Modelação e Desempenho de Redes e Serviços

Universidade de Aveiro

João Ferreira, Rafael Santos

Modelação e Desempenho de Redes e Serviços

Universidade de Aveiro

João Ferreira, Rafael Santos
(103625) joaop.ferreira@ua.pt, (98466) rafaelmsantos@ua.pt

Outubro de 2024

Índice

Chapter 1

Task 1

1.1 Exercise 1.a)

1.1.1 Results and Conclusions

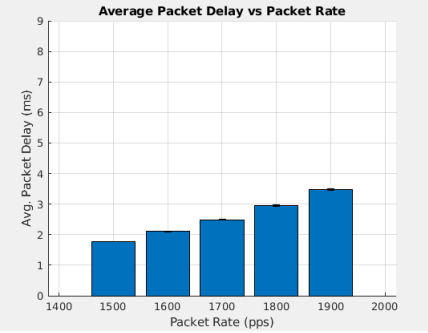

Figure 1.1: The average data packet delay results, $b = 10^{-6}$

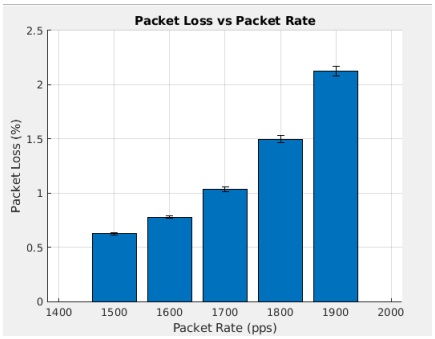

Figure 1.2: The average data packet loss results, $b = 10^{-6}$

When the packet rate in a system goes up, we expect the average packet delay to increase. Looking at the results, Figure 1.1 shows that the average packet delay indeed gets longer as the packet rate rises. This occurs because the arrival rate of packets is getting close to what the system can handle, leading to increased queuing times for each packet as they await processing.

In contrast, Figure 1.2 shows a steady increase in packet loss as the arrival rate rises, unlike the previous assumption that packet loss would remain unchanged. The system's packet loss rate is sensitive to the growing arrival rate, suggesting that, as the packet rate nears the system's capacity, packets are more frequently dropped. This increase in packet loss as arrival rates climb indicates that the queue size alone does not provide enough buffering for the higher load. Instead, packet loss starts to escalate as the arrival rate approaches and surpasses certain thresholds, showing that the system's ability to handle incoming packets diminishes significantly under higher load conditions.

The observed packet loss likely results from a combination of the constant bit error rate in the system's connection and the overload caused by the higher packet rate.

1.2 Exercise 1.b)

1.2.1 Results and Conclusions

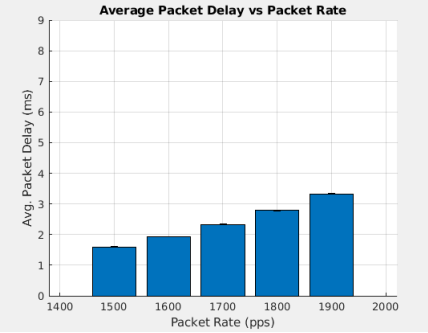

Figure 1.3: The average data packet delay results, $b = 10^{-4}$

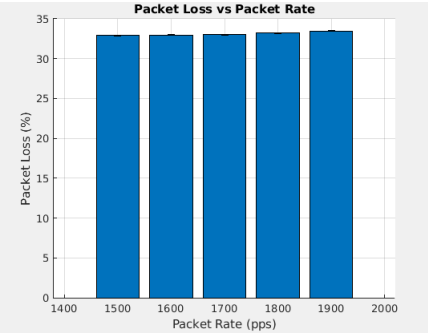

Figure 1.4: The average data packet loss results, $b = 10^{-4}$

When increasing the bit error rate by 100 times, a significant rise in packet loss is expected. Looking at Figures 1.3 and 1.4, we observe that packet loss increased from around between 0.5%-2.2% to approximately around 33%/34%. Meanwhile, the average packet delay shows little to no change with the higher bit error rate.

The jump in packet loss is expected because a higher bit error rate leads to more packet loss. However, the average packet delay remains unaffected, as explained previously. Since the bit error rate doesn't impact the link's capacity or the rate at which packets arrive, we would expect the delay times to stay fairly constant.

1.3 Exercise 1.c)

1.3.1 Code

```
% Exercicio 1

%% 1.c)
% Probability for different packet data sizes from 64, 110, and 1518 Bytes
prob_left = (1 - (0.19 + 0.23 + 0.17)) / ((109 - 65 + 1) + (1517 - 111 + 1));
ber_values = [1e-6, 1e-4];      % Bit error rates to test
ploss_ber = zeros(1, length(ber_values));  % Packet loss probabilities

% Loop over each bit error rate
for i = 1:length(ber_values)
    ber = ber_values(i);

    % Calculate loss probability for each packet size
    for packetSize = 64:1518
        no_error_prob = (1 - ber)^(packetSize * 8); % Probability of no bit error

        % Check specific packet sizes with predefined probabilities
        if packetSize == 64
            ploss_ber(i) = ploss_ber(i) + (1 - no_error_prob) * 0.19;
        elseif packetSize == 110
            ploss_ber(i) = ploss_ber(i) + (1 - no_error_prob) * 0.23;
        elseif packetSize == 1518
            ploss_ber(i) = ploss_ber(i) + (1 - no_error_prob) * 0.17;
        else
            ploss_ber(i) = ploss_ber(i) + (1 - no_error_prob) * prob_left;
        end
    end
end

% Convert results to percentage
ploss_ber = ploss_ber * 100;

% Display the results
fprintf(['Considering that the bit error rate is the only factor causing packet loss,\n' ...
    'the theoretical packet loss (%%) for a bit error rate of 10^-6 is: %.4f %%\n' ...
    'and for a bit error rate of 10^-4 is: %.4f %%\n'], ploss_ber(1), ploss_ber(2));
```

The code defines the probability for packets with fixed sizes (64, 110, and 1518 Bytes), given a priori, and calculates the probability for other sizes between 65 and 1517 Bytes. For each packet size between 64 and 1518 Bytes, the code calculates the probability of no error occurring. This probability represents the chance of a packet being transmitted successfully, with no errors in any of the bits. To obtain the probability of at least one error occurring, the code subtracts this probability from 1. Then, the code adjusts the packet loss calculation for specific sizes by multiplying the error probability by the weight of each packet size.

The program's output provided the following results: "Considering that the bit error rate is the only factor causing packet loss, the theoretical packet loss

(%) for a bit error rate of 10^{-6} is: 0.4937%, and for a bit error rate of 10^{-4} is: 32.8278%.

These results suggest that the packet loss observed in Figure 1.4 can be attributed entirely to the bit error rate, as our experimental values align closely with the theoretical values calculated in this code. This outcome supports the hypothesis that, under these conditions, packet loss is predominantly driven by bit errors rather than system processing limitations, as the system is able to handle all incoming packets within the given time constraints.

Chapter 2

Task 2

2.1 Exercise 2.a)

2.1.1 Sim3A Code

```
function [PLdata, PLvoip , APDdata, APDvoip , MPDdata, MPDvoip , TT] = Sim3A(lambda,C,f,P,n,B)
% INPUT PARAMETERS:
%  lambda - packet rate (packets/sec)
%  C      - link bandwidth (Mbps)
%  f      - queue size (Bytes)
%  P      - number of packets (stopping criterium)
%  n      - voip packet flows
%  B      - bit error rate
% OUTPUT PARAMETERS:
%  PLdata   - packet loss (%) of data packets
%  PLvoip  - Packet Loss (%) of VOip packets
%  APDdata  - average  data packet delay (milliseconds)
%  APDvoip  - average voip packet delay(miliseconds)
%  MPDdata  - maximum data packet delay (miliseconds)
%  MPDvoip  - maximum voip packet delay (miliseconds)
%  TT   - transmitted throughput (Mbps)

%Events:
ARRIVAL= 0;      % Arrival of a packet
DEPARTURE= 1;    % Departure of a packet

%State variables:
STATE = 0;          % 0 - connection is free; 1 - connection is occupied
QUEUEOCCUPATION= 0; % Occupation of the queue (in Bytes)
QUEUE= [];          % Size and arriving time instant of each packet in the queue

%Statistical Counters:
TOTALPACKETS= 0;     % No. of packets arrived to the system
TOTALPACKETSvoip=0;
LOSTPACKETS= 0;      % No. of packets dropped due to buffer overflow
LOSTPACKETSvoip=0;
TRANSPACKETS= 0;     % No. of transmitted packets
TRANSPACKETSvoip=0;
TRANSBYTES= 0;       % Sum of the Bytes of transmitted packets

DELAYS= 0;           % Sum of the delays of transmitted packets
DELAYSvoip=0;
MAXDELAY= 0;         % Maximum delay among all transmitted packets
MAXDELAYvoip=0;

% Initializing the simulation clock:
Clock= 0;

% Initializing the List of Events with the first ARRIVAL:
tmp= Clock + exprnd(1/lambda);
EventList = [ARRIVAL, tmp, GeneratePacketSize(), tmp, 0];

%generate VOip packets
for i=1:n
    tmp = unifrnd(0, 0.02);
    EventList = [EventList; ARRIVAL, tmp, randi([110, 130]), tmp, 1  ];
end
```

Figure 2.1: Sim3A - Part I

```
%Similation loop:
while (TRANSPACKETS+TRANSPACKETSvoip)<P % Stopping criterium
    EventList= sortrows(EventList,2);  % Order EventList by time
    Event= EventList(1,1);              % Get first event
    Clock= EventList(1,2);              %    and all
    PacketSize= EventList(1,3);        %    associated
    ArrInstant= EventList(1,4);    %    parameters.
    type= EventList(1,5);          %   get the packet type

    EventList(1,:)= [];                % Eliminate first event
    switch Event
        case ARRIVAL                    % If first event is an ARRIVAL
            if type == 0  % Data packet
                TOTALPACKETS= TOTALPACKETS+1;
                tmp= Clock + exprnd(1/lambda);
                EventList = [EventList; ARRIVAL, tmp, GeneratePacketSize(), tmp, 0];
                if STATE==0
                    STATE= 1;
                    EventList = [EventList; DEPARTURE, Clock + 8*PacketSize/(C*10^6), PacketSize, Clock, 0];
                else
                    if QUEUEOCCUPATION + PacketSize <= f
                        QUEUE= [QUEUE;PacketSize , Clock, 0];
                        QUEUEOCCUPATION= QUEUEOCCUPATION + PacketSize;
                    else
                        LOSTPACKETS= LOSTPACKETS + 1;
                    end
                end
            elseif type == 1 % VoIP packet
                TOTALPACKETSvoip= TOTALPACKETSvoip+1;
                tmp= Clock + unifrnd(0.016,0.024);
                EventList = [EventList; ARRIVAL, tmp, randi([110,130]), tmp, 1];
                if STATE==0
                    STATE= 1;
                    EventList = [EventList; DEPARTURE, Clock + 8*PacketSize/(C*10^6), PacketSize, Clock, 1];
                else
                    if QUEUEOCCUPATION + PacketSize <= f
                        QUEUE= [QUEUE;PacketSize , Clock, 1];
                        QUEUEOCCUPATION= QUEUEOCCUPATION + PacketSize;
                    else
                        LOSTPACKETSvoip= LOSTPACKETSvoip + 1;
                    end
                end
            end
```

Figure 2.2: Sim3A - Part II

```
        case DEPARTURE                  % If first event is a DEPARTURE
            if type == 0 % Data Packet
                if rand()<(1-B)^(PacketSize*8) % check if all bits are transmitted
                    TRANSBYTES= TRANSBYTES + PacketSize;
                    DELAYS= DELAYS + (Clock - ArrInstant);
                    if Clock - ArrInstant > MAXDELAY
                        MAXDELAY= Clock - ArrInstant;
                    end
                    TRANSPACKETS= TRANSPACKETS + 1;
                else
                    LOSTPACKETS = LOSTPACKETS+1;
                end
                if QUEUEOCCUPATION > 0
                    EventList = [EventList; DEPARTURE, Clock + 8*QUEUE(1,1)/(C*10^6), QUEUE(1,1), QUEUE(1,2), QUEUE(1,3)];
                    QUEUEOCCUPATION= QUEUEOCCUPATION - QUEUE(1,1);
                    QUEUE(1,:)= [];
                else
                    STATE= 0;
                end
            elseif type == 1  % VoIP Packet
                if rand()<(1-B)^(PacketSize*8)
                    TRANSBYTES= TRANSBYTES + PacketSize;
                    DELAYSvoip= DELAYSvoip + (Clock - ArrInstant);
                    if Clock - ArrInstant > MAXDELAYvoip
                        MAXDELAYvoip= Clock - ArrInstant;
                    end
                    TRANSPACKETSvoip= TRANSPACKETSvoip + 1;
                else
                    LOSTPACKETSvoip = LOSTPACKETSvoip +1;
                end
                if QUEUEOCCUPATION > 0
                    EventList = [EventList; DEPARTURE, Clock + 8*QUEUE(1,1)/(C*10^6), QUEUE(1,1), QUEUE(1,2), QUEUE(1,3)];
                    QUEUEOCCUPATION= QUEUEOCCUPATION - QUEUE(1,1);
                    QUEUE(1,:)= [];
                else
                    STATE= 0;
                end
            end
    end
end

%Performance parameters determination:
PLdata= 100*LOSTPACKETS/TOTALPACKETS;  % in percentage
PLvoip = 100*LOSTPACKETSvoip/TOTALPACKETSvoip;
APDdata= 1000*DELAYS/TRANSPACKETS;     % in milliseconds
APDvoip = 1000*DELAYSvoip/TRANSPACKETSvoip;
MPDdata= 1000*MAXDELAY;                % in milliseconds
MPDvoip=1000*MAXDELAYvoip;
TT= 1e-6*TRANSBYTES*8/Clock;    % in Mbps

end
```

Figure 2.3: Sim3A - Part III

```
function out= GeneratePacketSize()
    aux= rand();
    aux2= [65:109 111:1517];
    if aux <= 0.19
        out= 64;
    elseif aux <= 0.19 + 0.23
        out= 110;
    elseif aux <= 0.19 + 0.23 + 0.17
        out= 1518;
    else
        out = aux2(randi(length(aux2)));
    end
end
```

Figure 2.4: Sim3A - Part IV

We introduced 2 changes to Sim3 in order to be able to do this exercise correctly. First we needed to add the bit error Rate (B) to our simulator input parameters(yellow highlight on Figure 2.1). We also needed a way to simulate lost packets in a random order so that the results would not be made up by us. In order to do that, in departure moments, we calculate the probability of having an error transmitting a packet bit.Then we generate a random number, if that number is higher than the probability of a packet being transmitted without any error, then we consider that occured some error in some bit and we consider that packet as lost and applied this method to either DATA packets and VOIP packets(highlighted in yellow on Figure 2.3)

2.1.2 Results and Conclusions

```
>> ex2a
For 10 VoIP flows:
PacketLoss of data(%)           = 4.73e+00 +- 3.16e-02
PacketLoss of VoIP(%)           = 9.35e-01 +- 1.75e-02
Av. Packet Delay of data (ms)   = 2.12e+00 +- 1.61e-02
Av. Packet Delay of VoIP (ms)   = 1.73e+00 +- 1.57e-02
Max. Packet Delay of data (ms)  = 1.80e+01 +- 7.89e-01
Max. Packet Delay of VoIP (ms)  = 1.73e+01 +- 8.40e-01
Throughput (Mbps)               = 7.25e+00 +- 1.12e-02

For 20 VoIP flows:
PacketLoss of data(%)           = 4.74e+00 +- 2.44e-02
PacketLoss of VoIP(%)           = 9.53e-01 +- 1.86e-02
Av. Packet Delay of data (ms)   = 2.64e+00 +- 4.28e-02
Av. Packet Delay of VoIP (ms)   = 2.23e+00 +- 4.09e-02
Max. Packet Delay of data (ms)  = 2.10e+01 +- 1.60e+00
Max. Packet Delay of VoIP (ms)  = 2.05e+01 +- 1.60e+00
Throughput (Mbps)               = 7.72e+00 +- 1.48e-02

For 30 VoIP flows:
PacketLoss of data(%)           = 4.75e+00 +- 3.04e-02
PacketLoss of VoIP(%)           = 9.52e-01 +- 1.65e-02
Av. Packet Delay of data (ms)   = 3.56e+00 +- 6.67e-02
Av. Packet Delay of VoIP (ms)   = 3.16e+00 +- 6.14e-02
Max. Packet Delay of data (ms)  = 2.60e+01 +- 1.50e+00
Max. Packet Delay of VoIP (ms)  = 2.59e+01 +- 1.52e+00
Throughput (Mbps)               = 8.20e+00 +- 1.33e-02

For 40 VoIP flows:
PacketLoss of data(%)           = 4.75e+00 +- 4.61e-02
PacketLoss of VoIP(%)           = 9.43e-01 +- 1.65e-02
Av. Packet Delay of data (ms)   = 5.83e+00 +- 2.82e-01
Av. Packet Delay of VoIP (ms)   = 5.41e+00 +- 2.73e-01
Max. Packet Delay of data (ms)  = 3.39e+01 +- 2.12e+00
Max. Packet Delay of VoIP (ms)  = 3.36e+01 +- 2.14e+00
Throughput (Mbps)               = 8.68e+00 +- 2.08e-02
```

Figure 2.5: Estimation of all parameters

To interpret the obtained results, we can analyze how the increase in the number of VoIP flows gradually impacts different performance parameters, focusing on Packet Loss, Average Packet Delay, Max Packet Delay, and Throughput in Mbps.

Packet Loss

As the number of VoIP flows increases from 10 to 40, the packet loss for data remains almost constant around 4.73% - 4.75%. The packet loss for VoIP varies slightly but also stays close to 0.94% - 0.95%, without a significant increase. This suggests that the system is well-adjusted to handle the extra load up to 40 VoIP flows, maintaining packet loss at constant values. The stability of the losses can be explained by the high capacity of the link (10 Mbps), which allows for effective accommodation of packets even with the increased demand for VoIP.

Average Packet Delay

The average packet delay for both data and VoIP shows an increasing trend with the number of VoIP flows. For 10 VoIP flows, the average packet delay for data is 2.12 ms, while for VoIP it is 1.73 ms. With 40 VoIP flows, these values increase to 5.83 ms (data) and 5.41 ms (VoIP). This rise in average delays occurs because, with more VoIP flows, the arrival of packets approaches the capacity of the link. As the system becomes more overloaded, packets experience longer waiting times before being transmitted, reflecting the impact of the increased load on overall performance.

Max Packet Delay

Similar to the average delay, the maximum packet delay for both data and VoIP also increases as the number of VoIP flows rises. For 10 VoIP flows, the maximum packet delay for data is 18.0 ms, while for VoIP it is 17.3 ms. With 40 flows, these values increase significantly to 33.9 ms (data) and 33.6 ms (VoIP). The rise in maximum delays is a result of increased competition for the link, where, during peak traffic times, packets must wait longer before being transmitted. This behavior is expected, as more flows lead to higher demand for the channel, which in turn increases the variation in waiting times.

Throughput

Throughput also increases with the rise in VoIP flows, going from 7.25 Mbps with 10 VoIP flows to 8.68 Mbps with 40 VoIP flows. This occurs because the number of packets transmitted increases as the VoIP load grows, which raises the utilization of the channel's capacity. This result is consistent with a system that is being progressively used more as more VoIP flows are added.

In conclusion, as we increase the number of VoIP flows, we observe a trend of stability in packet loss but a gradual increase in the average and maximum delays of both data and VoIP packets. Throughput also rises due to the increased utilization of the 10 Mbps link capacity. These results are consistent with a system that, while well-dimensioned, begins to show signs of overload in packet delays, especially as it approaches the link's capacity. If the number of VoIP flows continues to increase, it is likely that the average and maximum delays will continue to grow, eventually impacting packet loss as the system reaches or exceeds its total capacity.

2.2 Exercise 2.b)

2.2.1 Results and Conclusions

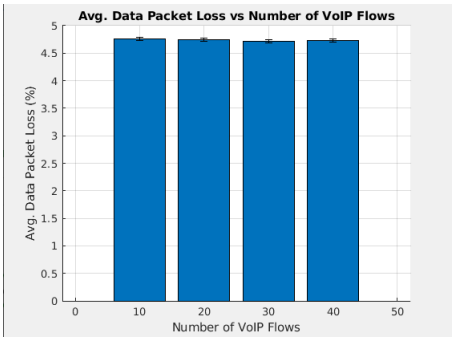

Figure 2.6: The average data packet loss results, $b = 10^{-5}$

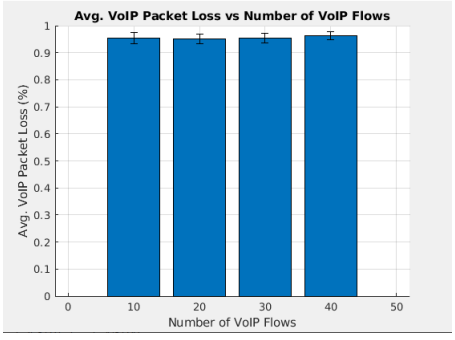

Figure 2.7: The average VoIP packet loss results, $b = 10^{-5}$

Analyzing Figures 2.6 and 2.7, we observe that data packets consistently experience a slightly higher average packet loss compared to VoIP packets. This difference aligns with packet size distributions: VoIP packets are smaller, with sizes evenly distributed between 110 and 130 bytes, while data packets vary widely from 64 to 1518 bytes, averaging around 620 bytes. Since larger packet sizes are generally more prone to loss, data packets naturally experience a higher packet loss rate than VoIP packets. Additionally, as the number of VoIP flows increases, the overall packet loss rate for both data and VoIP packets also rises due to the system nearing its link capacity.

2.3 Exercise 2.c)

2.3.1 Results and Conclusions

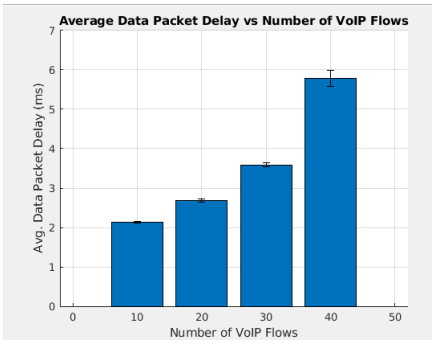

Figure 2.8: The average data packet delay results, $b = 10^{-5}$

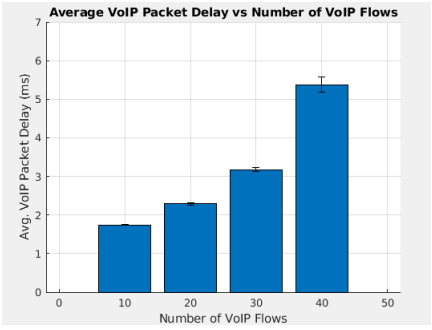

Figure 2.9: The average VoIP packet delay results, $b = 10^{-5}$

In Figures 2.8 and 2.9, we notice that data packets consistently have a slightly higher average packet delay than VoIP packets. This difference is expected, given that VoIP packet sizes are uniformly distributed between 110 and 130

bytes, whereas data packets range from 64 to 1518 bytes, averaging around 620 bytes. Larger packets generally experience more delay, which explains why data packets show a higher average delay than VoIP packets. Additionally, as the number of VoIP flows grows, the average packet delay for both data and VoIP packets increases exponentially. This happens because the rising number of arriving packets pushes the system closer to its link capacity.

2.4 Exercise 2.d)

2.4.1 Results and Conclusions

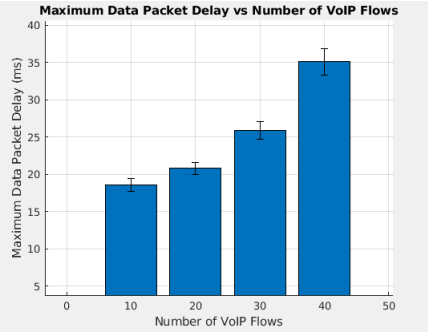

Figure 2.10: The maximum data packet delay results, $b = 10^{-5}$

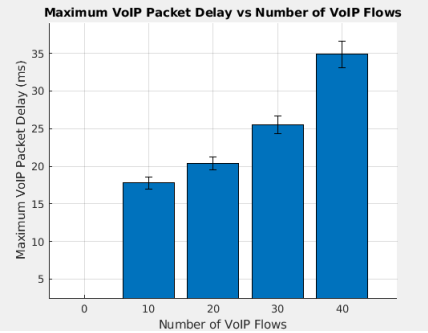

Figure 2.11: The maximum VoIP packet delay results, $b = 10^{-5}$

The analysis of the results reveals a clear trend: as the number of VoIP flows increases, the maximum delay for data and VoIP packets also increases significantly.

Increase in Maximum Delay

For 10 VoIP flows, the maximum delay is 18.0 ms for data and 17.5 ms for VoIP. However, when the number of VoIP flows increases to 40, these values rise dramatically to 39.7 ms for data and 39.6 ms for VoIP. This sharp increase in delays is indicative of the impact of the increased traffic load on the system.

Justification for the Delay

The increase in maximum delays can be attributed to the fact that, with a greater number of VoIP flows, the competition for the 10 Mbps link intensifies. The system approaches its maximum capacity, resulting in longer waiting times for packet transmission. During peak times, packets must wait longer before being processed, leading to greater delays.

Variation in the Reliability of the Results

The uncertainties ($\pm$) associated with the maximum delays also increase as the number of VoIP flows grows, suggesting greater variability in the system's performance under heavy load. The range of maximum delay widens, indicating that the consistency of latency may be affected as the number of flows increases.

In summary, the results show that while the system can handle an increasing number of VoIP flows, performance, measured by maximum delay, deteriorates as the load increases. This behavior is expected in systems approaching their capacity, highlighting the importance of monitoring and managing load in communication environments, especially concerning time-sensitive applications like VoIP. If the number of VoIP flows continues to rise, maximum delays are expected to increase further, which may eventually compromise service quality.

2.5 Exercise 2.e)

2.5.1 Results and Conclusions

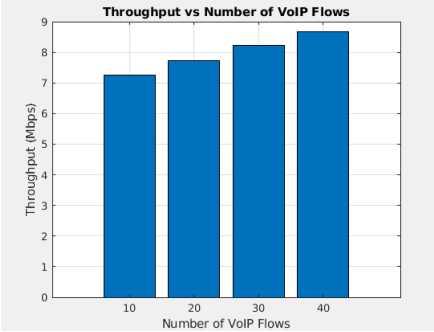

Figure 2.12: The total throughput results

The analysis of throughput results reveals a consistent growth trend as the number of VoIP flows increases. Below, we detail the observations:

Growth of Throughput

For 10 VoIP flows, the recorded throughput is 7.20 Mbps. As the number of flows increases to 20, this value rises to 7.64 Mbps. With 30 VoIP flows, the throughput reaches 8.25 Mbps, and with 40 flows, the value is 8.66 Mbps. This continuous increase in throughput suggests that the system can effectively utilize the 10 Mbps capacity of the link.

Justification for Throughput Increase

The growth in throughput is directly related to the higher number of packets transmitted. As more VoIP flows are added, the demand for packet transmission increases, leading to greater utilization of the channel's capacity. The system's efficiency in managing these packets results in a throughput that approaches the link's limit.

Stability of Throughput

Although the throughput increases with the number of VoIP flows, the fact that the variation associated with throughput is zero ($\pm$ 0.00) in all cases indicates stable system operation. This suggests that the system performs consistently under different VoIP loads, without significant fluctuations in the transmission rate.

In summary, the results demonstrate that the system scales well as the number of VoIP flows increases, maintaining a growing throughput. This adaptability is crucial in communication environments, as higher throughput allows for better utilization of the available bandwidth, ensuring a quality experience for VoIP users. However, it is important to monitor the system to ensure that as more flows are added, the quality of service is not compromised.

2.6 Exercise 2.f)

2.6.1 Code

```
lambda  = 1500;             % rate of arrival to the queue in pps
P = 100000;                 % Packets to be transmitted (stoping criteria)
C = 10;                     % Capacity of connection with 10Mbps
f = 1000000;                % Queue Size in Bytes
N = 20;                     % Times to run the simulation
n = [10 20 30 40];          % nr VoIP packets flows
B = 10^-5;                  % Bit error rate
alfa = 0.1;                 % 90% confidence interval for results

p64 = 0.19;
p110 = 0.23;
p1518 = 0.17;
p_other = 1 - (p64 + p110 + p1518);

%average packet size
PSize_average = p64 * 64 + p110 * 110 + p1518 * 1518 + p_other * mean(65:1517);

T_throughput = (lambda * PSize_average * 8) / 10^6; % Mbps


fprintf('Theoretical throughput: %.2f Mbps\n', T_throughput);
```

Figure 2.13: exercise 2-f

2.6.2 Results and Conclusions

We have different packet sizes and each one have a probability associated, so we need to calculate the average packet size.

We know that Thoughput is the packet arrival rate X average packet size. The packet arrival rate is represented by 'lambda' in packet per second and the average packet size is represented by the 'PSize$_{a}verage'$.

With this information, we obtain the throughput in bytes per second, so we need to multiply by 8 **to convert it to bits per second. Throughput is represented in Mbps, so we need to divide the result by** 10^6 **to convert it, because** $1\,\text{Mbps} = 1,000,000\,\text{bps}$.

Chapter 3

Task 3

3.1 Exercise 3.a)

3.1.1 Results and Conclusions

We can see 3 things by looking at the results of the figure 3.1 (below) :

- Average Delay on data packets is higher than the average delay on voip packets.
  - From analyzing the code we can see that Voip packets size vary between 110 and 130 bytes and data packets size can vary between 64 bytes(19% probability), 110(23% probability), 1518 bytes (17% probability) and can have other possibles ranges from 65 to 109 and 111 to 1517(all with the same probability). From that we can see that a significant portion of data packets can have higher size which can translate in a higher packet delay. Those values also increase as the voip packet flows also increases due to more congestion on the queue, knowing that the queue size didn't change.
- Data packets packet loss is higher than voip packets packet loss and also tends to enlarge that gap as the voip flows increase.
  - This can happen due to how crowded the queue is. If the queue is nearly full and a packet arrives, the probability that it is greater than the free space in the queue is higher when it is a data packet . When this happens, the packet is dropped.
- Data packets packet loss tends to enlarge that gap(talked on the point above) as the voip flows increase.
  - If the voip packet flows increase then the probability of having a free space where it fits a data packet and not a voip packet tends to be even lower, explaining the increase in the packet loss of data packets.

– The queue can also be filled with voip packets , and the free space of the queue can lower than a voip packet. As a consequence of that the packet loss for the voip packets als increase but is a much slower rate.

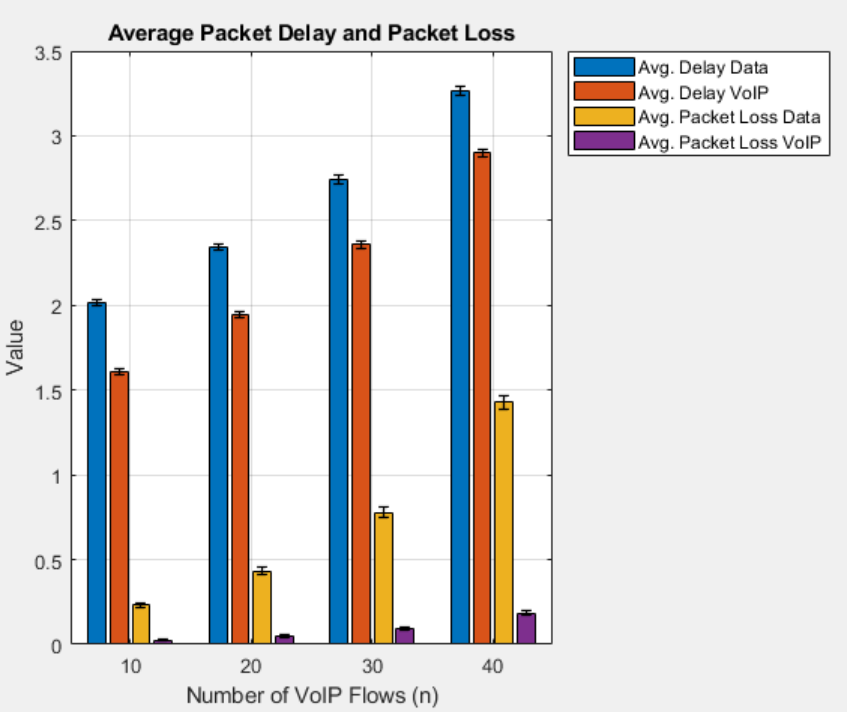

Figure 3.1: Packet Loss and Packet delay for data and voip packets

3.2 Exercise 3.b)

3.2.1 Results and Conclusions

On this exercise we use Sim4 that includes a priority for voip packets. what can we see:

- Average Delay on data packets is higher than exercise 3-a).
  - If voip packets have priority they are the ones that leave the queue earlier meaning the data packets that entered the queue are obligated to stay and wait that voips are sent.
- Average Delay on voip packets increases slightly as the number of voip flow increases too
  - Having priority inside the queue, they are the first ones to leave. But as the number of flows increase, the number of voip packets on the queue increases which leads to higher delay.
- Data packets packet loss is almost identical to exercise 3-a)
  - The probability of a data packet entering the queue is the same as in the exercise before. Despite that, it can take longer to leave the queue.

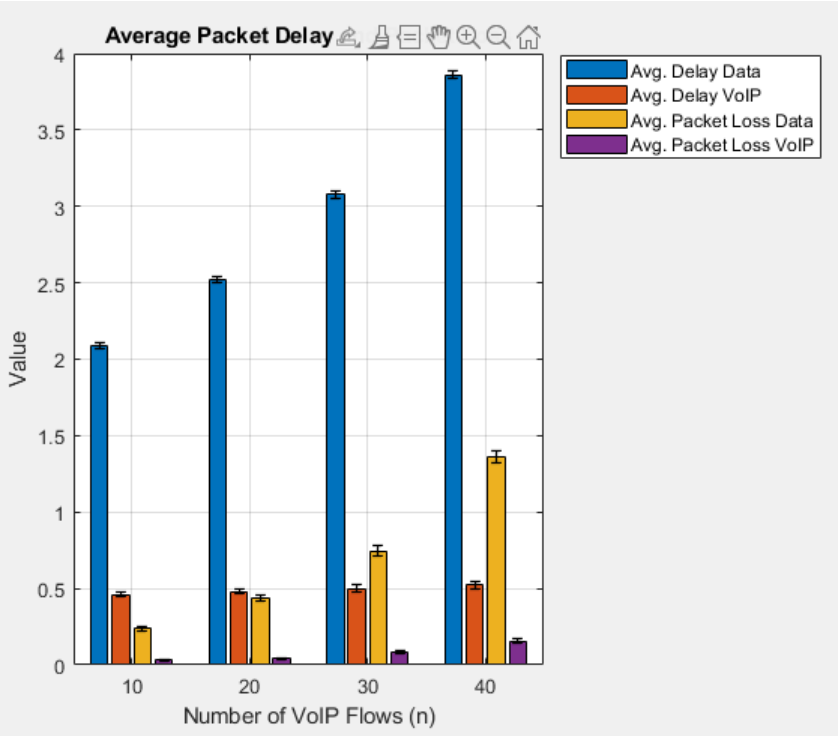

Figure 3.2: Packet Loss and Packet delay for data and voip packets

3.3 Exercise 3.c)

3.3.1 Sim4A Code

In this exercise we had to change Sim4 in order to respect the following statement : 'the following arriving VoIP packets are always accepted in the queue (if there is enough space) but arriving data packets are accepted in the queue only if the total queue occupation does not become higher than p (in %) of the queue size2'.

In order to do that, we needed to check on the packet arrival moment what kind of packet had just arrived. If the it was a voip packet it entered the queue(if there was any space left), but if it was a data packet we needed to check the queue occupation and what percentage of it was full. In order to do that we multiplied the queue size by the percentage of occupation asked, divided by 100 to remove the percentage and get the value. This can be seen on Figure 3.4 highlighted in yellow.

```
function [PLdata, PLvoip, APDdata, APDvoip, MPDdata, MPDvoip, TT] = Sim4A(lambda, C, f, P, n,p)
% INPUT PARAMETERS:
%  lambda - packet rate (packets/sec)
%  C      - link bandwidth (Mbps)
%  f      - queue size (Bytes)
%  P      - number of packets (stopping criterium)
%  n      - number of initial VoIP packets
%  p      - queue fill percentage
% OUTPUT PARAMETERS:
%  PLdata   - packet loss (%) of data packets
%  PLvoip  - Packet Loss (%) of Voip packets
%  APDdata  - average  data packet delay (milliseconds)
%  APDvoip  - average Voip packet delay (milliseconds)
%  MPDdata  - maximum data packet delay (milliseconds)
%  MPDvoip  - maximum Voip packet delay (milliseconds)
%  TT   - transmitted throughput (Mbps)

% Events:
ARRIVAL = 0;      % Arrival of a packet
DEPARTURE = 1;    % Departure of a packet

% State variables:
STATE = 0;           % 0 - connection is free; 1 - connection is occupied
QUEUEOCCUPATION = 0; % Occupation of the queue (in Bytes)
QUEUE = [];          % Size, arriving time instant, and type of each packet in the queue

% Statistical Counters:
TOTALPACKETS = 0;        % No. of packets arrived to the system
TOTALPACKETSvoip = 0;
LOSTPACKETS = 0;         % No. of packets dropped due to buffer overflow
LOSTPACKETSvoip = 0;
TRANSPACKETS = 0;        % No. of transmitted data packets
TRANSPACKETSvoip = 0;
TRANSBYTES = 0;          % Sum of the Bytes of transmitted packets

DELAYS = 0;              % Sum of the delays of transmitted data packets
DELAYSvoip = 0;
MAXDELAY = 0;            % Maximum delay among all transmitted data packets
MAXDELAYvoip = 0;

% Initializing the simulation clock:
Clock = 0;

% Initializing the List of Events with the first data packet ARRIVAL:
tmp = Clock + exprnd(1/lambda);
EventList = [ARRIVAL, tmp, GeneratePacketSize(), tmp, 0];
```

Figure 3.3: Sim4A - Part I

```
% Simulation loop:
while (TRANSPACKETS + TRANSPACKETSvoip) < P % Stopping criterion
    EventList = sortrows(EventList, 2);  % Order EventList by time
    Event = EventList(1, 1);              % Get first event
    Clock = EventList(1, 2);              % Current time
    PacketSize = EventList(1, 3);         % Packet size
    ArrInstant = EventList(1, 4);         % Arrival time
    type = EventList(1, 5);               % Packet type (0 for data, 1 for VoIP)

    EventList(1, :) = [];                 % Remove the processed event

    switch Event
        case ARRIVAL                     % If the event is an ARRIVAL
            if type == 0  % Data packet
                TOTALPACKETS = TOTALPACKETS + 1;
                tmp = Clock + exprnd(1/lambda);
                EventList = [EventList; ARRIVAL, tmp, GeneratePacketSize(), tmp, 0];
                if STATE == 0
                    STATE = 1;
                    EventList = [EventList; DEPARTURE, Clock + 8 * PacketSize / (C * 10^6), PacketSize, Clock, 0];
                else
                    if QUEUEOCCUPATION + PacketSize <= f*(p/100) % changes here, total queue ocupation <= queue size * p(%)/100
                        QUEUE = [QUEUE; PacketSize, Clock, 0]; % Add data packet to the queue
                        QUEUEOCCUPATION = QUEUEOCCUPATION + PacketSize;
                    else
                        LOSTPACKETS = LOSTPACKETS + 1;  % Data packet lost due to buffer overflow
                    end
                end
            elseif type == 1 % VoIP packet
                TOTALPACKETSvoip = TOTALPACKETSvoip + 1;
                tmp = Clock + unifrnd(0.016, 0.024);
                EventList = [EventList; ARRIVAL, tmp, randi([110, 130]), tmp, 1];
                if STATE == 0
                    STATE = 1;
                    EventList = [EventList; DEPARTURE, Clock + 8 * PacketSize / (C * 10^6), PacketSize, Clock, 1];
                else
                    if QUEUEOCCUPATION + PacketSize <= f
                        QUEUE = [QUEUE; PacketSize, Clock, 1]; % Add VoIP packet to the queue
                        QUEUEOCCUPATION = QUEUEOCCUPATION + PacketSize;
                    else
                        LOSTPACKETSvoip = LOSTPACKETSvoip + 1; % VoIP packet lost due to buffer overflow
                    end
                end
            end
```

Figure 3.4: Sim4A - Part II

```
        case DEPARTURE                    % If the event is a DEPARTURE
            if type == 0 % Data packet
                TRANSBYTES = TRANSBYTES + PacketSize;
                DELAYS = DELAYS + (Clock - ArrInstant);
                if Clock - ArrInstant > MAXDELAY
                    MAXDELAY = Clock - ArrInstant;
                end
                TRANSPACKETS = TRANSPACKETS + 1;
            elseif type == 1 % VoIP packet
                TRANSBYTES = TRANSBYTES + PacketSize;
                DELAYSvoip = DELAYSvoip + (Clock - ArrInstant);
                if Clock - ArrInstant > MAXDELAYvoip
                    MAXDELAYvoip = Clock - ArrInstant;
                end
                TRANSPACKETSvoip = TRANSPACKETSvoip + 1;
            end

            % Serve the next packet in the queue, prioritizing VoIP
            if QUEUEOCCUPATION > 0
                % changes here
                QUEUE = sortrows(QUEUE, -3);% orders the queue in descending order(-) y the packet type(column number 3)
                EventList = [EventList; DEPARTURE, Clock + 8*QUEUE(1,1)/(C*10^6), QUEUE(1,1), QUEUE(1,2), QUEUE(1,3)];
                QUEUEOCCUPATION= QUEUEOCCUPATION - QUEUE(1,1);
                QUEUE(1,:)= [];
            else
                STATE = 0;
            end
        end
end
```

Figure 3.5: Sim4A - Part III

```
function out = GeneratePacketSize()
    aux = rand();
    aux2 = [65:109 111:1517];
    if aux <= 0.19
        out = 64;
    elseif aux <= 0.19 + 0.23
        out = 110;
    elseif aux <= 0.19 + 0.23 + 0.17
        out = 1518;
    else
        out = aux2(randi(length(aux2)));
    end
end
```

Figure 3.6: Sim4A - Part IV

In Figure 3.5 we can see the prioritization of voip packets. In order to achieve that we ordered the packet Queue in descending order(represented by the '-' signal) of the packet type which is represented on the column nº3 of the Queue(data packet are identified by the number 0 and voip packets are identified by the number 1).

3.4 Exercise 3.d)

3.4.1 Results and Conclusions

We can see that:

- Packet loss for data packet increases gradually as the voip flows increase
  - They are likely to be dropped when the queue is on 90% occupation as it is harder for packets to enter the queue when the voip packet flows increase.
- Packet loss of voip packets is 0 or close to 0 independently of the number of flows
  - Voip packets are prioritized in relation to data packets and because of their likelihood to be smaller they can fit better and in more quantity inside the queue. So the results are what we expected.

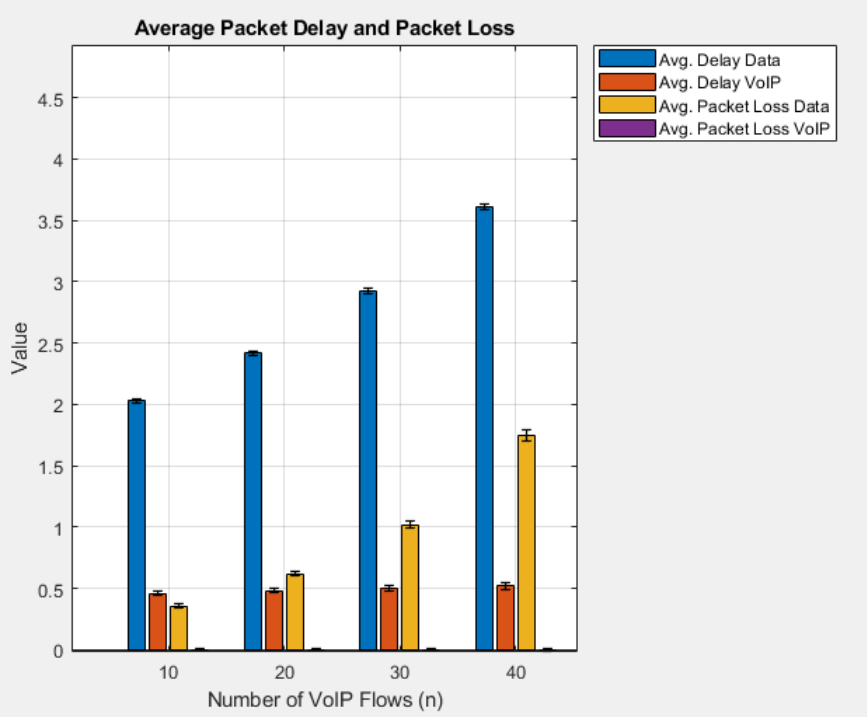

Figure 3.7: Packet Loss and Packet delay for data and voip packets

3.5 Exercise 3.e)

3.5.1 Results and Conclusions

We can see that:

- Average Delay is higher than ex3-d).
  - this happens because if the occupation percentage for data packets to be accepted decreases then the queue fills up faster(more packets, probably bigger packets).
- Packet loss for data packet increases gradually as the voip flows increase
  - They are likely to be dropped when the queue is on 60% occupation as the queue fills up faster and the data packet acceptance probability is much higher than the exercise 3-d).
- Packet loss of voip packets is 0 independently of the number of flows
  - Voip packets are prioritized and accepted in the queue(if there's enough space), meaning that even if the queue is near capacity voip packets have precedence over data packets.

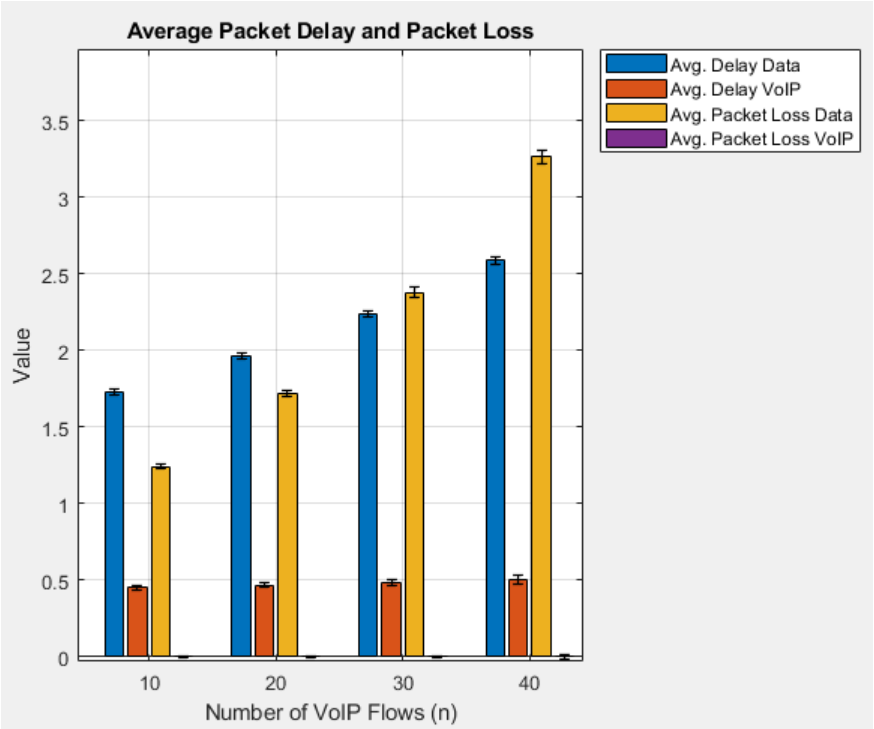

Figure 3.8: Packet Loss and Packet delay for data and voip packets

Our Contributions are:

- João Ferreira (103625) - 50%
- Rafael Santos (98466) - 50%