns queue-limit $n(4) $n(5) 5
$ns queue-limit $n(4) $n(6) 5
```

Which gives the following plot 5:

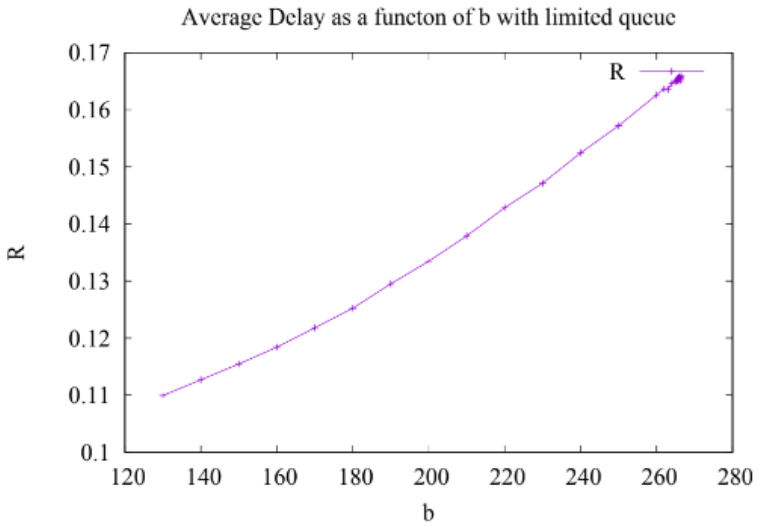

Figure 5: Delay R as a function of B for a queue limit of 5

In this case, a better model than the two M/M/1 models given in the previous question would be to use two different M/M/1/5 models where 5 is the buffer limit.

The M/M/1/5 model considers the finite buffer capacity of 5 which would better represent the packet loss and delay as the queue becomes full.

If we look at the figure 5 , we see that the limited queue system begins to "degrade" more slowly.Meaning the the delay does not become unstable. This is due to the fact that instead of creating a large (maybe infinite) queue, the extra packets are just dropped. This also explains the fact that in a limited queue system, the delay is low compared to an infinite queue system.

Using our new model of two M/M/1/5 queues added together, we get the following plot 6.

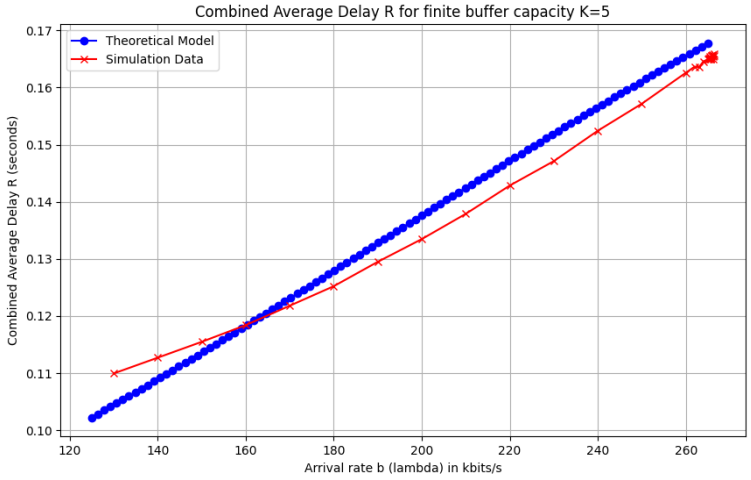

Figure 6: Delay R for two M/M/1/5 as a function of b for a queue limit of 5

As seen in the plot, the two curve folllow relatively closely, further justifying our model. However, there is still the problem of the factor 10.

8 Question 9

We can start by talking about the advantages and disadvantages of each.

8.0.1 Analytical Modeling

Advantages:

- **Efficiency:** Provides quick results with less computational effort.
- **Predictability:** Uses well-defined mathematical formulas, which give exact results. (For example 800 / 3 is the bottleneck of our system)
- **Cost-effective:** Requires less resources compared to simulations.

Disadvantages:

- **Simplifications:** Often relies on simplifications that may not capture all real-world complexities.
- **Assumptions:** Assumes ideal conditions which might not hold true, leading to potential inaccuracies.
- **Limited scenarios:** Finding and solving the equations to complex queue systems may be too difficult.

8.0.2 Simulations

Advantages:

- **Capacity to study complex systems:** Can model complex network behaviors.
- **Versatility:** Applicable to practically any queing system.
- **Detailed statistics:** Provides any statistics we want without having to find complex equations. (Such as jitter)
- **Randomness :** Simulations can be used with randomness programmed into the system.

Disadvantages:

- **Resource-intensive:** Requires significant computational resources and time, especially for large scale networks.
- **Precision:** We can never get the exact value of theoretical value (such as bottleneck). However, like in our queue system, there is randomness in the queue system that will always be present, thus the theoretical value will never be attained but the simulation value will be close to it.

In conclusion, both analytical modeling and simulations provide several advantages and disadvantages. Depending on our system and our background knowledge, such as finding formulas, we need to choose accordingly.

9 Code

The following subsections presents the code used to give the results and plots above.

Of course, this is not all the code. For example, the code to calculate the system for a queue size of 5 was done using the same code but just changing the names of the files and a part of the queue modelisation code.

9.1 Analyze file to calculate all the statistics of the simulation

```
    BEGIN {
        highest_packet_id = 0;
        nrecvd = 0;
        nsent = 0;
        ndrops = 0;
        simtime = 0;
        startsim = 0.01;
        meanNbClients = 0.0;
        nclients = 0;
        lasttime = startsim;
}

{
        action = $1;
        time = $2;
        node_1 = $3;
        node_2 = $4;
        traffic = $5;
        flow_id = $8;
        node_1_address = $9;
        node_2_address = $10;
        seq_no = $11;
        packet_id = $12;
```

```
        # split($2, a, ".");
# time = a[1] "," a[2];
simtime = time;
        if ( packet_id > highest_packet_id )
                highest_packet_id = packet_id;

        if ( action == "+" && node_1 == 0 ) {
                nsent++;
                start_time[packet_id] = time;
                traffictype[packet_id] = traffic;
                meanNbClients = 1.0*meanNbClients + 1.0*nclients*(time-lasttime);
                #printf("%d %f %f\n", nclients, nclients*(time-lasttime), meanNbClients);
                nclients++;
                lasttime = time;
        }
        if ( action != "d" ) {
                if ( action == "r" && node_2 == 4) {
                        end_time[packet_id] = time;
                        meanNbClients = 1.0*meanNbClients + 1.0*nclients*(time-lasttime);
                        #printf("%d %f %f\n", nclients, nclients*(time-lasttime), meanNbClients);
                        nclients--;
                        lasttime = time;
                }
        } else {
                meanNbClients = 1.0*meanNbClients + 1.0*nclients*(time-lasttime);
                #printf("%d %f %f\n", nclients, nclients*(time-lasttime), meanNbClients);
                nclients--;
                end_time[packet_id] = -1;
                ndrops++;
        }
}
END {
        for ( packet_id = 0; packet_id <= highest_packet_id; packet_id++ ) {
                start = start_time[packet_id];
                end = end_time[packet_id];
                if(start < end) {
                        nrecvd++;
                        packet_duration += end - start;
                        packet_duration2 += (end-start)**2;
                        #printf("%d %f\n", packet_id, end-start);
                }
        }
        if(nrecvd) {
                delay = packet_duration/nrecvd;
                if((packet_duration2/nrecvd)-(packet_duration/nrecvd)**2.0>0) {
                        jitter = sqrt((packet_duration2/nrecvd)-(packet_duration/nrecvd)**2.0);
                } else {
                        jitter = 0;
                }
                xe = 1.0*1024*8*nsent/(1000*simtime);
                xs = 1.0*1024*8*nrecvd/(1000*simtime);
                meanNbClients = 1.0*meanNbClients/simtime;
#SimTime NbOfSentPackets NbofRecvdPackets NbDrops InputRate OutputRate Delay Jitter
                printf("%f %d %d %d %f %f %f %f %f\n", simtime, nsent, nrecvd, ndrops, xe, xs, delay,
        }
}
```

9.2 runsim file of the simulations of varying b

```
    #
# Generates results in columns ready to be plotted
# The first column is lambda (the input rate)
# The second column is the simulation time
# Etc.
#

resfile=res1

rm -i $resfile

#seed=21113944 # <<<<<<< CHANGE HERE
b=120
p=0.1
while [ $b -le 250 ]
do
  b=$(echo "$b + 10" | bc)
  ns lab2_test.tcl $b $p
  echo -n "Analyzing ... "
  echo -n "$b " >> $resfile; awk -f analyze_delay.awk /tmp/out.tr >> $resfile
  echo " "
done

for b in 262 263 264 265.1 265.2 265.3 265.4 265.5 265.6 265.7 265.8 265.9 266 266.2 266.3 266.4 266.
  ns lab2_test.tcl $b $p
  echo -n "Analyzing ... "
  echo -n "$b " >> $resfile; awk -f analyze_delay.awk /tmp/out.tr >> $resfile
  echo "End."
  echo "Results summary line for the last simulation:"
  tail -1 $resfile
  echo " "
done




printf "\n\n\tDone\n"

printf "\n\tThe results are in $resfile\n\n"
```

9.3 Simulation of varying p

```
    #
# Generates results in columns ready to be plotted
# The first column is lambda (the input rate)
# The second column is the simulation time
```

```
# Etc.
#

resfile=res1_p

rm -i $resfile

#seed=21113944 # <<<<<<< CHANGE HERE
b=100
p=0.1
for p in 0.05 0.1 0.11 0.12 0.13 0.14 0.15 0.17 0.20 0.25 0.30 0.35 0.40 0.45 0.5 0.55 0.60 0.65 0.70
  ns lab2_test.tcl $b $p
  echo -n "Analyzing ... "
  echo -n "$p " >> $resfile; awk -f analyze_delay.awk /tmp/out.tr >> $resfile
  echo " "
  echo "Results summary line for the last simulation:"
  tail -1 $resfile
  echo " "
done



printf "\n\n\tDone\n"

printf "\n\tThe results are in $resfile\n\n"
```

9.4 Code to plot delay R as a function of b

```
    # To generate a new figure, modify the title,
# xlabel and ylabel and comment/uncomment the adequate instructions.

set title 'Average Delay as a functon of b'
set xlabel '{b}'
set ylabel 'R'

set term postscript eps enhanced color "Times-Roman" 22
set output 'Average_b_delay.eps'

# plot the 7th field of the file res1 as a function of the 1st field

plot "res1" using 1:8 t "R" w lp
```

9.5 Code to plot the packet loss ratio as a function of p

```
# To generate a new figure, modify the title,
# xlabel and ylabel and comment/uncomment the adequate instructions.

set title 'Ratio'
```

```
set xlabel '{p}'
set ylabel 'ratio'
set style line 1 lt 2 lw 2 pt 3 ps 0.5

set term postscript eps enhanced color "Times-Roman" 22
set output 'ratio.eps'



plot 'res1_p'  using 1:(($3-$4)/$3)
```

10 Modelising the system by 2 M/M/1 queues for question 7 in python (with plots)

```
import numpy as np
import matplotlib.pyplot as plt

# Param
b_values = np.linspace(100, 265, 100)
p1 = 0.25
p2 = 0.75
mu1 = 100  # Service rate for the top route
mu2 = 200  # Service rate for the bottom route

# Calculate average delay for each b using the two M/M/1 models
delays = []
for b in b_values:
    lambda_1 = p1 * b  # Arrival rate for the first queue
    lambda_2 = p2 * b  # Arrival rate for the second queue

    if lambda_1 < mu1 and lambda_2 < mu2:
        R1 = 1 / (mu1 - lambda_1)  # Delay for the first M/M/1 system
        R2 = 1 / (mu2 - lambda_2)  # Delay for the second M/M/1 system
        R_total = p1 * R1 + p2 * R2
        delays.append(R_total)
    else:
        delays.append(float('inf'))

# Plotting the theoretical model results
plt.figure(figsize=(10, 6))
plt.plot(b_values, np.array(delays)*10, marker='o', linestyle='-', color='blue', label='Theoretical M

# Load and plot data from 'res1'
data = np.loadtxt('res1', delimiter=' ')
plt.plot(data[:, 0], data[:, 7], marker='x', linestyle='-', color='red', label='Simulation Data')
```

```
plt.title('Combined Average Delay R as a function of arrival rate b')
plt.xlabel('Arrival rate b (lambda) in kbits/s')
plt.ylabel('Combined Average Delay R (seconds)')
plt.legend()
plt.grid(True)
plt.show()
```

10.1 Question 8 comparison plot and calculation for two M/M/1/5 queues

```
    import numpy as np
import matplotlib.pyplot as plt

# Parameters
b_values = np.linspace(125, 265, 100)
p1 = 0.25
p2 = 0.75
mu1 = 100  # Service rate for the top route
mu2 = 200  # Service rate for the bottom route
K = 5  # Buffer size

def mm1k_delay(lam, mu, K):
    rho = lam / mu
    if rho < 1:
        P0 = (1 - rho) / (1 - rho**(K+1))
        PK = rho**K * P0
        # Average number of customers in the system (Formula from FICHE)
        L = (rho - (K+1)*rho**(K+1) + K*rho**(K+2)) / ((1-rho)*(1-rho**(K+1)))
        # Average delay in the system
        W = L / (lam * (1 - PK))
        return W
    else:
        return float('inf')

# Calculate average delay for each b using the two M/M/1/K models
delays = []
for b in b_values:
    lambda_1 = p1 * b  # Arrival rate for the first queue
    lambda_2 = p2 * b  # Arrival rate for the second queue

    if lambda_1 < mu1 and lambda_2 < mu2:
        R1 = mm1k_delay(lambda_1, mu1, K)  # Delay for the first M/M/1/K system
        R2 = mm1k_delay(lambda_2, mu2, K)  # Delay for the second M/M/1/K system
        R_total = p1 * R1 + p2 * R2
        delays.append(R_total)
    else:
        delays.append(float('inf'))

# Plotting the theoretical model results
plt.figure(figsize=(10, 6))
plt.plot(b_values, np.array(delays)*10, marker='o', linestyle='-', color='blue', label='Theoretical M


data = np.loadtxt('res1_changedqueue', delimiter=' ')
plt.plot(data[:, 0], data[:, 7], marker='x', linestyle='-', color='red', label='Simulation Data')

plt.title('Combined Average Delay R for finite buffer capacity K=5')
```

```
plt.xlabel('Arrival rate b (lambda) in kbits/s')
plt.ylabel('Combined Average Delay R (seconds)')
plt.legend()
plt.grid(True)
plt.show()
```

10.2 Queue modelisation

```
#  Create the main simulaton object that implements especially the event scheduler.
set ns [new Simulator]

# Trace file:
$ns trace-all [open /tmp/out.tr w]

# nam trace file for network animations:
$ns namtrace-all [open /tmp/animation w]
$ns color 1 Blue
$ns color 2 Red

# Some input parameters:
# the traffic rate and the random generator seed

set seed 21113944
if {$argc > 0 } { set b [lindex $argv 0] }
if {$argc > 1 } { set p [lindex $argv 1] }
puts "Simulation with  b = $b ,  p = $p"

global defaultRNG
$defaultRNG seed $seed;


# Create nodes
set n(0) [$ns node]    ;$n(0) set X_ 30   ;$n(0) set Y_ 10
set n(1) [$ns node]    ;$n(1) set X_ 40   ;$n(1) set Y_ 10
set n(2) [$ns node]    ;$n(2) set X_ 45   ;$n(2) set Y_ 20
set n(3) [$ns node]    ;$n(3) set X_ 50   ;$n(3) set Y_ 10
set n(4) [$ns node]    ;$n(4) set X_ 60   ;$n(4) set Y_ 10 # Extra node for the packet limit on 3
set n(5) [$ns node]    ;$n(5) set X_ 70   ;$n(5) set Y_ 20 # Destination For First poisson process
set n(6) [$ns node]    ;$n(6) set X_ 70   ;$n(6) set Y_ 10 # For second poisson process


# Create links
$ns simplex-link $n(0) $n(1) 800kb 0ms DropTail  ;$ns simplex-link-op $n(0) $n(1) label " 800kbits"
$ns simplex-link $n(1) $n(2) 100kb 0ms DropTail  ;$ns simplex-link-op $n(1) $n(2) label " 100kbits"
$ns simplex-link $n(1) $n(3) 200kb 0ms DropTail  ;$ns simplex-link-op $n(1) $n(3) label " 200kbits"
$ns simplex-link $n(2) $n(3) 100kb 0ms DropTail  ;$ns simplex-link-op $n(2) $n(3) label " 100kbits"
$ns simplex-link $n(3) $n(4) 512kb 0ms DropTail  ;$ns simplex-link-op $n(3) $n(4) label " 512kbits li
$ns simplex-link $n(4) $n(5) 1000kb 0ms DropTail  ;$ns simplex-link-op $n(4) $n(5) label " Dummy link
$ns simplex-link $n(4) $n(6) 1000kb 0ms DropTail  ;$ns simplex-link-op $n(4) $n(6) label " Dummy link



#Defining Links to get head after
set link01 [$ns link $n(0) $n(1)]
set link12 [$ns link $n(1) $n(2)]
set link13 [$ns link $n(1) $n(3)]
```

```
set link23 [$ns link $n(2) $n(3)]
set link34 [$ns link $n(3) $n(4)]
set link45 [$ns link $n(4) $n(5)]
set link46 [$ns link $n(4) $n(6)]



# Configure here the size of all link queues ("infinite")
$ns queue-limit $n(0) $n(1) 1000000
$ns queue-limit $n(1) $n(2) 1000000
$ns queue-limit $n(1) $n(3) 1000000
$ns queue-limit $n(3) $n(4) 1000000
$ns queue-limit $n(4) $n(5) 1000000
$ns queue-limit $n(4) $n(6) 1000000


# Add here all entries of routing tables
$ns rtproto Manual

$n(0) add-route [$n(5) id] [$link01 head]
$n(0) add-route [$n(6) id] [$link01 head]

$n(1) add-route [$n(5) id] [$link12 head]
$n(1) add-route [$n(6) id] [$link13 head]

$n(2) add-route [$n(5) id] [$link23 head]


$n(3) add-route [$n(5) id] [$link34 head]
$n(3) add-route [$n(6) id] [$link34 head]

$n(4) add-route [$n(5) id] [$link45 head]
$n(4) add-route [$n(6) id] [$link46 head]




# Add here the uniform losses over links (1)--(2) and (2)--(3)


# Create an error model and configure it
set losses [new ErrorModel]
$losses set rate_ $p       ;
$losses unit pkt           ;
$losses ranvar [new RandomVariable/Uniform] ;
$losses drop-target [new Agent/Null] ;

set losses2 [new ErrorModel]
$losses2 set rate_ $p       ;
$losses2 unit pkt           ;
$losses2 ranvar [new RandomVariable/Uniform] ;
$losses2 drop-target [new Agent/Null] ;


# Attach the error model to specific links
```

```
$ns lossmodel $losses $n(1) $n(2)
$ns lossmodel $losses2 $n(2) $n(3)




# UDP
# Create here agents for the UDP transport layer
#Creating the first Poisson Process
set udp1 [new Agent/UDP]
$udp1 set packetSize_ 1000
$udp1 set fid_ 1
$ns attach-agent $n(0) $udp1

set null1 [new Agent/Null]
$ns attach-agent $n(5) $null1
$ns connect $udp1 $null1


set udp2 [new Agent/UDP]
$udp2 set packetSize_ 1000
$udp2 set fid_ 2
$ns attach-agent $n(0) $udp2


set null2 [new Agent/Null]
$ns attach-agent $n(6) $null2
$ns connect $udp2 $null2


# Add here traffic generators for the application layer (Poisson) Through On-Off
set poisson1 [new Application/Traffic/Exponential]
$poisson1 attach-agent $udp1

set poisson2 [new Application/Traffic/Exponential]
$poisson2 attach-agent $udp2



# Poisson Traffic (Obtained from an ON/OFF Exponential traffic)
$poisson1 set rate_ 1000Mb     # To ensure sending one packet
$poisson1 set burst_time_ 0.0  # during the zero ON period
$poisson1 set idle_time_ [expr 32.0 / $b ] # lambda in packets per second
$poisson1 set packetSize_ 1000


$poisson2 set rate_ 1000Mb     # To ensure sending one packet
$poisson2 set burst_time_ 0.0  # during the zero ON period
$poisson2 set idle_time_ [expr 32.0 / (3.0 * $b)] # lambda in packets per second
$poisson2 set packetSize_ 1000
```

```
# Schedule here when to stop the sending of traffic packets (at t=10000)
$ns at 0.01 "$poisson1 start"
$ns at 10000.00 "$poisson1 stop"

$ns at 0.01 "$poisson2 start"
$ns at 10000.00 "$poisson2 stop"

$ns at 10001.1 "$ns halt"
puts -nonewline "Running ...."; flush stdout

$ns run
puts " End."
puts "Trace file out.tr is in directory /tmp"
puts "Animation file animation is in directory /tmp"
```

References