Universidade de Aveiro

# Modelação e Desempenho de Redes e Seviços

## Mini-Project nr.1

André Clérigo (98485), Pedro Rocha (98256)

Departamento de Eletrónica, Telecomunicações e Informática

November 8, 2022

# Contents

# List of Figures

# Chapter 1

# Task 1

## 1.1 Exercise 1.a)

### 1.1.1 Code

```
rates = [1500 1600 1700 1800 1900]; % rate of arrival to ...
    the queue in pps
P = 100000;     % Packets to be transmitted (stopping criteria)
C = 10;         % Capacity of connection with 10Mbps
f = 1000000;    % Queue Size in Bytes
N = 20;         % Times to run the simulation
b = 10^-6;      % Bit error rate
alfa = 0.1;     % 90% confidence interval for results

% Variables to store bar graph data
APD_values = zeros(1, length(rates));
APD_terms = zeros(1, length(rates));
PL_values = zeros(1, length(rates));
PL_terms = zeros(1, length(rates));

% Variables to store the simulation results
PL = zeros(1, N);
APD = zeros(1, N);
MPD = zeros(1, N);
TT = zeros(1, N);

% Test all rates required
for i = 1:length(rates)
    % Run the simulator N times
    for it = 1:N
            [PL(it), APD(it), MPD(it), TT(it)] = ...
                Simulator2(rates(i), C, f, P, b);
    end

    % Calculate Avg. Packet Delay
    media = mean(APD);
    term = norminv(1-alfa/2)*sqrt(var(APD)/N);
    APD_values(i) = media;
    APD_terms(i) = term;

    % Calculate Avg. Packet Loss
    media = mean(PL);
    term = norminv(1-alfa/2)*sqrt(var(PL)/N);
    PL_values(i) = media;
    PL_terms(i) = term;
end

% Figure to show the asked results for 1.a) i)
figure(1);
hold on;
grid on;
bar(rates, APD_values');  % Bar Graph
ylim([0 9]);              % Set y axis values between 0 and 9
er = errorbar(rates, APD_values', APD_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Packet Rate (pps)')
ylabel('Avg. Packet Delay (ms)')
title('Average Packet Delay vs Packet Rate');
hold off;

% Figure to show the asked results for 1.b) i)
figure(2);
hold on;
grid on;
bar(rates, PL_values');    % Bar Graph
er = errorbar(rates, PL_values', PL_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Packet Rate (pps)')
ylabel('Packet Loss (%)')
title('Packet Loss vs Packet Rate');
hold off;
```

### 1.1.2 Results and Conclusions

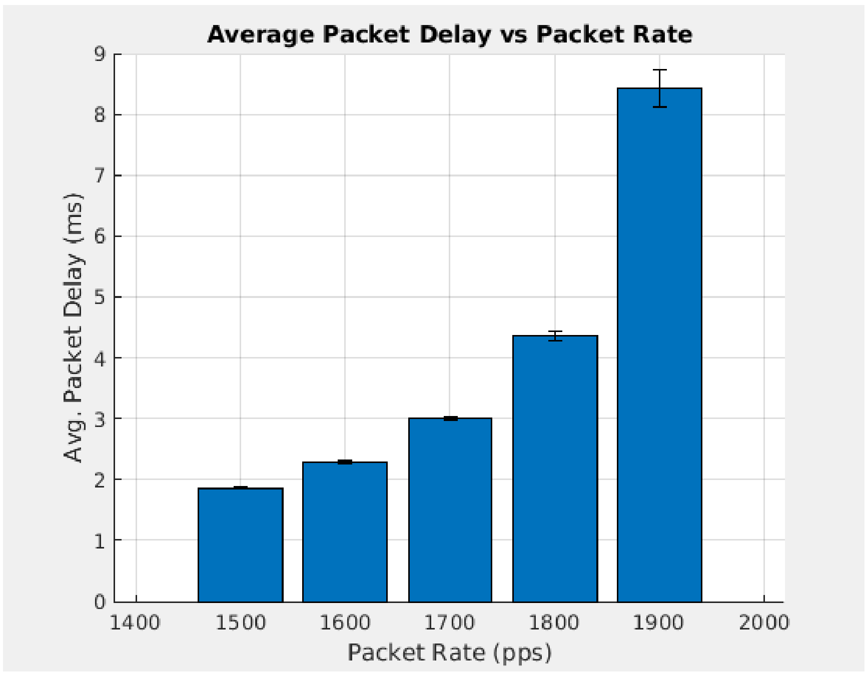

Figure 1.1: The average packet delay results, b = 1e-6

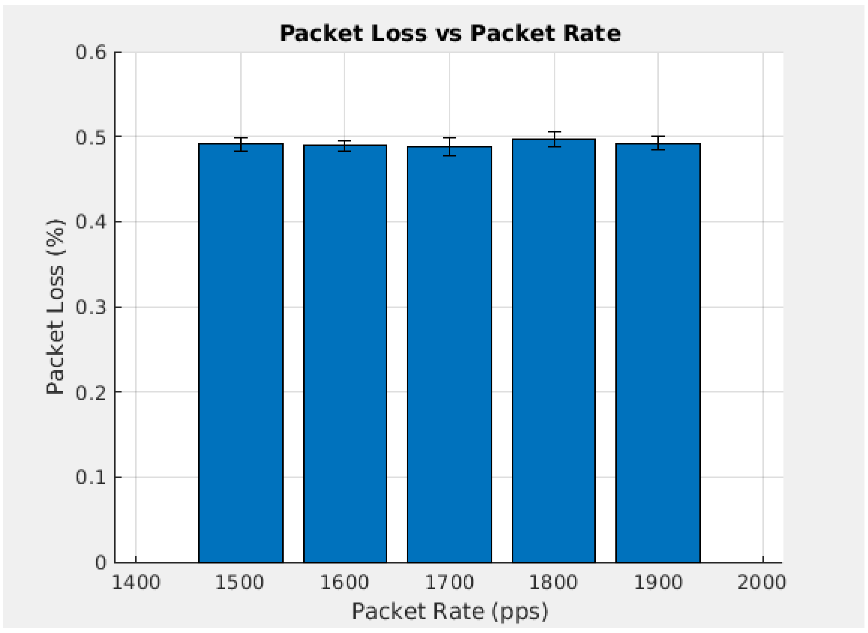

Figure 1.2: The average packet loss results, b = 1e-6

When increasing the packet rate on a system, it is expected that the average packet delay will increase. From the lab classes we know that the average packet size generated by the simulator is $\approx$ 620 bytes, with that information and the specifications given in the enunciate we can derive that:

| Packet Size (Bytes) | C (Mbps) | C (MBps) | Queue Size (MB) |
|---|---|---|---|
| 620 | 10 | 1.25 | 1 |

Table 1.1: Values derived by the enunciate

| Packet Rate (pps) | Arrival Packets (MBps) | Dropped Packets |
|---|---|---|
| 1500 | 0.93 | 0 |
| 1600 | 0.99 | 0 |
| 1700 | 1.05 | 0 |
| 1800 | 1.12 | 0 |
| 1900 | 1.18 | 0 |

Table 1.2: Calculated values

Analysing the results gotten we can see that on Figure 1.1 the average packet delay increases as the packet rate gets higher, and on Figure 1.2 we can see that the packet loss value doesn't change significantly with the packet rate variation.

The average packet delay increases exponentially because the rate of packet arrival is approaching the system's capacity (as seen in Table 1.1 and in Table 1.2). The value of packet loss is constant, or within margin of error, because the packet arrival rate never hits the system's capacity and even if that were to happen the queue size (1MB) is enough to create a good buffer, therefore, all packets can be processed in time. The value of packet loss got can be explained by the constant bit error rate in our system's connection.

## 1.2 Exercise 1.b)

### 1.2.1 Code

```
rates = [1500 1600 1700 1800 1900]; % rate of arrival to ...
    the queue in pps
P = 100000;     % Packets to be transmitted (stopping criteria)
C = 10;         % Capacity of connection with 10Mbps
f = 1000000;    % Queue Size in Bytes
N = 20;         % Times to run the simulation
b = 10^-4;      % Bit error rate
alfa = 0.1;     % 90% confidence interval for results

% Variables to store bar graph data
APD_values = zeros(1, length(rates));
APD_terms = zeros(1, length(rates));
PL_values = zeros(1, length(rates));
PL_terms = zeros(1, length(rates));

% Variables to store the simulation results
PL = zeros(1, N);
APD = zeros(1, N);
MPD = zeros(1, N);
TT = zeros(1, N);

% Test all rates required
for i = 1:length(rates)
    % Run the simulator N times
    for it = 1:N
            [PL(it), APD(it), MPD(it), TT(it)] = ...
                Simulator2(rates(i), C, f, P, b);
    end

    % Calculate Avg. Packet Delay
    media = mean(APD);
    term = norminv(1-alfa/2)*sqrt(var(APD)/N);
    APD_values(i) = media;
    APD_terms(i) = term;

    % Calculate Avg. Packet Loss
    media = mean(PL);
    term = norminv(1-alfa/2)*sqrt(var(PL)/N);
    PL_values(i) = media;
    PL_terms(i) = term;
end

% Figure to show the asked results for 1.a) i)
figure(1);
hold on;
grid on;
bar(rates, APD_values');  % Bar Graph
ylim([0 9]);              % Set y axis values between 0 and 9
```

```
er = errorbar(rates, APD_values', APD_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Packet Rate (pps)')
ylabel('Avg. Packet Delay (ms)')
title('Average Packet Delay vs Packet Rate');
hold off;

% Figure to show the asked results for 1.b) i)
figure(2);
hold on;
grid on;
bar(rates, PL_values');     % Bar Graph
er = errorbar(rates, PL_values', PL_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Packet Rate (pps)')
ylabel('Packet Loss (%)')
title('Packet Loss vs Packet Rate');
hold off;
```

### 1.2.2 Results and Conclusions

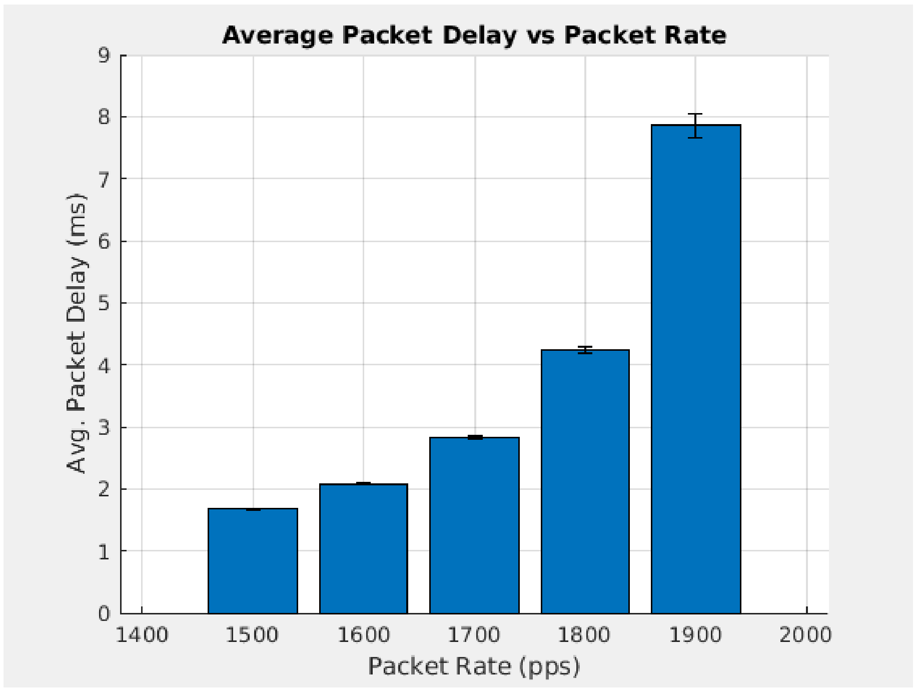

Figure 1.3: The average packet delay results, b = 1e-4

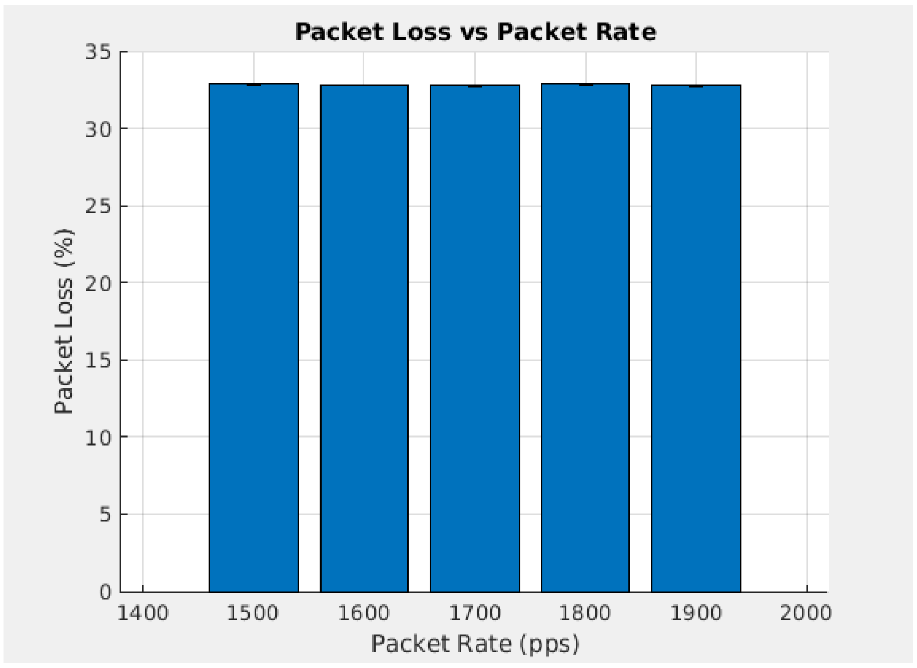

Figure 1.4: The average packet loss results, b = 1e-4

When increasing the bit error rate by a factor of 100x, it is expected that the packet loss will increase significantly.

Analysing Figure 1.3 and Figure 1.4 we see that the packet loss increased from ≈ 0.5% to ≈ 33%, and that this value stays constant with the increasing of packet rate. We can also see that the average packet delay did not change significantly with the changing of bit error rate.

The increasing of packet loss is to be expected because with higher bit error rate comes a higher packet loss. The maintaining of packet delay can be explained by what was described in the previous Section 1.1.2, since the bit error rate does not change the link capacity nor the packet arrival rate, the average delay values are expected to be similar.

## 1.3 Exercise 1.c)

### 1.3.1 Code

```
% Equal probability for Data Packets with a size different ...
    of 64, 110 and 1518 Bytes
prob_left = (1 - (0.19 + 0.23 + 0.17)) / ((109 - 65 + 1) + ...
    (1517 - 111 + 1));
b = [1e-6 1e-4];                % Values of ber to test
ploss_ber = [0 0];              % Probabilities of packet loss

```

```
% Test the ber values
for i = 1 : length(b)
    % Calculate the prob. of having at least 1 error for ...
        the given packet size
    for packetSize = 64 : 1518
        if (packetSize == 64)
            ploss_ber(i) = ploss_ber(i) + (1 ...
                -(nchoosek(packetSize*8, 0) * b(i)^0 * ...
                (1-b(i))^(packetSize*8))) * 0.19;
        elseif (packetSize == 110)
            ploss_ber(i) = ploss_ber(i) + (1 ...
                -(nchoosek(packetSize*8, 0) * b(i)^0 * ...
                (1-b(i))^(packetSize*8))) * 0.23;
        elseif (packetSize == 1518)
            ploss_ber(i) = ploss_ber(i) + (1 ...
                -(nchoosek(packetSize*8, 0) * b(i)^0 * ...
                (1-b(i))^(packetSize*8))) * 0.17;
        else
            ploss_ber(i) = ploss_ber(i) + (1 ...
                -(nchoosek(packetSize*8, 0) * b(i)^0 * ...
                (1-b(i))^(packetSize*8))) * prob_left;
        end
    end
end

ploss_ber = (ploss_ber ./ (0.19 + 0.23 + 0.17 + ((109 - 65 ...
    + 1) + (1517 - 111 + 1)) * prob_left)) * 100;

fprintf('Considering that the bit error rate is the only ...
    factor that can cause Packet to be lost, the ...
    theoretical packet loss (%%)\nfor a bit error rate of ...
    10^-6 is: %.4f %%\nand for a bit error rate of 10^-4 ...
    is: %.4f %%\n', ploss_ber(1), ploss_ber(2));
```

### 1.3.2 Results and Conclusions

The program's ouput was the following: "Considering that the bit error rate is the only factor that can cause Packet to be lost, 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 %".

The results got, let us believe that the packet loss experienced in exercise 1.a) and 1.b) are only due to the bit error rate, as the theoretical values are similar to the results got in our experiences. This results further confirm our hypothesis that packet loss is only due to bit error rate as the system can process all packets in time.

# Chapter 2

# Task 2

## 2.1 Approach

In Simulator3A we need to add the bit error rate function that is implemented on Simulator2 but apply it on Simulator3. Therefore we only change the code that processes the Departure packets as the code bellow shows.

```
function [PLd, PLv, APDd, APDv, MPDd, MPDv, TT] = ...
    Simulator3A(lambda,C,f,P,n,b)
%Statistical Counters:
TOTALPACKETSD = 0; % No of data packets arrived
TOTALPACKETSV = 0; % No of voip packets arrived
LOSTPACKETSD = 0;  % No of data packets dropped due to overflow
LOSTPACKETSV = 0;  % No of voip packets dropped due to overflow
TRANSMITTEDPACKETSD = 0; % No. of transmitted data packets
TRANSMITTEDPACKETSV = 0; % No. of transmitted voip packets
TRANSMITTEDBYTESD = 0; % Sum Bytes of transmitted data packets
TRANSMITTEDBYTESV = 0; % Sum Bytes of transmitted voip packets
DELAYSD = 0;    % Sum of delays of transmitted data packets
DELAYSV = 0;    % Sum of delays of transmitted voip packets
MAXDELAYD = 0;  % Maximum delay in all transmitted data packets
MAXDELAYV = 0;  % Maximum delay in all transmitted voip packets
...
case DEPARTURE        % If first event is a DEPARTURE
    % nchoosek(PacketSize*8, 0) * b^0 * (1-b)^(PacketSize*8))
    if (rand() < (1-b)^(PacketSize*8))
        if (PacketType == DATA)         % Data Packet
            TRANSMITTEDBYTESD= TRANSMITTEDBYTESD + PacketSize;
            DELAYSD = DELAYSD + (Clock - ArrivalInstant);
            if Clock - ArrivalInstant > MAXDELAYD
                MAXDELAYD= Clock - ArrivalInstant;
            end
            TRANSMITTEDPACKETSD= TRANSMITTEDPACKETSD + 1;
        else                            % VoIP Packet
            TRANSMITTEDBYTESV= TRANSMITTEDBYTESV + PacketSize;
            DELAYSV= DELAYSV + (Clock - ArrivalInstant);
            if Clock - ArrivalInstant > MAXDELAYV
                MAXDELAYV= Clock - ArrivalInstant;
            end
            TRANSMITTEDPACKETSV= TRANSMITTEDPACKETSV + 1;
        end
    else
        if (PacketType == DATA)          % Data Packet
            LOSTPACKETSD = LOSTPACKETSD + 1;
        else                             % VoIP Packet
            LOSTPACKETSV = LOSTPACKETSV + 1;
        end
    end

    if QUEUEOCCUPATION > 0
        EventList = [EventList; DEPARTURE, Clock + ...
            8*QUEUE(1,1)/(C*10^6), QUEUE(1,1), QUEUE(1,2), ...
            QUEUE(1,3)]; % Add PacketType
        QUEUEOCCUPATION= QUEUEOCCUPATION - QUEUE(1,1);
        QUEUE(1,:)= [];
    else
        STATE= 0;
    end
...
%Performance parameters determination:
PLd = 100*LOSTPACKETSD/TOTALPACKETSD;       % in %
PLv = 100*LOSTPACKETSV/TOTALPACKETSV;       % in %
APDd = 1000*DELAYSD/TRANSMITTEDPACKETSD;    % in milliseconds
APDv = 1000*DELAYSV/TRANSMITTEDPACKETSV;    % in milliseconds
MPDd = 1000*MAXDELAYD;                      % in milliseconds
MPDv = 1000*MAXDELAYV;                      % in milliseconds
TT = 10^(-6)*(TRANSMITTEDBYTESD+TRANSMITTEDBYTESV)*8/Clock; ...
    % in Mbps
...
```

## 2.2 Exercise 2.a)

### 2.2.1 Code

```
rate = 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
nVoips = [10 20 30 40];      % nr voip packets flows
b = 10^-5;      % Bit error rate
alfa = 0.1;     % 90% confidence interval for results

% Variables to store bar graph data
PLd_values = zeros(1, length(nVoips));
PLd_terms = zeros(1, length(nVoips));
```

```
PLv_values = zeros(1, length(nVoips));
PLv_terms = zeros(1, length(nVoips));
APDd_values = zeros(1, length(nVoips));
APDd_terms = zeros(1, length(nVoips));
APDv_values = zeros(1, length(nVoips));
APDv_terms = zeros(1, length(nVoips));

% Variables to store the simulation results
PLd = zeros(1, N);
PLv = zeros(1, N);
APDd = zeros(1, N);
APDv = zeros(1, N);
MPDd = zeros(1, N);
MPDv = zeros(1, N);
TT = zeros(1, N);

% Test all number of VoIP flows
for i = 1 : length(nVoips)
    % Run the simulator N times
    for it = 1:N
        [PLd(it), PLv(it), APDd(it), APDv(it), MPDd(it), ...
            MPDv(it), TT(it)] = Simulator3A(rate, C, f, P, ...
            nVoips(i), b);
    end

    % Calculate Avg. Packet Loss for Data Packets
    media = mean(PLd);
    term = norminv(1-alfa/2)*sqrt(var(PLd)/N);
    PLd_values(i) = media;
    PLd_terms(i) = term;

    % Calculate Avg. Packet Loss for VoIP Packets
    media = mean(PLv);
    term = norminv(1-alfa/2)*sqrt(var(PLv)/N);
    PLv_values(i) = media;
    PLv_terms(i) = term;

    % Calculate Avg. Packet Delay for Data Packets
    media = mean(APDd);
    term = norminv(1-alfa/2)*sqrt(var(APDd)/N);
    APDd_values(i) = media;
    APDd_terms(i) = term;

    % Calculate Avg. Packet Delay for VoIP Packets
    media = mean(APDv);
    term = norminv(1-alfa/2)*sqrt(var(APDv)/N);
    APDv_values(i) = media;
    APDv_terms(i) = term;
end

% Figure to show the asked results for 2.a) i)
figure(1);
hold on;
grid on;
```

```
bar(nVoips, APDd_values');% Bar Graph
ylim([0 7]);                % Set y axis values between 0 and 7
er = errorbar(nVoips, APDd_values', APDd_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Data Packet Delay (ms)')
title('Average Data Packet Delay vs Number of VoIP Flows');
hold off;

% Figure to show the asked results for 2.a) ii)
figure(2);
hold on;
grid on;
bar(nVoips, PLd_values'); % Bar Graph
er = errorbar(nVoips, PLd_values', PLd_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Data Packet Loss (%)')
title('Avg. Data Packet Loss vs Number of VoIP Flows');
hold off;

% Figure to show the asked results for 2.a) iii)
figure(3);
hold on;
grid on;
bar(nVoips, APDv_values');% Bar Graph
ylim([0 7]);                % Set y axis values between 0 and 7
er = errorbar(nVoips, APDv_values', APDv_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. VoIP Packet Delay (ms)')
title('Average VoIP Packet Delay vs Number of VoIP Flows');
hold off;

% Figure to show the asked results for 2.a) iv)
figure(4);
hold on;
grid on;
bar(nVoips, PLv_values'); % Bar Graph
er = errorbar(nVoips, PLv_values', PLv_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. VoIP Packet Loss (%)')
title('Avg. VoIP Packet Loss vs Number of VoIP Flows');
hold off;
```

### 2.2.2 Results and Conclusions

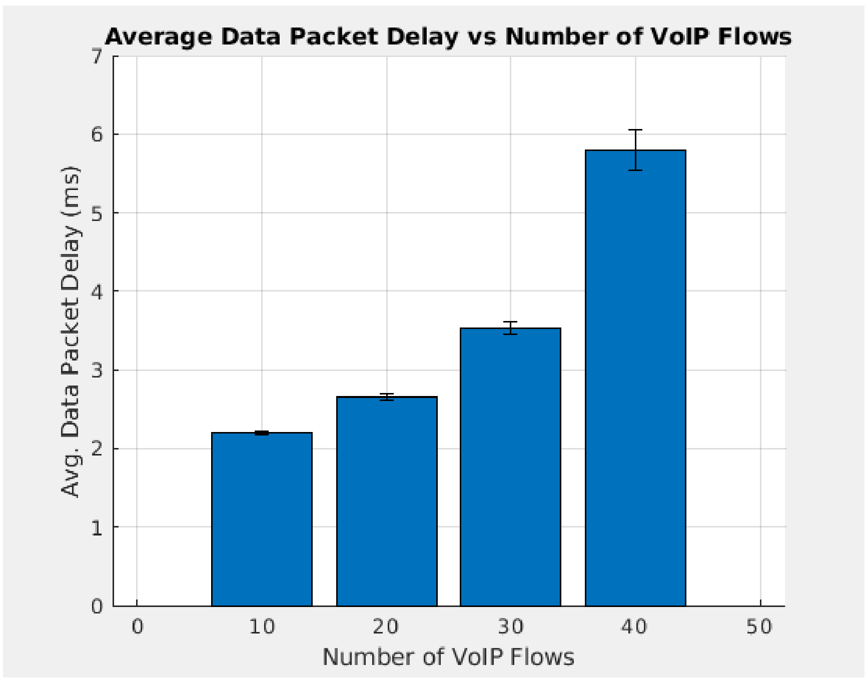

Figure 2.1: The average data packet delay results, b = 1e-5

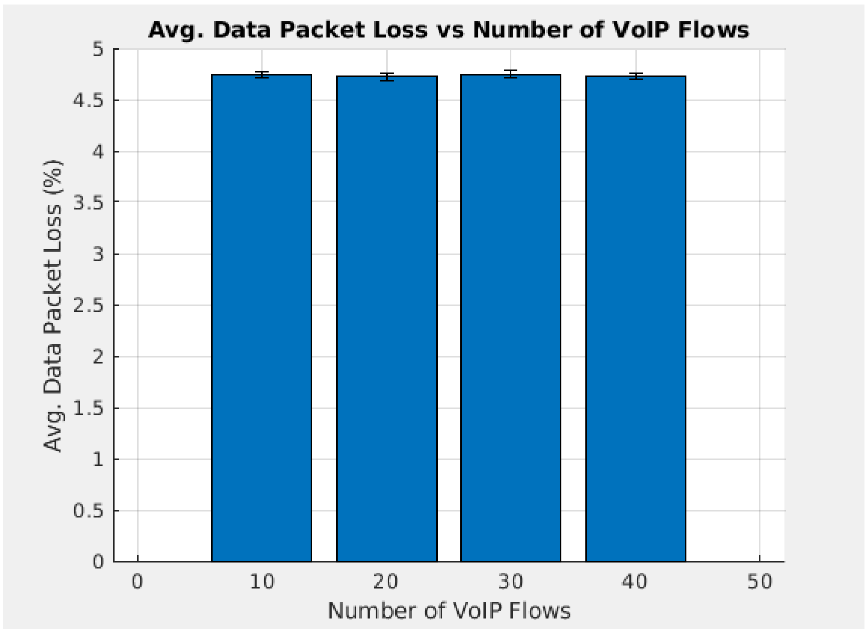

Figure 2.2: The average data packet loss results, b = 1e-5

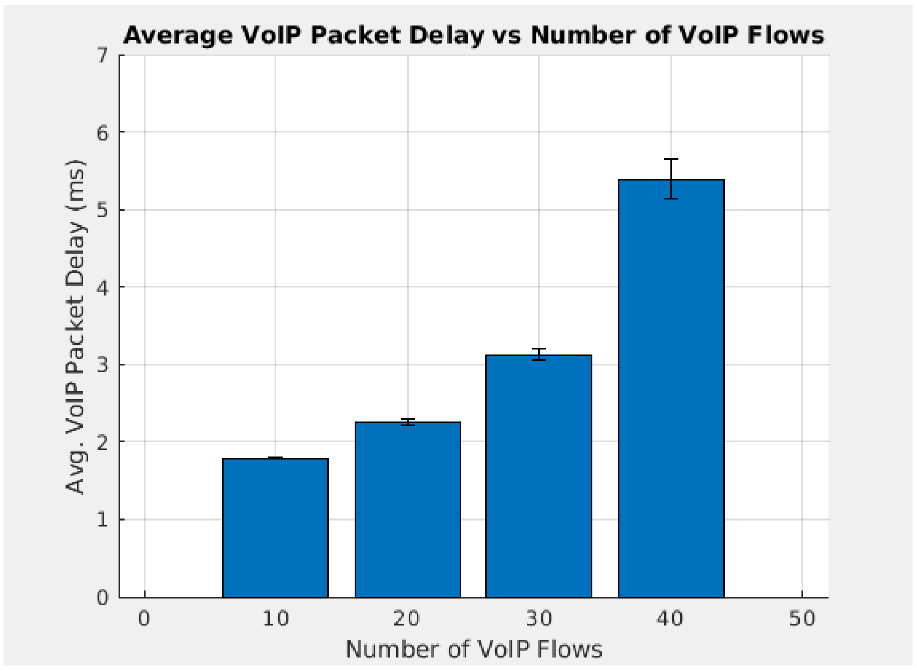

Figure 2.3: The average VoIP packet delay results, b = 1e-5

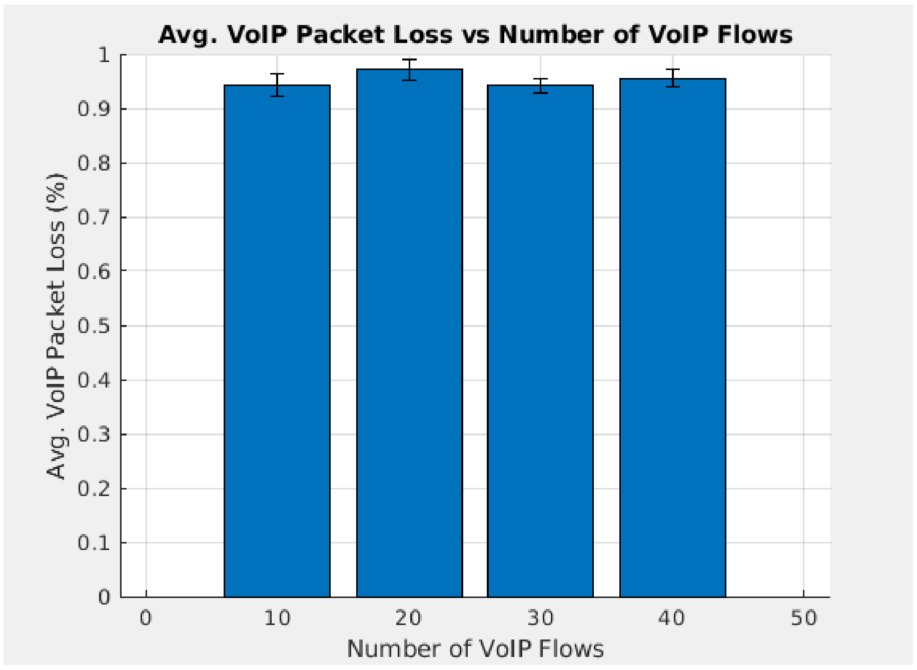

Figure 2.4: The average VoIP packet delay loss, b = 1e-5

Analysing Figure 2.1 and Figure 2.3 we can see that a particularity of these graphs is the fact that data packets have always a slightly higher delay than VoIP. Bearing in mind that VoIP packet sizes are evenly distributed between 110 and 130 bytes and data packets have values ranging from 64 to 1518 bytes with an average of 620 bytes. These results are expected, since larger packets tend to suffer a greater delay on average, so it is expected that the average packet delay for data packets to be greater than the average packet delay for VoIP packets. We also see that as the number of VoIP flows increases the average packet delay (for data and VoIP) also increases exponentially, this effect occurs because the number of packets arriving is increasingly approaching the system's link capacity.

The packet loss values can also be explained by the difference in the packets size, since the bit error rate is constant, it is expected to see a higher value of packet loss for data packets than VoIP packets. Since the data packet size is $\approx$ 5 times the size of VoIP packets (on average), we also see a packet loss $\approx$ 5 times higher for data packets when compared to packet loss of VoIP packets. We can also see that even though the packet loss values stays constant, the values fluctuate more than what was seen in the previous Task, this may be an indication that the packet rate is approaching the saturation threshold of the system's capacity and queue.

## 2.3 Exercise 2.b)

### 2.3.1 Code

```
rate = 1500;     % rate of arrival to the queue in pps
P = 100000;      % Packets to be transmitted (stoping criteria)
C = 10;          % Capacity of connectio with 10Mbps
f = 10000;       % Queue Size in Bytes
N = 20;          % Times to run the simulation
nVoips = [10 20 30 40];     % nr voip packets flows
b = 10^-5;       % Bit error rate
alfa = 0.1;      % 90% confidence interval for results

% Variables to store bar graph data
PLd_values = zeros(1, length(nVoips));
PLd_terms = zeros(1, length(nVoips));
PLv_values = zeros(1, length(nVoips));
PLv_terms = zeros(1, length(nVoips));
APDd_values = zeros(1, length(nVoips));
APDd_terms = zeros(1, length(nVoips));
APDv_values = zeros(1, length(nVoips));
APDv_terms = zeros(1, length(nVoips));

% Variables to store the simulation results
PLd = zeros(1, N);
```

```
PLv = zeros(1, N);
APDd = zeros(1, N);
APDv = zeros(1, N);
MPDd = zeros(1, N);
MPDv = zeros(1, N);
TT = zeros(1, N);

% Test all number of VoIP flows
for i = 1 : length(nVoips)
    % Run the simulator N times
    for it = 1:N
        [PLd(it), PLv(it), APDd(it), APDv(it), MPDd(it), ...
            MPDv(it), TT(it)] = Simulator3A(rate, C, f, P, ...
            nVoips(i), b);
    end

    % Calculate Avg. Packet Loss for Data Packets
    media = mean(PLd);
    term = norminv(1-alfa/2)*sqrt(var(PLd)/N);
    PLd_values(i) = media;
    PLd_terms(i) = term;

    % Calculate Avg. Packet Loss for VoIP Packets
    media = mean(PLv);
    term = norminv(1-alfa/2)*sqrt(var(PLv)/N);
    PLv_values(i) = media;
    PLv_terms(i) = term;

    % Calculate Avg. Packet Delay for Data Packets
    media = mean(APDd);
    term = norminv(1-alfa/2)*sqrt(var(APDd)/N);
    APDd_values(i) = media;
    APDd_terms(i) = term;

    % Calculate Avg. Packet Delay for VoIP Packets
    media = mean(APDv);
    term = norminv(1-alfa/2)*sqrt(var(APDv)/N);
    APDv_values(i) = media;
    APDv_terms(i) = term;
end

% Figure to show the asked results for 2.b) i)
figure(1);
hold on;
grid on;
bar(nVoips, APDd_values');% Bar Graph
ylim([0 4]);              % Set y axis values between 0 and 4
er = errorbar(nVoips, APDd_values', APDd_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Data Packet Delay (ms)')
title('Average Data Packet Delay vs Number of VoIP Flows');
hold off;
```

```

% Figure to show the asked results for 2.b) ii)
figure(2);
hold on;
grid on;
bar(nVoips, PLd_values'); % Bar Graph
er = errorbar(nVoips, PLd_values', PLd_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Data Packet Loss (%)')
title('Avg. Data Packet Loss vs Number of VoIP Flows');
hold off;

% Figure to show the asked results for 2.b) iii)
figure(3);
hold on;
grid on;
bar(nVoips, APDv_values'); % Bar Graph
ylim([0 4]);               % Set y axis values between 0 ...
    and 4
er = errorbar(nVoips, APDv_values', APDv_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. VoIP Packet Delay (ms)')
title('Average VoIP Packet Delay vs Number of VoIP Flows');
hold off;

% Figure to show the asked results for 2.b) iv)
figure(4);
hold on;
grid on;
bar(nVoips, PLv_values');   % Bar Graph
er = errorbar(nVoips, PLv_values', PLv_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. VoIP Packet Loss (%)')
title('Avg. VoIP Packet Loss vs Number of VoIP Flows');
hold off;
```

### 2.3.2 Results and Conclusions

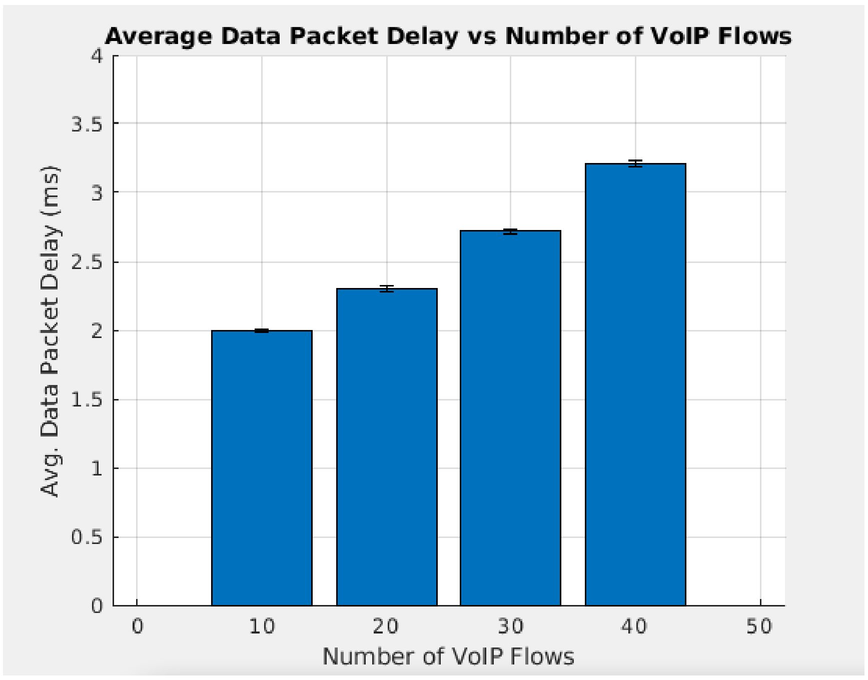

Figure 2.5: The average data packet delay results, b = 1e-5, f = 10000 Bytes

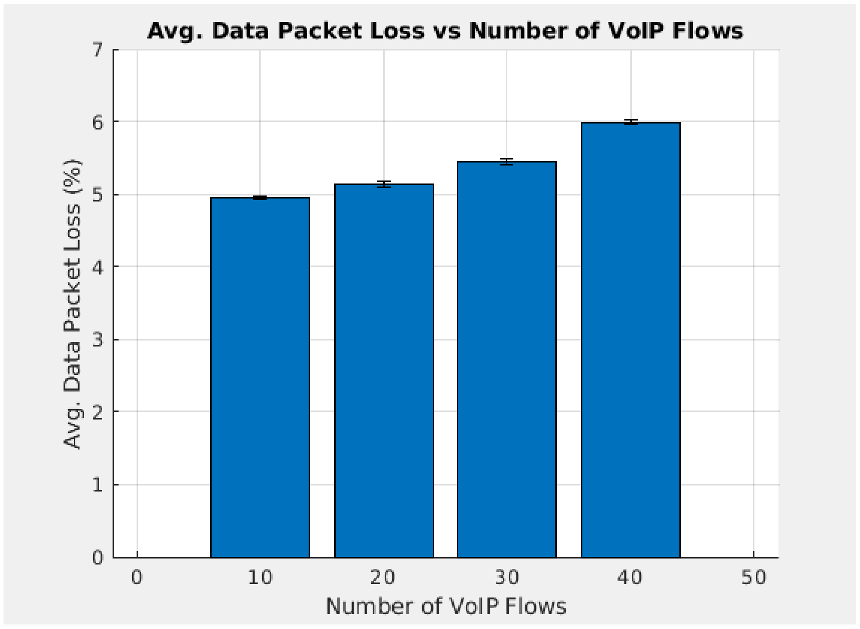

Figure 2.6: The average data packet loss results, b = 1e-5, f = 10000 Bytes

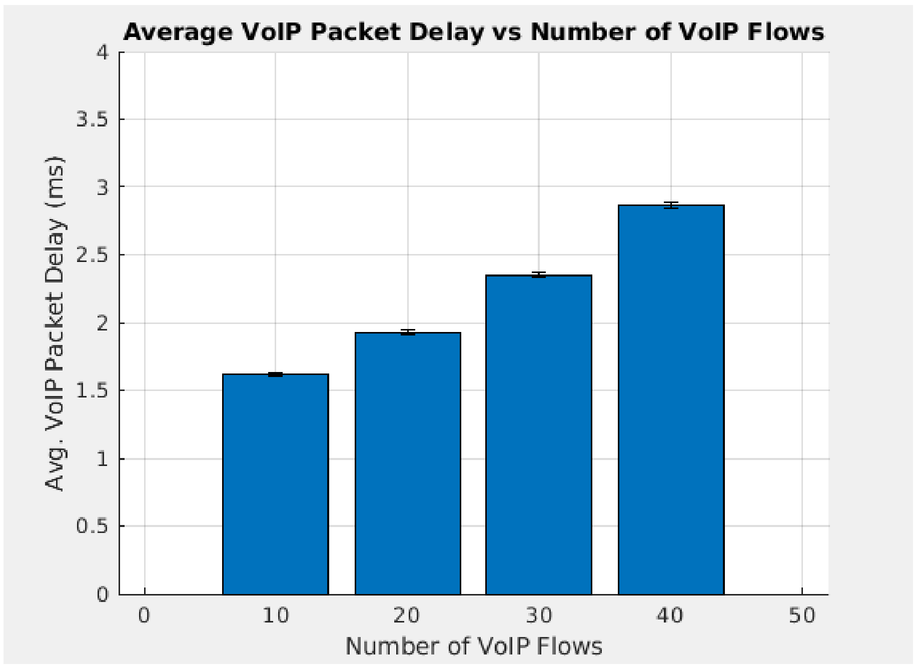

Figure 2.7: The average VoIP packet delay results, b = 1e-5, f = 10000 Bytes

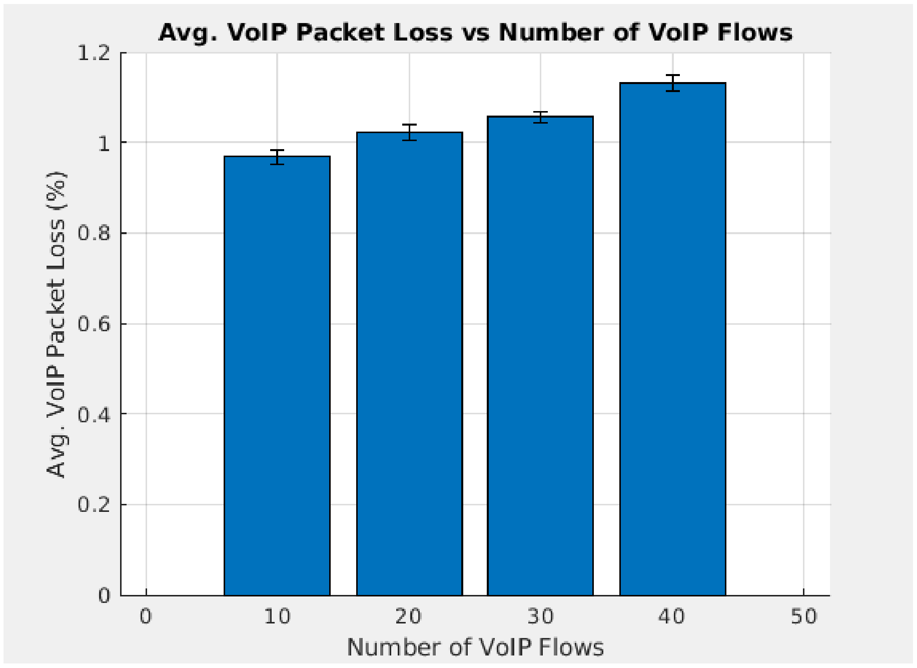

Figure 2.8: The average VoIP packet loss results, b = 1e-5, f = 10000 Bytes

Analysing Figure 2.5 and Figure 2.7 we see the same behaviour as in the previous exercise. As the number of VoIP flows increases so does the average packet delay, we also see that the average packet delay for data packets is slightly higher than the delay for VoIP packets, this behaviour is to be expected as the reduction of the queue size doesn't impact the system's capacity nor the packets size. What we see however, is a decrease in average packet delay (for data and VoIP) when compared with the results got in the previous experiment, these results can be explained by the reduction of queue size by 100 times. By being smaller, the packets will wait less time (on average) to be processed.

Looking at Figure 2.6 and Figure 2.8 we can see a couple of new things. Firstly, we see the same behaviour as in the previous exercise where the average packet loss for data packets is higher than VoIP packets, again an expected behaviour because we didn't change the packets size and the bit error rate is constant.

Secondly, we can see that the packet loss increases as the number of VoIP flows increases. This behaviour can be explained by the reduction of the queue size.

Lastly, by comparing our results with the previous exercise, we can see that the packet loss values both for data and VoIP packets are slightly higher for a number of 10 VoIP flows, this is an indication that in previous exercise the packet arrival rate was saturating the system's link capacity and was occupying a fair share of the queue, and when decreased started to drop some packets.

## 2.4 Exercise 2.c)

### 2.4.1 Code

```
% Eq prob for Data with size different of 64, 10 and 1518
prob_left_data = (1 - (0.19 + 0.23 + 0.17)) / ((109 - 65 + ...
    1) + (1517 - 111 + 1));
% Equal probability for VoIP Packets
prob_voip = 1 / (130 - 110 + 1);

b = 1e-5;                  % Values of ber to test
ploss_data = 0;            % Probabilities of data packet loss
ploss_voip = 0;            % Probabilities of VoIP packet loss

% Calculate the prob. of having at least 1 error for the ...
    given packet size
for packetSize = 64 : 1518
    if (packetSize == 64)
        ploss_data = ploss_data + (1 ...
            -(nchoosek(packetSize*8, 0) * b^0 * ...
                (1-b)^(packetSize*8))) * 0.19;
    elseif (packetSize == 110)
        ploss_data = ploss_data + (1 ...
            -(nchoosek(packetSize*8, 0) * b^0 * ...
            (1-b)^(packetSize*8))) * 0.23;
    elseif (packetSize == 1518)
        ploss_data = ploss_data + (1 ...
            -(nchoosek(packetSize*8, 0) * b^0 * ...
            (1-b)^(packetSize*8))) * 0.17;
    else
        ploss_data = ploss_data + (1 ...
            -(nchoosek(packetSize*8, 0) * b^0 * ...
            (1-b)^(packetSize*8))) * prob_left_data;
    end
end

ploss_data = (ploss_data ./ (0.19 + 0.23 + 0.17 + ((109 - ...
    65 + 1) + (1517 - 111 + 1)) * prob_left_data)) * 100;

% Calculate the prob. of having at least 1 error
for packetSize = 110 : 130
    ploss_voip = ploss_voip + (1 -(nchoosek(packetSize*8, ...
        0) * b^0 * (1-b)^(packetSize*8))) * prob_voip;
end

ploss_voip = ploss_voip .* 100;

fprintf('Considering that the bit error rate is the only ...
    factor that can cause Packet to be lost, the ...
    theoretical packet loss (%%)\nfor data packet is: %.4f ...
    %%\nand for a voip packets is: %.4f %%\n', ploss_data, ...
    ploss_voip);
```

### 2.4.2 Results and Conclusions

The program's output was: "Considering that the bit error rate is the only factor that can cause packet to be lost, the theoretical packet loss (%) for data packet is: 4.7365 % and for a VoIP packets is: 0.9554 %".

The results got, let us believe that the packet loss experienced in exercise 2.a) is only due to the bit error rate, as the theoretical values are similar to the results got in our experiments. These results further confirm our hypothesis that packet loss is only due to bit error rate as the system's capacity link is not fully saturated.

On exercise 2.b) we see that initially for 10 flows, the results got are close to the theoretical value but as the VoIP flows increase the value of packet loss got further away from the theoretical value, again a good confirmation that, in this case, the number of packets arriving is higher than the system's capacity and queue can handle. Therefore packets start to drop not only because of bit error rate but because the queue is full.

# Chapter 3

# Task 3

## 3.1 Exercise 3.a)

### 3.1.1 Code

Code of the experiment in exercise 3.a), using Simulator3 (developed in the lab classes), that estimates the performance parameters when both services (VoIP and Data) are statistically multiplexed in a single FIFO queue.

```
rate = 1500;                 % rate of arrival (pps)
P = 100000;                  % stoping criteria (nr. of packets)
C = 10;                      % capacity of the connection: ...
    10Mbps
f = 10^4;                    % queue size (Bytes)
N = 20;                      % times to run the simulation
voip_flows = [10 20 30 40]; % nr voip flows
b = 10^-5;                   % bit error rate
alfa = 1 - 0.9;              % 90% confidence interval

% Variables to store bar graph data
APDdata_values = zeros(1, length(voip_flows));
APDdata_terms = zeros(1, length(voip_flows));
PLdata_values = zeros(1, length(voip_flows));
PLdata_terms = zeros(1, length(voip_flows));

APDvoip_values = zeros(1, length(voip_flows));
APDvoip_terms = zeros(1, length(voip_flows));
PLvoip_values = zeros(1, length(voip_flows));
PLvoip_terms = zeros(1, length(voip_flows));

% Variables to store simulation results
PLdata = zeros(1, N);
PLvoip = zeros(1, N);
APDdata = zeros(1, N);
APDvoip = zeros(1, N);
MPDdata = zeros(1, N);
```

```
MPDvoip = zeros(1, N);
TT = zeros(1, N);
% Test all numbers of VoIP flows
for i = 1:length(voip_flows)
    % Run the simulator N times
    for it = 1:N
            [PLdata(it), PLvoip(it), APDdata(it), ...
                APDvoip(it), MPDdata(it), MPDvoip(it), ...
                TT(it)] = Simulator3(rate, C, f, P, ...
                voip_flows(i));
    end

    % Calculate Avg. Packet Delay for DATA packets
    media = mean(APDdata);
    term = norminv(1-alfa/2)*sqrt(var(APDdata)/N);
    APDdata_values(i) = media;
    APDdata_terms(i) = term;

    % Calculate Avg. Packet Delay for VoIP packets
    media = mean(APDvoip);
    term = norminv(1-alfa/2)*sqrt(var(APDvoip)/N);
    APDvoip_values(i) = media;
    APDvoip_terms(i) = term;

    % Calculate Packet Loss for DATA packets
    media = mean(PLdata);
    term = norminv(1-alfa/2)*sqrt(var(PLdata)/N);
    PLdata_values(i) = media;
    PLdata_terms(i) = term;

    % Calculate Packet Loss for VoIP packets
    media = mean(PLvoip);
    term = norminv(1-alfa/2)*sqrt(var(PLvoip)/N);
    PLvoip_values(i) = media;
    PLvoip_terms(i) = term;
end

% Plot Avg. Packet Delay for DATA packets
figure(1);
hold on;
grid on;
bar(voip_flows, APDdata_values');
ylim([0 4]);
er = errorbar(voip_flows, APDdata_values', APDdata_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Packet Delay of DATA packets(ms)')
title('Avg. DATA Packet Delay vs VoIP Flows');
hold off;

% Plot Avg. Packet Delay for VoIP packets
figure(2);
hold on;
```

```
grid on;
bar(voip_flows, PLdata_values');
er = errorbar(voip_flows, PLdata_values', PLdata_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Packet Loss of DATA packets(%)')
title('DATA Packet Loss vs VoIP Flows');
hold off;

% Plot Packet Loss for DATA packets
figure(3);
hold on;
grid on;
bar(voip_flows, APDvoip_values');
ylim([0 4]);
er = errorbar(voip_flows, APDvoip_values', APDvoip_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Avg. Packet Delay of VoIP packets(ms)')
title('Avg. VoIP Packet Delay vs VoIP Flows');
hold off;

% Plot Packet Loss for VoIP packets
figure(4);
hold on;
grid on;
bar(voip_flows, PLvoip_values');
er = errorbar(voip_flows, PLvoip_values', PLvoip_terms);
er.Color = [0 0 0];
er.LineStyle = 'none';
xlabel('Number of VoIP Flows')
ylabel('Packet Loss of VoIP packets(%)')
title('VoIP Packet Loss vs VoIP Flows');
hold off;
```

### 3.1.2 Results and Conclusions

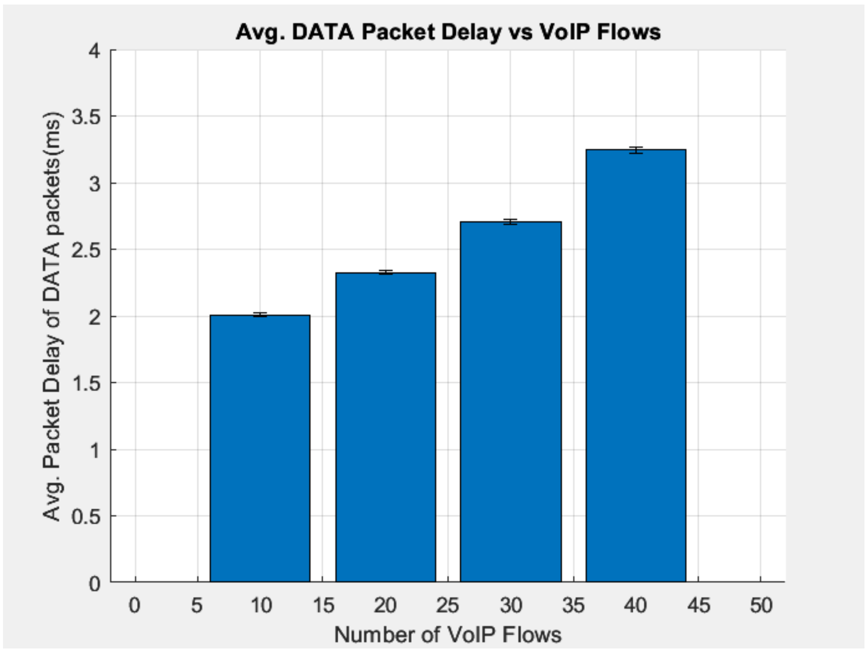

Figure 3.1: The average data packet delay results

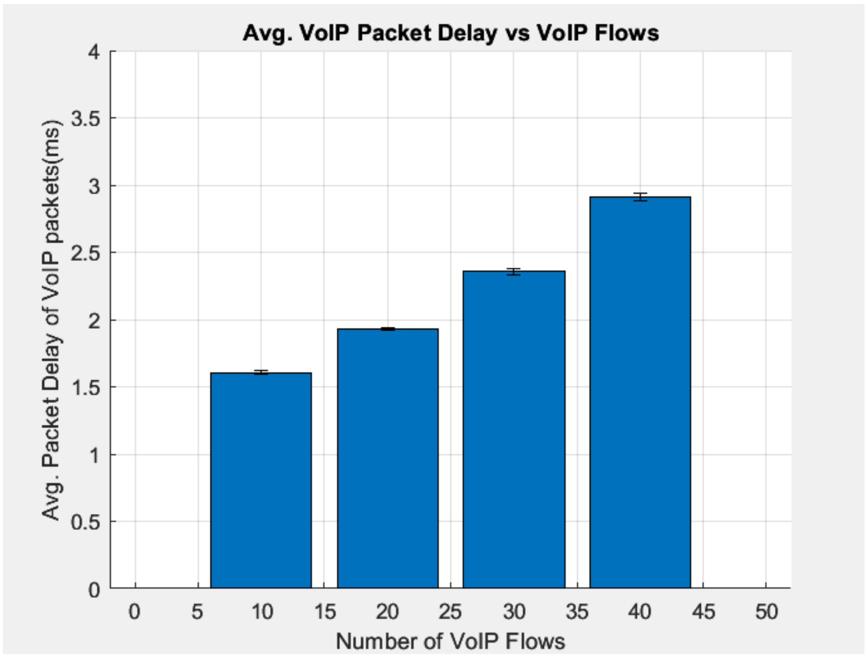

Figure 3.2: The average VoIP packet delay results

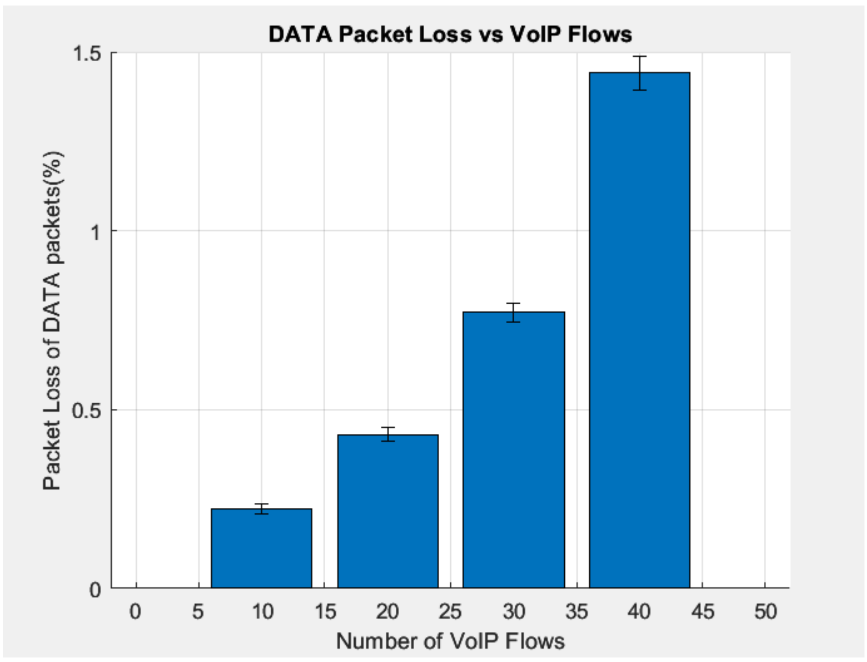

Figure 3.3: Data packet loss results

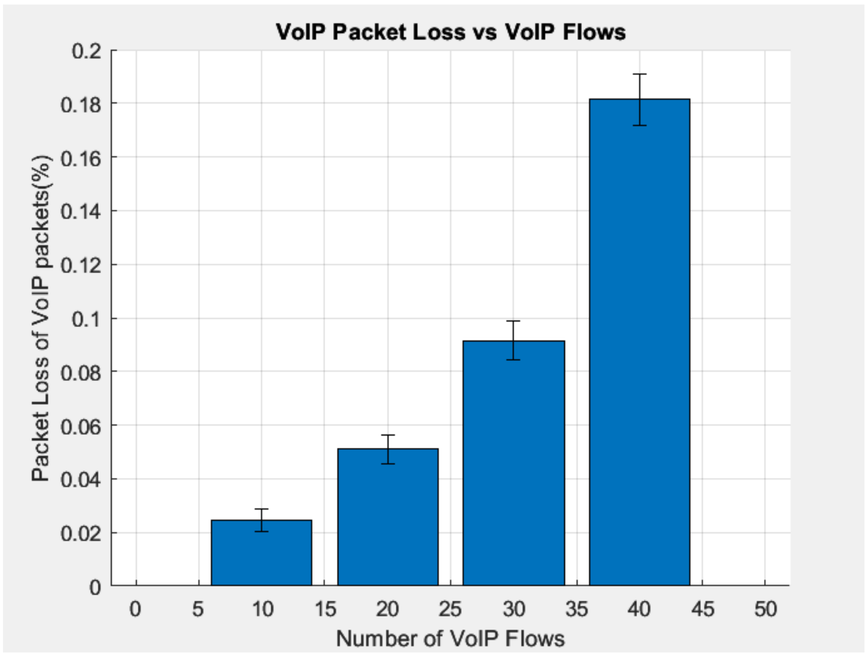

Figure 3.4: VoIP packet loss results

When analysing Figure 3.1 and Figure 3.2, we can directly conclude, as predictable, that as the VoIP flows increase, the average packet delay does too, for both services. Regarding this scenario, there's a slightly bigger delay in the data packets in comparison with the VoIP ones, as the size of the data packets is approximately 620 bytes, while the VoIP packets have an average size of 120 bytes. Therefore, we can conclude what was expected: larger packets have a higher delay.
When it comes to the average packet loss, represented in the Figure 3.3 and Figure 3.4, the data packets have a higher packet loss than the VoIP packets, since the former are $\approx 5$ times bigger than the latter. Finally, it can be directly concluded that the packet loss will increase as the VoIP flows also increase, as expected because there will be 2, 3 and 4 times more VoIP flows, respectively.

## 3.2 Exercise 3.b)

### 3.2.1 Code

The code of exercise 3.b) has the intent of doing the same experiments that were made in exercise 3.a), having, as the only difference in the experiment code, the following lines:

```matlab
for it = 1:N
    [PLdata(it), PLvoip(it), APDdata(it), APDvoip(it), ...
        MPDdata(it), MPDvoip(it), TT(it)] = ...
        Simulator4(rate, C, f, P, voip_flows(i));
end
```

The Simulator4 was also developed in the lab classes and the relevant difference between it and the Simulator3 is that the former prioritizes the VoIP packets.

### 3.2.2 Results and Conclusions

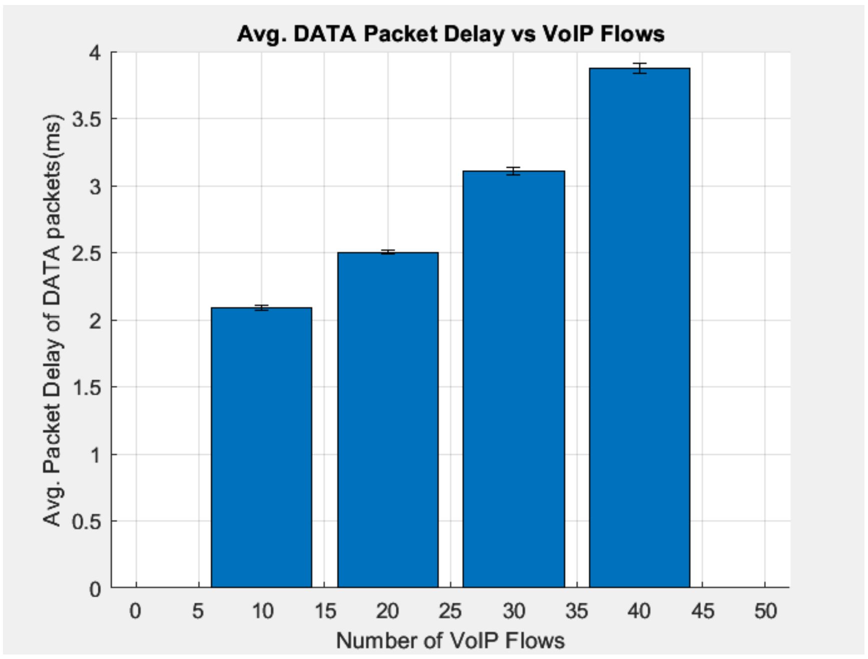

Figure 3.5: The average data packet delay results

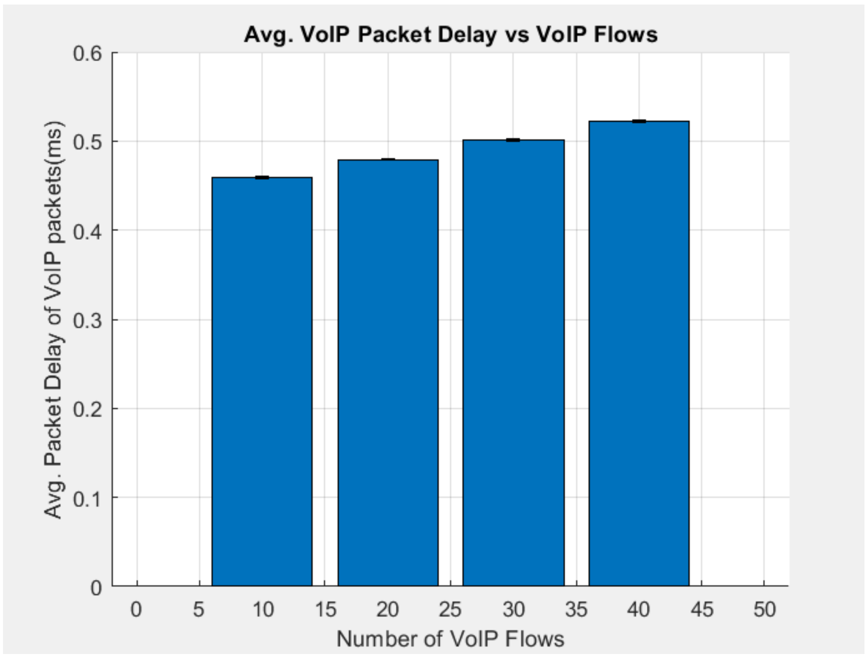

Figure 3.6: The average VoIP packet delay results

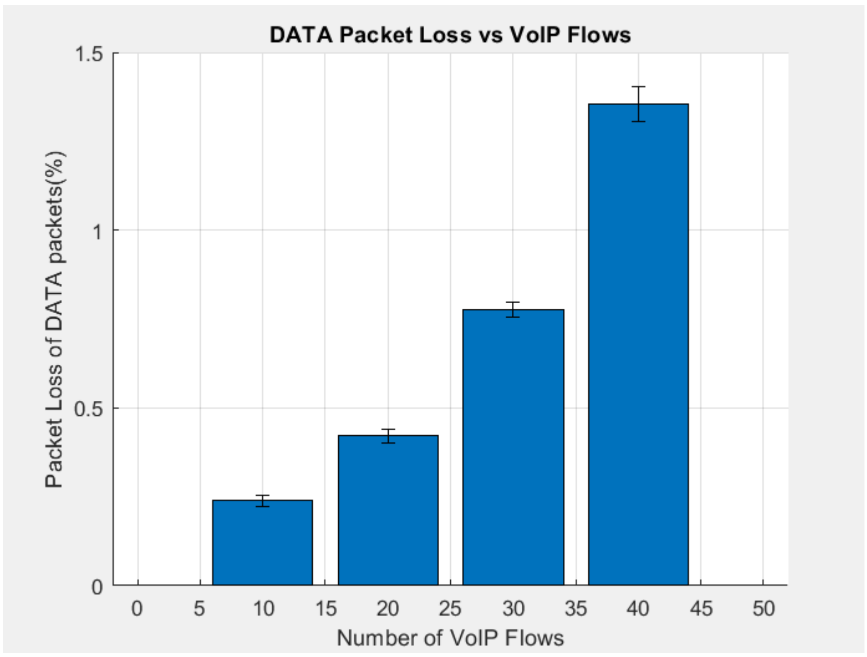

Figure 3.7: Data packet loss results

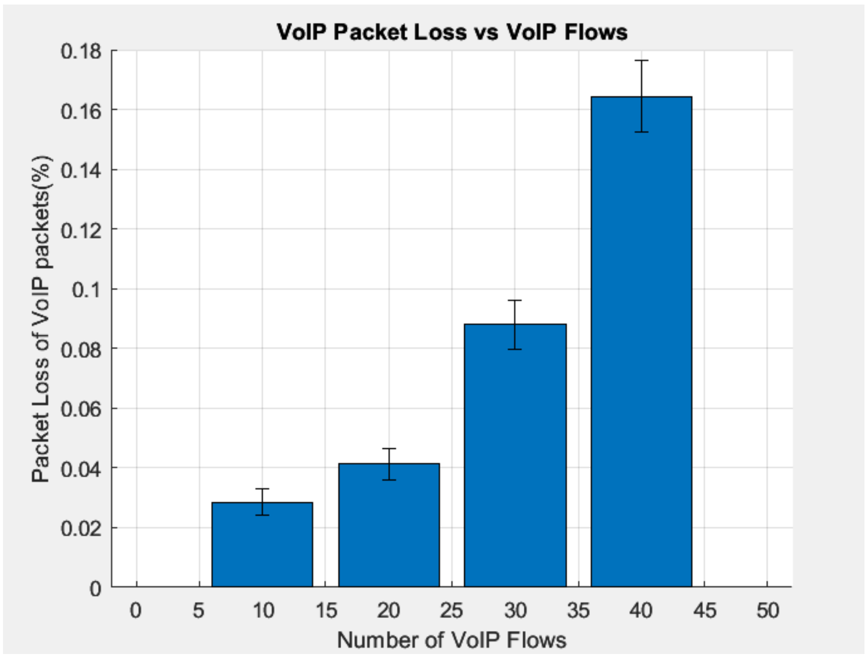

Figure 3.8: VoIP packet loss results

As seen in Figure 3.5 and Figure 3.6, the average packet delay increases as the VoIP flows increase in both services, however, the increase in the VoIP service is very slight, since the packets of this service have higher priority, therefore, they won't be much time in the queue and since they have a small packet size, compared to the data packets, even though there are more VoIP flows, it doesn't make much change. Comparing with the results got in the exercise 3.a), the average packet delay for the data service gets higher, as the VoIP packets get prioritized, which is also reflected in the decrease of average packet delay in the VoIP service.

In terms of packet loss between this 2 services, there's a big difference, again, because of the average packet size being much higher in the data service than the VoIP service. However, when compared to the exercise 3.a) values, there's almost no change, since we have the same capacity, queue size and packet rate. There is, although, a very slight decrease of packet loss in the VoIP service, when there are more VoIP flows since they have more priority in Simulator4 - which will put them in front of the other packets in the queue, as if they were in the queue for longer than the data packets -, only noticeable when we have a higher number of VoIP flows. In conclusion, it's also relevant to note that the packet loss for the VoIP service doesn't get much affected, since the VoIP packets have a very small size and even without priority the queue usually has space for them.

## 3.3 Exercise 3.c)

### 3.3.1 Code

The code of exercise 3.c) has the intent of doing the same experiments that were made in exercise 3.b), using a different version of Simulator4, therefore, the code showed below is the Simulator4A's most relevant differences.

```
function [PLd, PLv, APDd, APDv, MPDd, MPDv, TT] = ...
    Simulator4A(lambda,C,f,P,n)
```

...

```
%Simulation loop:
while (TRANSMITTEDPACKETSD + TRANSMITTEDPACKETSV) < P   % ...
    Stopping criterium
    EventList= sortrows(EventList,2);    % Order EventList ...
        by time
    Event= EventList(1,1);               % Get first event and
    Clock= EventList(1,2);               %   and
    PacketSize= EventList(1,3);          %   associated
    ArrivalInstant= EventList(1,4);      %   parameters.
    PacketType= EventList(1,5);             %   get the packet ...
        type
    EventList(1,:)= [];                     % Eliminate first ...
        event
    switch Event
        case ARRIVAL                        % If first event ...
            is an ARRIVAL
            if (PacketType == DATA)         % Data Packet
                TOTALPACKETSD= TOTALPACKETSD+1;
                tmp= Clock + exprnd(1/lambda);
                EventList = [EventList; ARRIVAL, tmp, ...
                    GeneratePacketSize(), tmp, DATA];
                if STATE==0
                    STATE= 1;
                    EventList = [EventList; DEPARTURE, ...
                        Clock + 8*PacketSize/(C*10^6), ...
                        PacketSize, Clock, DATA];
                else
```

Most relevant difference:

```

                    if (QUEUEOCCUPATION + PacketSize)/f ≤ ...
                        0.9   % only accepts data packets ...
                        if the queue doesn't become higher ...
                        than 90%
                        QUEUE= [QUEUE;PacketSize , Clock, ...
                            DATA];
                        QUEUEOCCUPATION= QUEUEOCCUPATION + ...
                            PacketSize;
                    else
                        LOSTPACKETSD= LOSTPACKETSD + 1;
                    end
                end
```

```
            else                                        % VoIP ...
                Packet
                TOTALPACKETSV = TOTALPACKETSV+1;
                tmp = Clock + unifrnd(0.016, 0.024);
                EventList = [EventList; ARRIVAL, tmp, ...
                    randi([110, 130]), tmp, VOIP];
                if STATE == 0
                    STATE = 1;
                    EventList = [EventList; DEPARTURE, ...
                        Clock + 8*PacketSize/(C*10^6), ...
                        PacketSize, Clock, VOIP];
                else
                    if QUEUEOCCUPATION + PacketSize ≤ f
                        QUEUE = [QUEUE;PacketSize , Clock, ...
                            VOIP];
                        QUEUEOCCUPATION = QUEUEOCCUPATION + ...
```

```
                                        PacketSize;
                              else
                                   LOSTPACKETSV= LOSTPACKETSV + 1;
                              end
                         end
                    end
```

Even though this simulator has the same principles of the Simulator4, developed in the lab classes, the previous referred difference is that there's a new packet discard strategy considered, which is: VoIP packets are always accepted in the queue (if there is enough space) but data packets are accepted in the queue only if the total queue occupation does not become higher than 90% (a simplified version of WRED – Weighted Random Early Discard).

### 3.3.2 Results and Conclusions

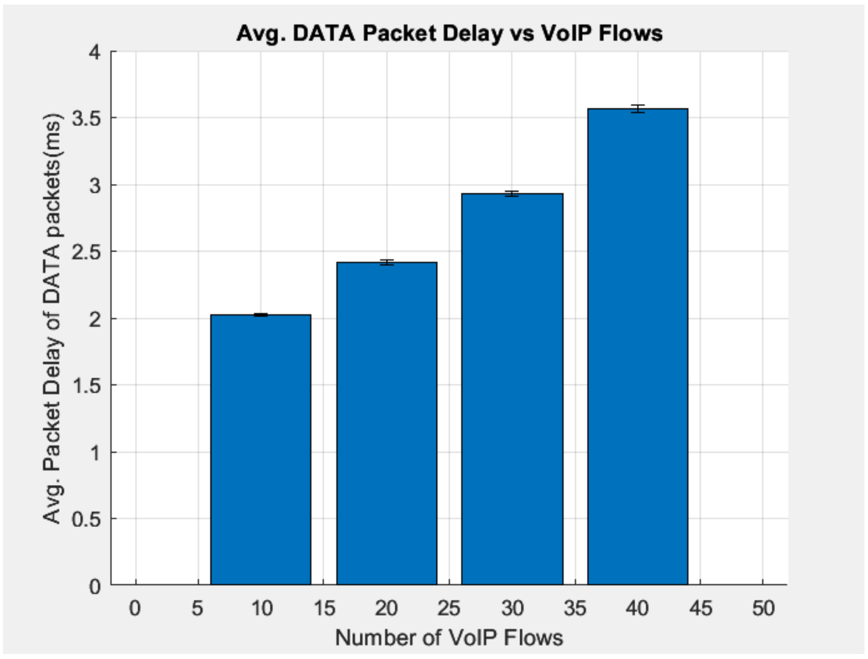

Figure 3.9: The average data packet delay results

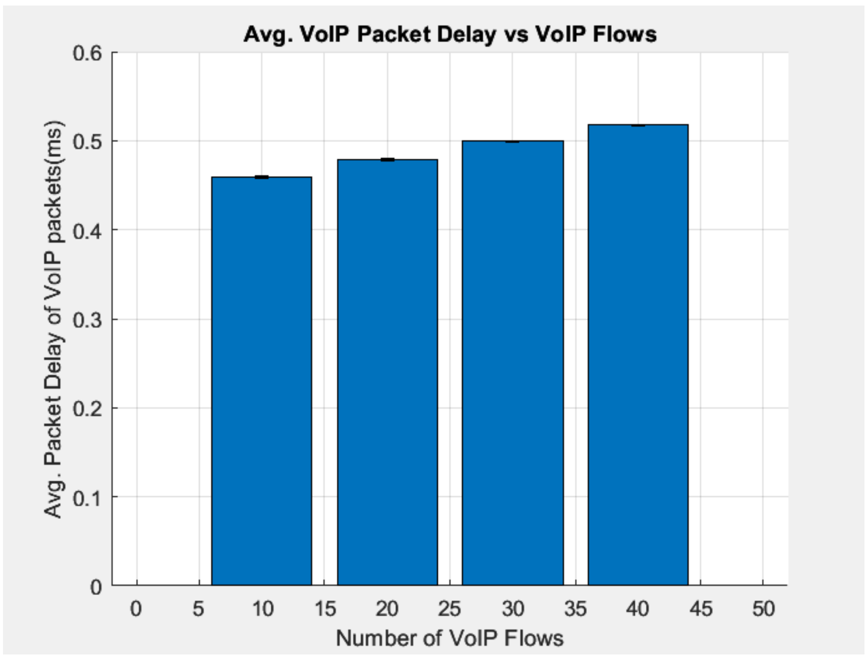

Figure 3.10: The average VoIP packet delay results

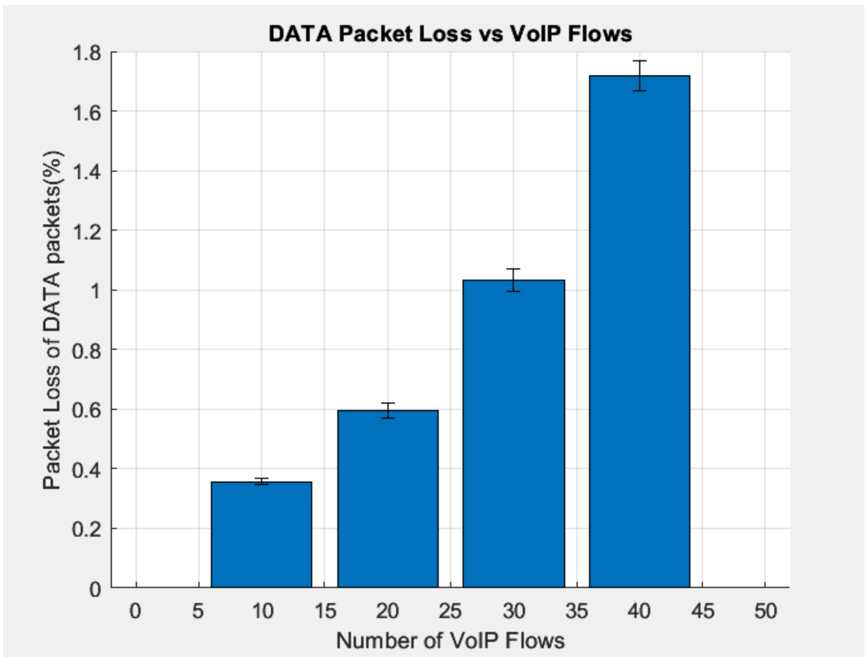

Figure 3.11: Data packet loss results

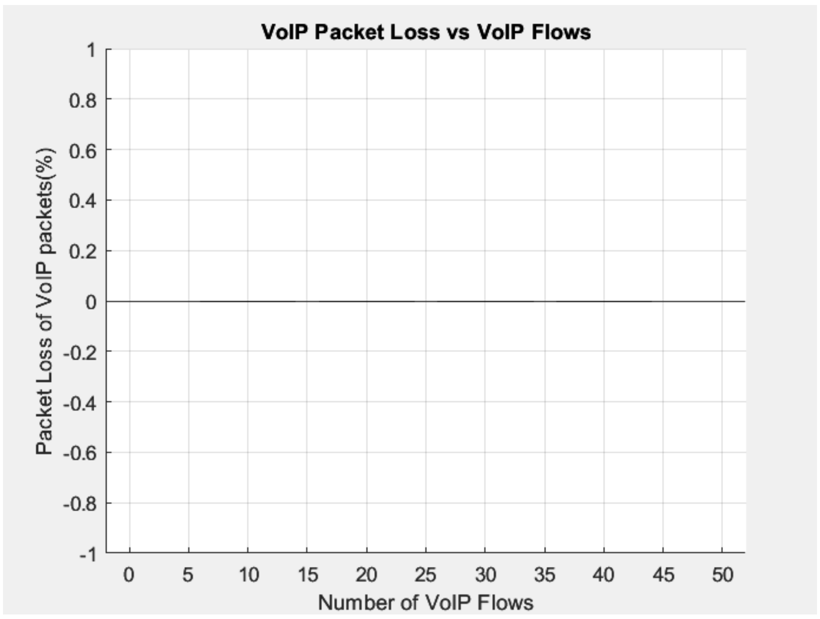

Figure 3.12: VoIP packet loss results

Regarding the Figure 3.9 and Figure 3.10, it's possible to retain that the average packet delay, as expected, increases as the number of VoIP flows raise, for both services. However, when it comes to the VoIP service there's a really small increase, since they have priority over the data packets and have a small packet size.
Besides that, when comparing with exercise 3.b) there's only a residual difference, because the VoIP service didn't get much affected by the Simulator4A change. Even tough the data service got some changes, they didn't affect their packet delay, since the departure procedure wasn't affected.

On the other hand, the packet loss, represented in Figure 3.11 and Figure 3.12, the data service increases as the number of VoIP flows also raises. However, what really raises attention is the fact that the VoIP packet loss is 0%. This happens due to the fact that not only the VoIP packets have a higher priority, they also always have, at least, 10% of the queue reserved for them to arrive, since the data packets can only enter in the queue if this entry doesn't reflect in a 90% or higher occupation. This 10% is 1000 bytes (because f = 10KBytes), which can fit at least 7 VoIP packets of their maximum size (130 Bytes), therefore it's not likely that VoIP packets are going to be discarded.

Comparing the results of this exercise with the ones of exercise 3.b) about the packet loss, we can conclude that's were the major difference stays. Since the data packets in this exercise get discarded more easily, because of the new rule in the Simulator4A.

## 3.4 Exercise 3.d)

### 3.4.1 Code

The code of exercise 3.d) has the intent of doing the same experiments that were made in exercise 3.b) and 3.c), using a different version of Simulator4 and Simulator4A, therefore, the code showed below is the Simulator4B's.

```
function [PLd, PLv, APDd, APDv, MPDd, MPDv, TT] = ...
    Simulator4B(lambda,C,f,P,n)
```

...

```
%Similation loop:
while (TRANSMITTEDPACKETSD + TRANSMITTEDPACKETSV) < P ...
                  % Stopping criterium
    EventList= sortrows(EventList,2);    % Order EventList ...
        by time
    Event= EventList(1,1);               % Get first event and
    Clock= EventList(1,2);               %   and
    PacketSize= EventList(1,3);          %   associated
    ArrivalInstant= EventList(1,4);      %   parameters.
    PacketType= EventList(1,5);          %   get the packet ...
        type
    EventList(1,:)= [];                  % Eliminate first ...
        event
    switch Event
        case ARRIVAL                     % If first event ...
            is an ARRIVAL
            if (PacketType == DATA)      % Data Packet
                TOTALPACKETSD= TOTALPACKETSD+1;
                tmp= Clock + exprnd(1/lambda);
                EventList = [EventList; ARRIVAL, tmp, ...
                    GeneratePacketSize(), tmp, DATA];
                if STATE==0
                    STATE= 1;
                    EventList = [EventList; DEPARTURE, ...
                        Clock + 8*PacketSize/(C*10^6), ...
                        PacketSize, Clock, DATA];
                else
                    if QUEUEOCCUPATION + PacketSize ≤ f
                        QUEUE= [QUEUE;PacketSize , Clock, ...
                            DATA];
                        QUEUEOCCUPATION= QUEUEOCCUPATION + ...
                            PacketSize;
                    else
                        LOSTPACKETSD= LOSTPACKETSD + 1;
                    end
                end
            else                                    % VoIP ...
                Packet
                TOTALPACKETSV = TOTALPACKETSV+1;
                tmp = Clock + unifrnd(0.016, 0.024);
                EventList = [EventList; ARRIVAL, tmp, ...
                    randi([110, 130]), tmp, VOIP];
                if STATE == 0
                    STATE = 1;
                    EventList = [EventList; DEPARTURE, ...
                        Clock + 8*PacketSize/(C*10^6), ...
                        PacketSize, Clock, VOIP];
                else
```

Most relevant difference:

```
                    if (QUEUEOCCUPATION + PacketSize)/f ≤ ...
                        0.9   % only accepts VoIP packets ...
                        if the queue doesn't become higher ...
                        than 90%
                        QUEUE = [QUEUE;PacketSize , Clock, ...
                            VOIP];
                        QUEUEOCCUPATION = QUEUEOCCUPATION + ...
                            PacketSize;
                    else
```

```
                        LOSTPACKETSV= LOSTPACKETSV + 1;
                    end
                end
            end
```

### 3.4.2 Results and Conclusions

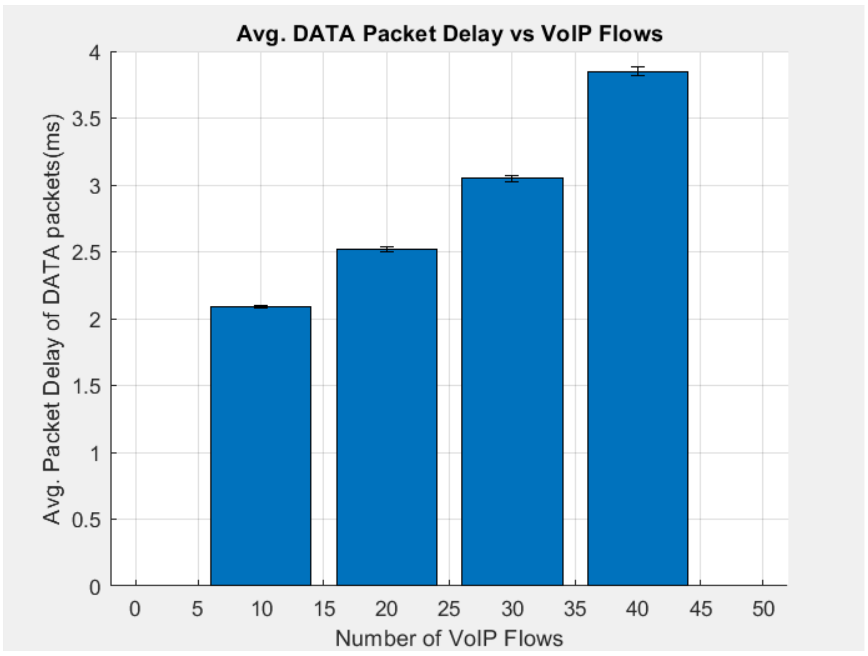

Figure 3.13: The average data packet delay results

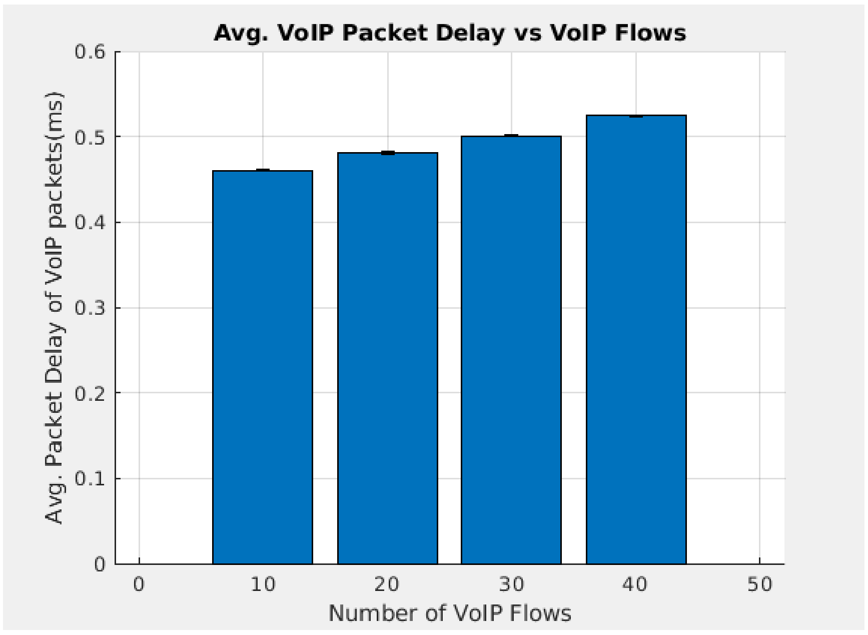

Figure 3.14: The average VoIP packet delay results

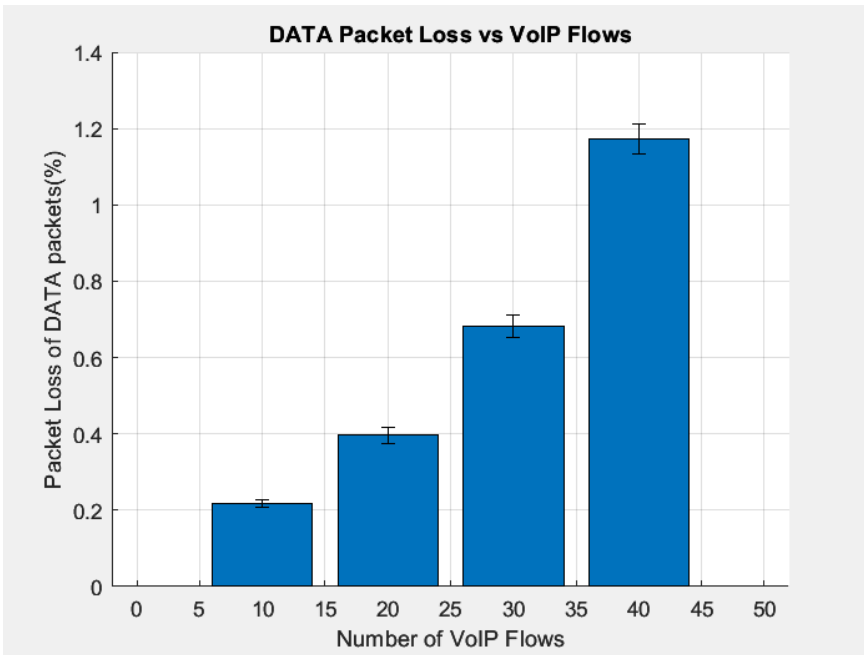

Figure 3.15: Data packet loss results

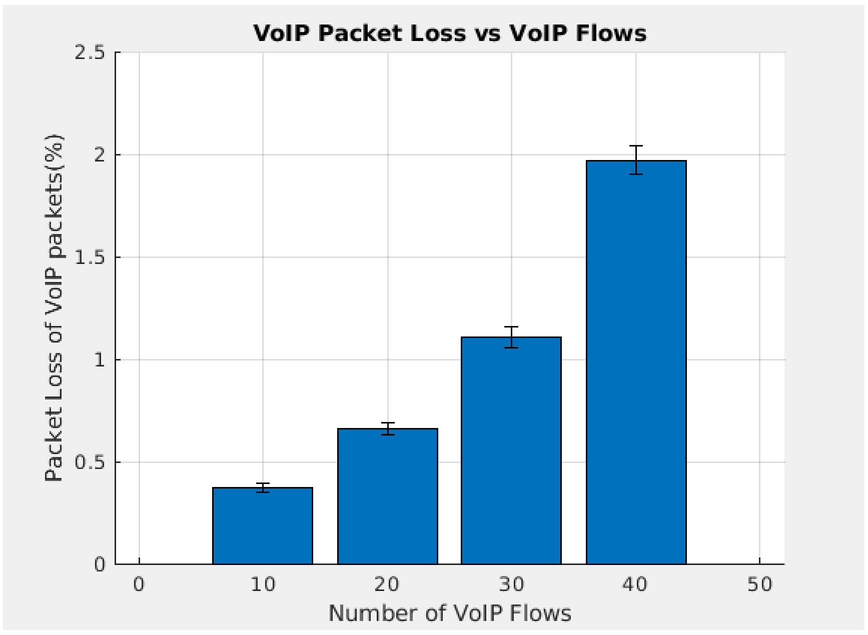

Figure 3.16: VoIP packet loss results

Finally, in exercise 3.d), when analysing Figure 3.13 and Figure 3.14, we can conclude that the average packet delay increases as the number of VoIP flows raises, since there's more packets entering the queue. With that in mind, the data packets still have a higher delay due to the fact that they have a bigger size and less priority than the VoIP service. The elevation in the average VoIP packets delay is residual, since they have a higher priority than the data packets.
Even though the discarding rule changed - stating that VoIP packets are accepted in the queue only if the total queue occupation does not become higher than 90% -, the VoIP average packet delay didn't get a significant difference and the data packets average delay got values more similar to exercise 3.b) than 3.c), which is expected, because the data packets no longer have an early discard rule. This results didn't get significant differences due to the same reasons listed before: the departure procedure wasn't very affected by the Simulator changes.

When it comes to packet loss, depicted in Figure 3.15 and Figure 3.16, we can observe that between services, even though they both increase as the number of VoIP flows increase, since there are more packets trying to enter the queue, the packet loss is higher in the VoIP than the data service for the first time, since the VoIP packets have another discard rule, therefore, there's more chance of them being lost. Another factor to this variant, is the fact that, since data packets are bigger, in average, there's more probability of the queue being more full. The same goes when comparing the results of VoIP packet loss for this exercise and exercise 3.b); the packet loss of the VoIP service is quite bigger.
Unlike exercise 3.c), the packet loss for the data service isn't empty, since the data packets occupy more of the queue and are more easily considered lost, because the queue doesn't have enough space for them. However, it's a smaller packet loss compared to exercise 3.b), which is more notable for smaller numbers of VoIP flows since they won't flood the queue until 90% as much -, as we must not forget that this packets still have priority in the queue over the data ones-, because there's always 10% of the queue available for data packets, which may fit some of them but not all, since the range of data packets size is 64 to 1518 bytes.

# Chapter 4

# Information

The contributions between each member of the group were equal.
The project's repository can be viewed here: `https://github.com/PedroRocha9/MDRS`
A collaborator invitation in GitHub was sent to the professor's organization email: asou@ua.pt.