universidade de aveiro

Departamento de Eletrónica, Telecomunicações e Informática

Desempenho e Dimensionamento de Redes

Trabalho Prático 3

2020/2021

P1

Alunos:

Rodrigo Santos, 89180

Alexey Kononov, 89227

Introdução:

Neste relatório será usado em diversas alíneas um simulador de um serviço de reprodução de vídeo com o nome “simulator2” e outro de nome “simulator3”, para informação detalhada sobre as suas construções, código e funcionamento ver o apêndice A e B, no final do relatório.

Task 3:

a)

```
P = 10000;
alfa = 0.1;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 0;
N = 10;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataPL = zeros(1,5);
dataMPD = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
errhighPL = zeros(1,5);
errhighMPD = zeros(1,5);
errhighAPD = zeros(1,5);
errhighTT = zeros(1,5);
errlowPL = zeros(1,5);
errlowMPD = zeros(1,5);
errlowAPD = zeros(1,5);
errlowTT = zeros(1,5);
i = 1;

for lam = lambda
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lam, C, f, P, b);
    end

    alfa= 0.1; %90% confidence interval%
    dataPL(i) = mean(resultsPL);
    termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/N);
    errhighPL(i) = termPL;
    errlowPL(i) = - termPL;
    dataAPD(i) = mean(resultsAPD);
    termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/N);
    errhighAPD(i) = termAPD;
    errlowAPD(i) = - termAPD;
    dataMPD(i) = mean(resultsMPD);
    termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/N);
    errhighMPD(i) = termMPD;
    errlowMPD(i) = - termMPD;
    dataTT(i) = mean(resultsTT);
    termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/N);
    errhighTT(i) = termTT;
    errlowTT(i) = - termTT;
    i = i+1;
end
```

```
53 -    figure(1)
54 -    h = bar(lambda,dataPL);
55 -    hold on
56 -    grid on
57 -    title("Packet Loss")
58 -    er = errorbar(lambda,dataPL,errlowPL,errhighPL);
59 -    er.Color = [0 0 0];
60 -    er.LineStyle = 'none';
61 -    set(h, {'DisplayName'}, {'dataPL'}')
62 -    legend('Location','northwest')
63 -    hold off
64
65 -    figure(2)
66 -    h = bar(lambda,dataAPD);
67 -    hold on
68 -    grid on
69 -    title("Average Packet Delay")
70 -    er2 = errorbar(lambda,dataAPD,errlowAPD,errhighAPD);
71 -    er2.Color = [0 0 0];
72 -    er2.LineStyle = 'none';
73 -    set(h, {'DisplayName'}, {'dataAPD'}')
74 -    legend('Location','northwest')
75 -    hold off
76
77 -    figure(3)
78 -    h = bar(lambda,dataMPD);
79 -    hold on
80 -    grid on
81 -    title("Maximum Packet Delay")
82 -    er3 = errorbar(lambda,dataMPD,errlowMPD,errhighMPD);
83 -    er3.Color = [0 0 0];
84 -    er3.LineStyle = 'none';
85 -    set(h, {'DisplayName'}, {'dataMPD'}')
86 -    legend('Location','northwest')
87 -    hold off
88
89 -    figure(4)
90 -    h = bar(lambda,dataTT);
91 -    hold on
92 -    grid on
93 -    title("Total Throughput")
94 -    er4 = errorbar(lambda,dataTT,errlowTT,errhighTT);
95 -    er4.Color = [0 0 0];
96 -    er4.LineStyle = 'none';
97 -    set(h, {'DisplayName'}, {'dataTT'}')
98 -    legend('Location','northwest')
99 -    hold off
```

Figura 1: Código Matlab Task 3.a

Análise de Código:

Após a inicialização das variáveis (linhas 3 - 26) é feita a simulação para cada valor de número de pedidos por segundo (*lambda*). De seguida para os resultados das simulações são feitos os cálculos das margens de erro bem como a média dos valores obtidos para cada estatística, armazenando estes valores (linhas 28 - 51). Por fim é feito o plot dos valores obtidos (linhas 53 - 99).

Resultados e Análise:

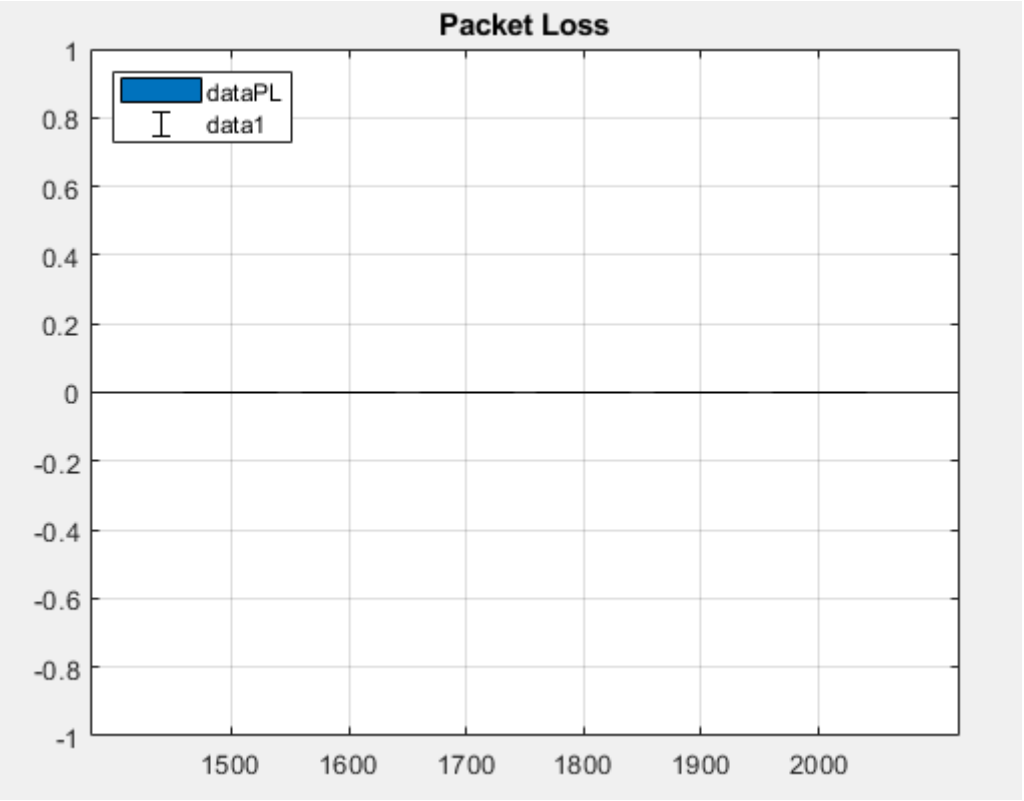

Figura 2: Packet Loss em função do número de pacotes por segundo

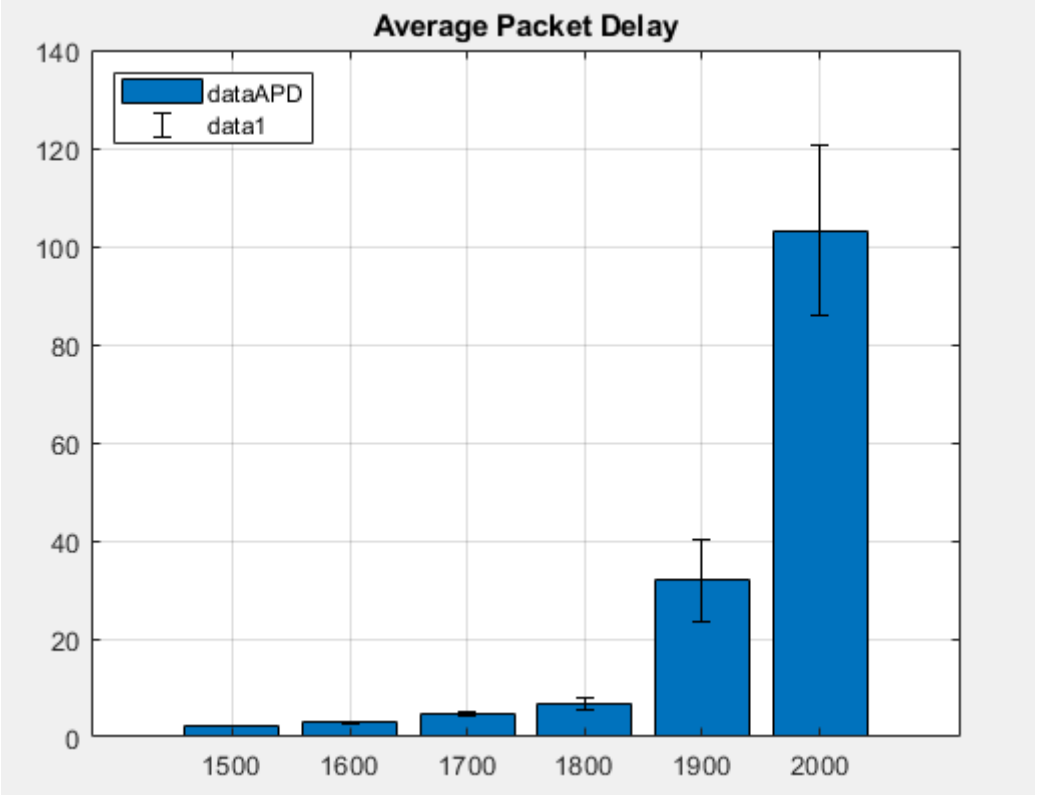

Figura 3: Average Packet Delay em função do número de pacotes por segundo

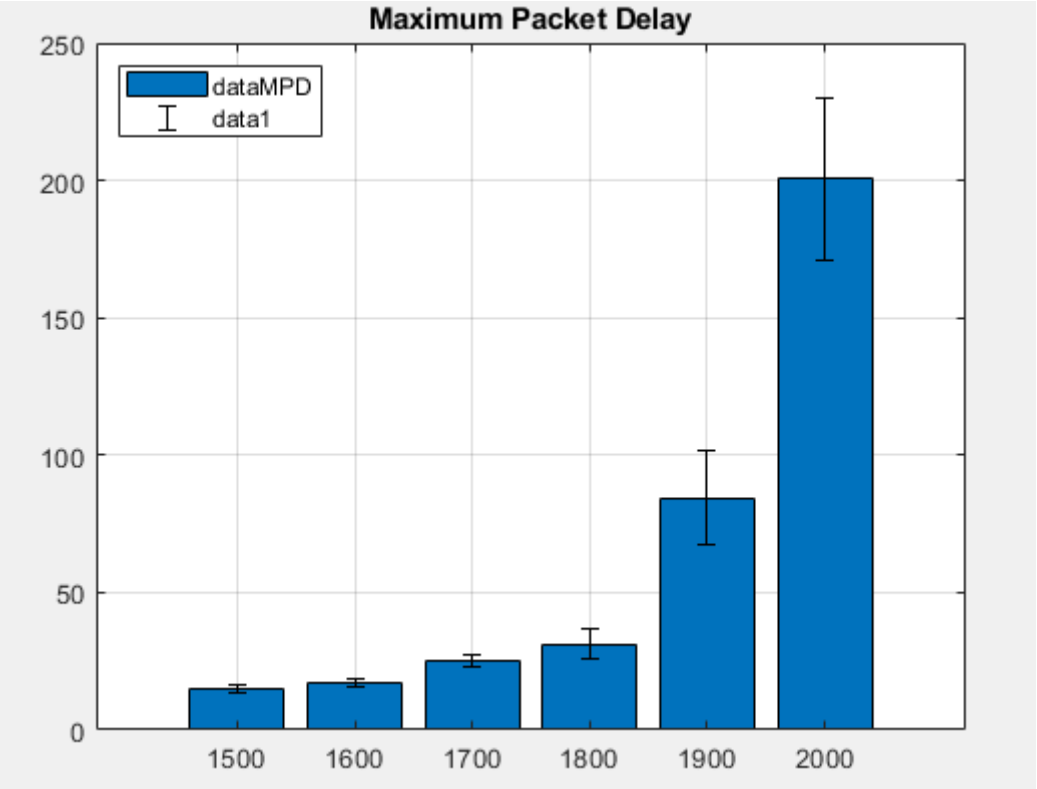

Figura 4: Maximum Packet Delay em função do número de pacotes por segundo

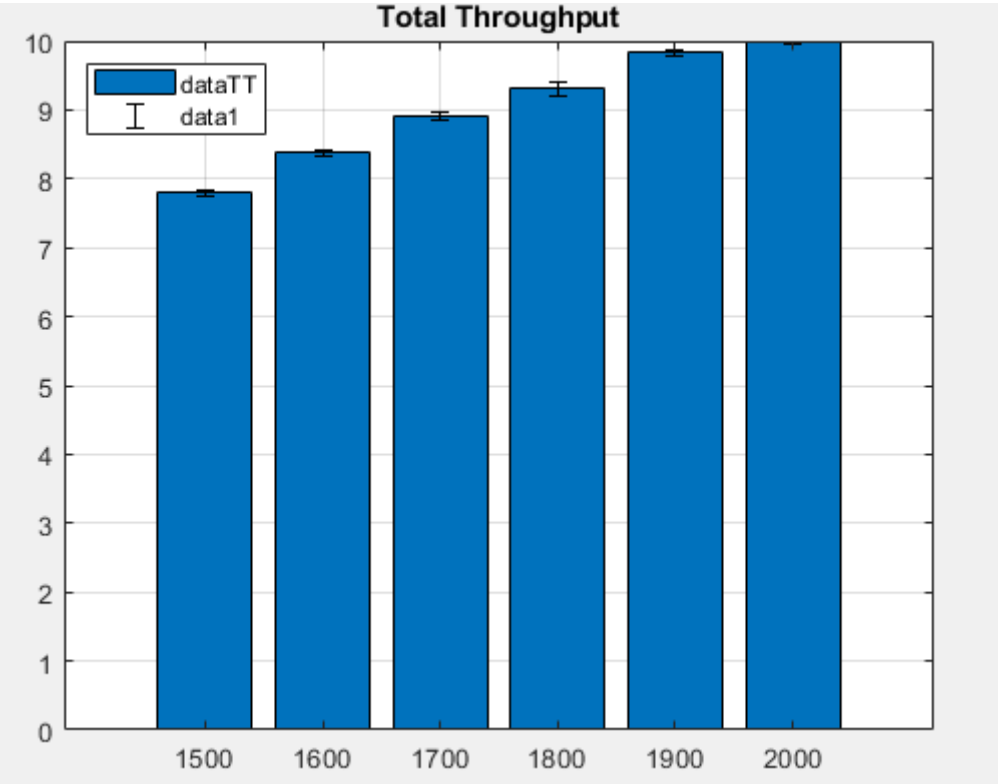

Figura 5: Total Throughput em função do número de pacotes por segundo

Da observação dos gráficos percebemos que o Packet Loss é sempre 0 para as condições de simulação indicadas dado ao facto de o BER ser 0 e a fila de espera ser grande o suficiente para não se perder nenhum pacote. Como seria de esperar o valor tanto do APD

(average packet delay), como do MPD (maximum packet delay) e o TT (Total Throughput) aumentam conforme o aumento do número de pacotes por segundo.

b)

```
P = 10000;
alfa = 0.1;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 0;
N = 40;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataPL = zeros(1,5);
dataMPD = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
errhighPL = zeros(1,5);
errhighMPD = zeros(1,5);
errhighAPD = zeros(1,5);
errhighTT = zeros(1,5);
errlowPL = zeros(1,5);
errlowMPD = zeros(1,5);
errlowAPD = zeros(1,5);
errlowTT = zeros(1,5);
i = 1;

for lam = lambda
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lam, C, f, P, b);
    end
    alfa= 0.1; %90% confidence interval%

    dataPL(i) = mean(resultsPL);
    termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/N);
    errhighPL(i) = termPL;
    errlowPL(i) = - termPL;
    dataAPD(i) = mean(resultsAPD);
    termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/N);
    errhighAPD(i) = termAPD;
    errlowAPD(i) = - termAPD;
    dataMPD(i) = mean(resultsMPD);
    termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/N);
    errhighMPD(i) = termMPD;
    errlowMPD(i) = - termMPD;
    dataTT(i) = mean(resultsTT);
    termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/N);
    errhighTT(i) = termTT;
    errlowTT(i) = - termTT;
    i = i+1;
end
```

```
52 -    figure(1)
53 -    h = bar(lambda,dataPL);
54 -    hold on
55 -    grid on
56 -    title("Packet Loss")
57 -    er = errorbar(lambda,dataPL,errlowPL,errhighPL);
58 -    er.Color = [0 0 0];
59 -    er.LineStyle = 'none';
60 -    set(h, {'DisplayName'}, {'dataPL'}')
61 -    legend('Location','northwest')
62 -    hold off
63
64 -    figure(2)
65 -    h = bar(lambda,dataAPD);
66 -    hold on
67 -    grid on
68 -    title("Average Packet Delay")
69 -    er2 = errorbar(lambda,dataAPD,errlowAPD,errhighAPD);
70 -    er2.Color = [0 0 0];
71 -    er2.LineStyle = 'none';
72 -    set(h, {'DisplayName'}, {'dataAPD'}')
73 -    legend('Location','northwest')
74 -    hold off
75
76 -    figure(3)
77 -    h = bar(lambda,dataMPD);
78 -    hold on
79 -    grid on
80 -    title("Maximum Packet Delay")
81 -    er3 = errorbar(lambda,dataMPD,errlowMPD,errhighMPD);
82 -    er3.Color = [0 0 0];
83 -    er3.LineStyle = 'none';
84 -    set(h, {'DisplayName'}, {'dataMPD'}')
85 -    legend('Location','northwest')
86 -    hold off
87
88 -    figure(4)
89 -    h = bar(lambda,dataTT);
90 -    hold on
91 -    grid on
92 -    title("Total Throughput")
93 -    er4 = errorbar(lambda,dataTT,errlowTT,errhighTT);
94 -    er4.Color = [0 0 0];
95 -    er4.LineStyle = 'none';
96 -    set(h, {'DisplayName'}, {'dataTT'}')
97 -    legend('Location','northwest')
98 -    hold off
```

Figura 6: Código Matlab Task 3.b

Análise de Código:

Código similar ao da alínea anterior, única alteração no valor de N (número de simulações) na linha 8.

Resultados e Análise:

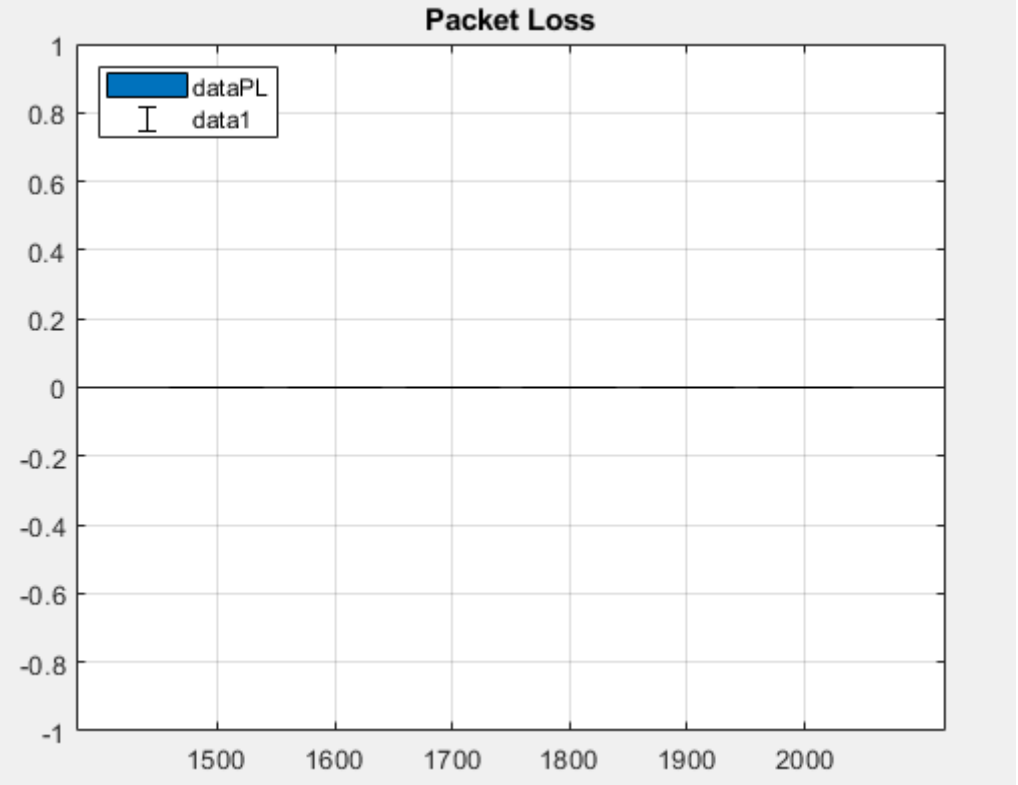

Figura 7: Packet Loss em função do número de pacotes por segundo

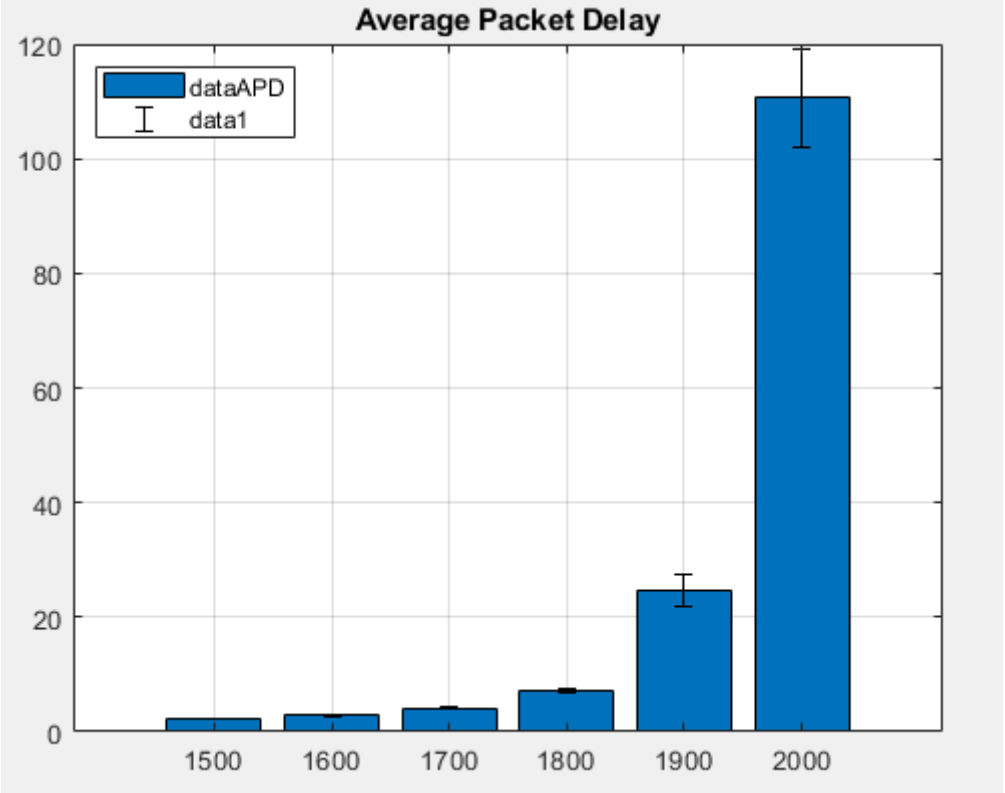

Figura 8: Average Packet Delay em função do número de pacotes por segundo

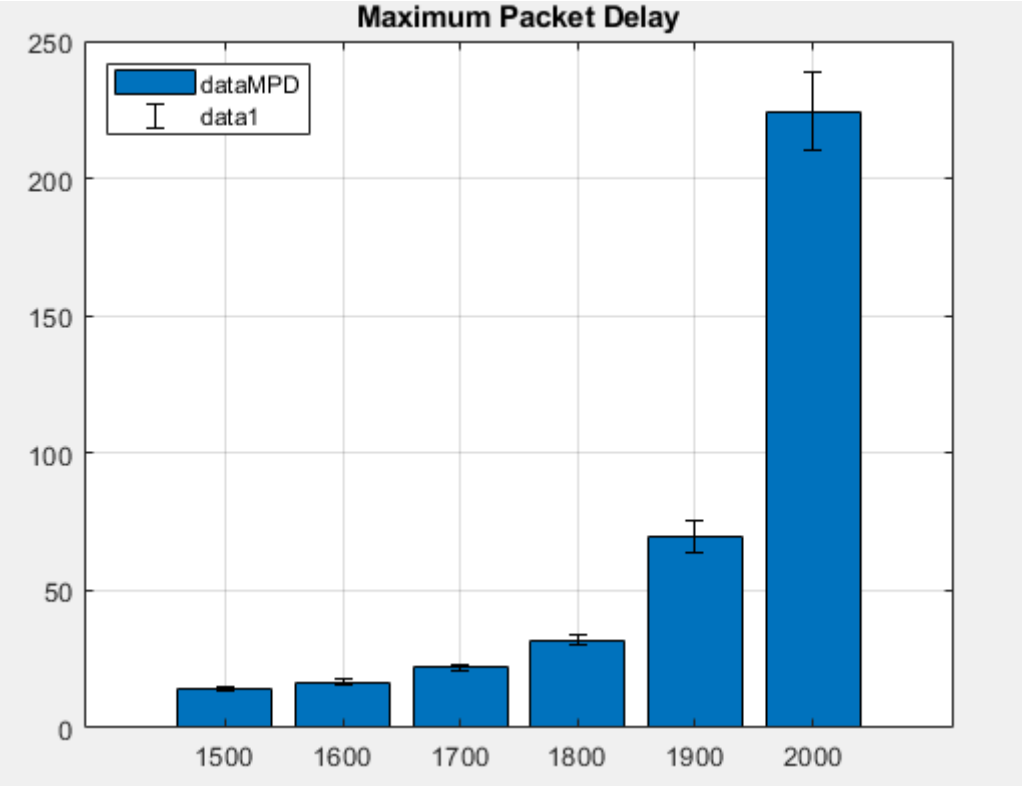

Figura 9: Maximum Packet Delay em função do número de pacotes por segundo

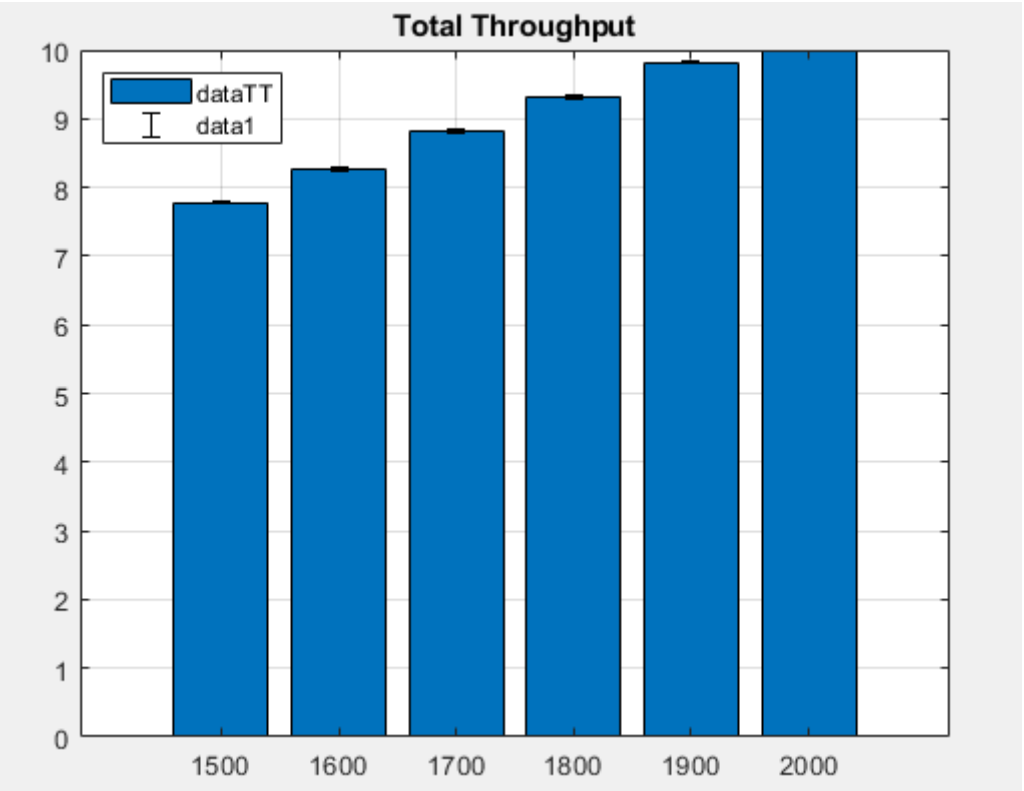

Figura 10: Total Throughput em função do número de pacotes por segundo

Resultados semelhantes aos da alínea anterior sendo que estes contêm intervalos de margem de erro menores devido ao facto de uma vez que se fizeram mais simulações os

resultados tendem a uniformizar, e a variação nos resultados de simulações diferentes são esbatidas nos cálculos de erro.

c)

```
% Task 3c

% Variables
P = 10000;
alfa = 0.1;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 0;
N = 40;
resultsPL = zeros(1,40);
resultsMPD = zeros(1,40);
resultsAPD = zeros(1,40);
resultsTT = zeros(1,40);
w_mm1 = zeros(1,5);
tt_mm1 = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
i = 1;

% MM1 WAITING QUEUE
sizes = [65:1:109 111:1:1517];
rest = 1 - 0.16 - 0.2 - 0.25; % probability of all sizes in sizes
B = (64 * 0.16 + 1518 * 0.20 + 110 * 0.25 + mean(sizes) * rest) * 8; % Average Packet Size
u = (C*1e6)/B;

for lam = lambda
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lam, C, f, P, b);
    end

    dataAPD(i) = mean(resultsAPD);
    dataTT(i) = mean(resultsTT);

    W = 1/(u - lam) * 1000;
    TT = lam * B * 1e-6;

    w_mm1(i) = W;
    tt_mm1(i) = TT;

    i = i +1;
end
```

```
44      % MG1 WAITING QUEUE
45
46 -    w_mg1 = zeros (1,5);
47 -    tt_mg1 = zeros(1,5);
48 -    S1 = (64 * 8) /10e6; %C*1e6;
49 -    S2 = (110 * 8) / 10e6; %C*1e6;
50 -    S3 = (1518 * 8) / 10e6; %C*1e6;
51 -    S4 = ((mean(sizes)) * 8) / 10e6; %C*1e6;
52 -    S42 = 0;
53
54 -    for i = [65:109 111:1517]
55 -        S42 = S42 + ((i*8)/10e6)^2;
56 -    end
57 -    S42 = S42 / length(sizes);
58 -    ES = 0.16 * S1 + 0.25 * S2 + 0.2 * S3 + (1-0.16-0.25-0.2) * S4;
59 -    ES2 = 0.16 * S1^2 + 0.25 * S2^2 + 0.2 * S3^2 + (1-0.16-0.25-0.2) * S42;
60
61 -    i=1;
62 -    for lam = lambda
63
64 -        w_mg1(i) = ((lam * ES2) / (2 * (1 - lam * ES)) + ES) * 1000;
65
66 -        tt_mg1(i) = lam * B * 1e-6 ;
67 -        i=i+1;
68
69 -    end
70
71 -    data_w = [dataAPD;w_mm1;w_mg1];
72 -    figure(1)
73 -    h = bar(lambda,data_w);
74 -    hold on
75 -    grid on
76 -    title("Average Packet Delay")
77 -    set(h, {'DisplayName'}, {'W simulation','W MM1 theoric','W MG1 theoric'}')
78 -    legend('Location','northwest')
79 -    hold off
80
81 -    data_tt = [dataTT;tt_mm1;tt_mg1];
82 -    figure(2)
83 -    h = bar(lambda,data_tt);
84 -    hold on
85 -    grid on
86 -    title("Total Throughput")
87 -    set(h, {'DisplayName'}, {'TT simulation','TT MM1 theoric','TT MG1 theoric'}')
88 -    legend('Location','northwest')
89 -    hold off
```

Figura 11: Código Matlab Task 3.c

Análise de Código:

Nas primeiras 21 linhas de código são inicializadas as variáveis que descrevem o cenário do problema, de seguida é calculada a probabilidade somada de todos os tamanhos no array “sizes” bem como o tamanho médio do pacote. Nas linhas 22 a 42 é corrido o simulator2 e armazenados os resultados da simulação, bem como calculados os valores teóricos de throughput e average packet delay, para uma fila de espera do tipo M/M/1, para cada valor do número de pacotes por segundo.

Resultados e Análise:

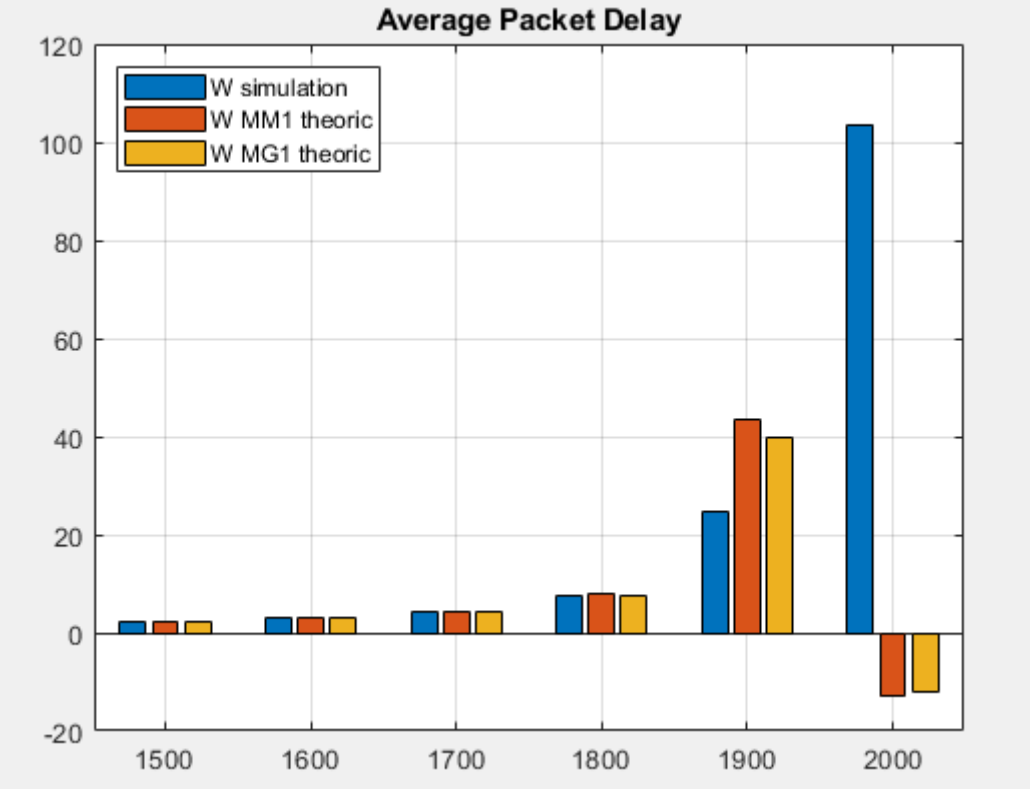

Figura 12: Average Packet Delay

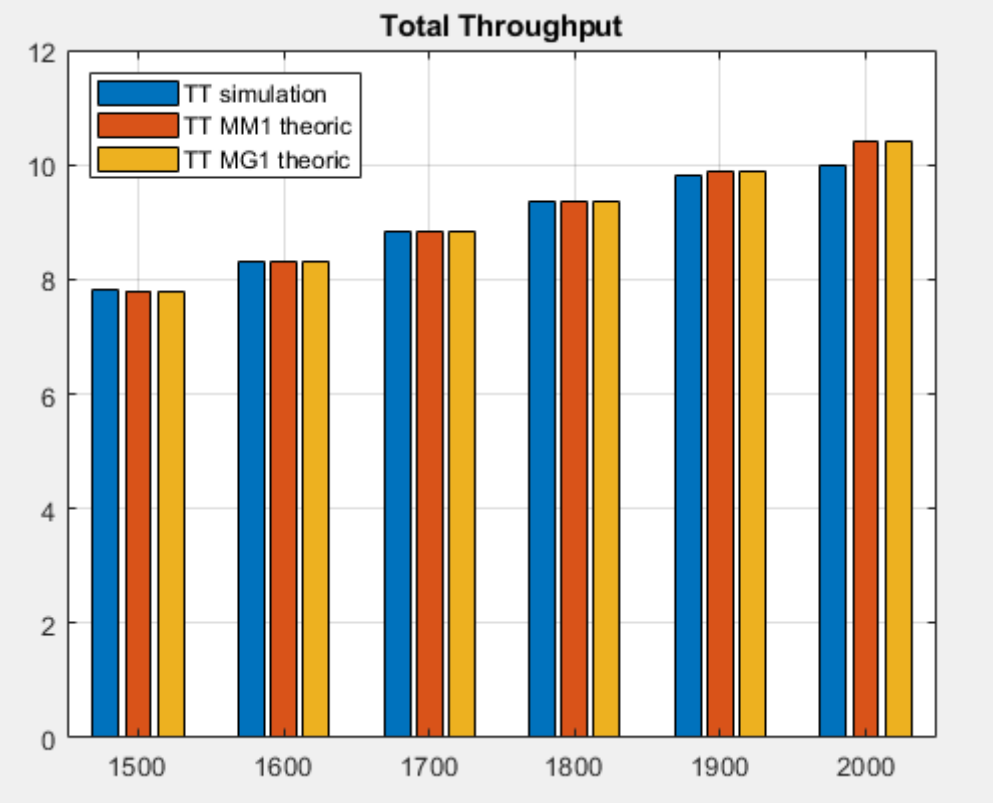

Figura 13: Total Throughput

Como se pode observar nos resultados obtidos, os valores obtidos por simulação são em geral muito próximos dos valores teóricos. No gráfico do Average Packet Delay é possível observar que os valores teóricos são em geral similares aos valores de simulação, contudo, quando o número de pacotes por segundo chega aos 2000 os valores teóricos (para M/M/1 e M/G/1) são negativos. Isto deve-se ao facto de ser aplicada a fórmula de cálculo onde sempre que o valor de lambda é superior ao valor de miu (variável "u"), o valor médio torna-se negativo, pois W = 1/(miu - lambda).

d)

```
P = 10000;
alfa = 0.1;
lambda = 1800;
C = 10;
f_values = 2500:2500:20000;
b = 0;
N = 40;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataPL = zeros(1,5);
dataMPD = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
errhighPL = zeros(1,5);
errhighMPD = zeros(1,5);
errhighAPD = zeros(1,5);
errhighTT = zeros(1,5);
errlowPL = zeros(1,5);
errlowMPD = zeros(1,5);
errlowAPD = zeros(1,5);
errlowTT = zeros(1,5);
i = 1;

for f = f_values
    for it = 1:N
    [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lambda, C, f, P, b);
    end
    alfa= 0.1; %90% confidence interval%

    dataPL(i) = mean(resultsPL);
    termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/N);
    errhighPL(i) = termPL;
    errlowPL(i) = - termPL;
    dataAPD(i) = mean(resultsAPD);
    termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/N);
    errhighAPD(i) = termAPD;
    errlowAPD(i) = - termAPD;
    dataMPD(i) = mean(resultsMPD);
    termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/N);
    errhighMPD(i) = termMPD;
    errlowMPD(i) = - termMPD;
    dataTT(i) = mean(resultsTT);
    termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/N);
    errhighTT(i) = termTT;
    errlowTT(i) = - termTT;
    i = i+1;
end
```

```
53 -    figure(1)
54 -    h = bar(f_values,dataPL);
55 -    hold on
56 -    grid on
57 -    title("Packet Loss")
58 -    er = errorbar(f_values,dataPL,errlowPL,errhighPL);
59 -    er.Color = [0 0 0];
60 -    er.LineStyle = 'none';
61 -    set(h, {'DisplayName'}, {'dataPL'}')
62 -    legend('Location','northwest')
63 -    hold off
64
65 -    figure(2)
66 -    h = bar(f_values,dataAPD);
67 -    hold on
68 -    grid on
69 -    title("Average Packet Delay")
70 -    er2 = errorbar(f_values,dataAPD,errlowAPD,errhighAPD);
71 -    er2.Color = [0 0 0];
72 -    er2.LineStyle = 'none';
73 -    set(h, {'DisplayName'}, {'dataAPD'}')
74 -    legend('Location','northwest')
75 -    hold off
76
77 -    figure(3)
78 -    h = bar(f_values,dataMPD);
79 -    hold on
80 -    grid on
81 -    title("Maximum Packet Delay")
82 -    er3 = errorbar(f_values,dataMPD,errlowMPD,errhighMPD);
83 -    er3.Color = [0 0 0];
84 -    er3.LineStyle = 'none';
85 -    set(h, {'DisplayName'}, {'dataMPD'}')
86 -    legend('Location','northwest')
87 -    hold off
88
89 -    figure(4)
90 -    h = bar(f_values,dataTT);
91 -    hold on
92 -    grid on
93 -    title("Total Throughput")
94 -    er4 = errorbar(f_values,dataTT,errlowTT,errhighTT);
95 -    er4.Color = [0 0 0];
96 -    er4.LineStyle = 'none';
97 -    set(h, {'DisplayName'}, {'dataTT'}')
98 -    legend('Location','northwest')
99 -    hold off
```

Figura 14: Código Matlab Task 3.d

Análise de Código:

Após a inicialização das variáveis (linhas 3 - 26) é feita a simulação para cada valor do tamanho da fila de espera em Bytes (*f_values*) num ciclo *for*. De seguida para os resultados das simulações são feitos os cálculos das margens de erro bem como a média dos valores obtidos para cada estatística, armazenando estes valores em arrays (linhas 28 - 51). Por fim é feito o plot dos valores obtidos (linhas 53 - 99).

Resultados e Análise:

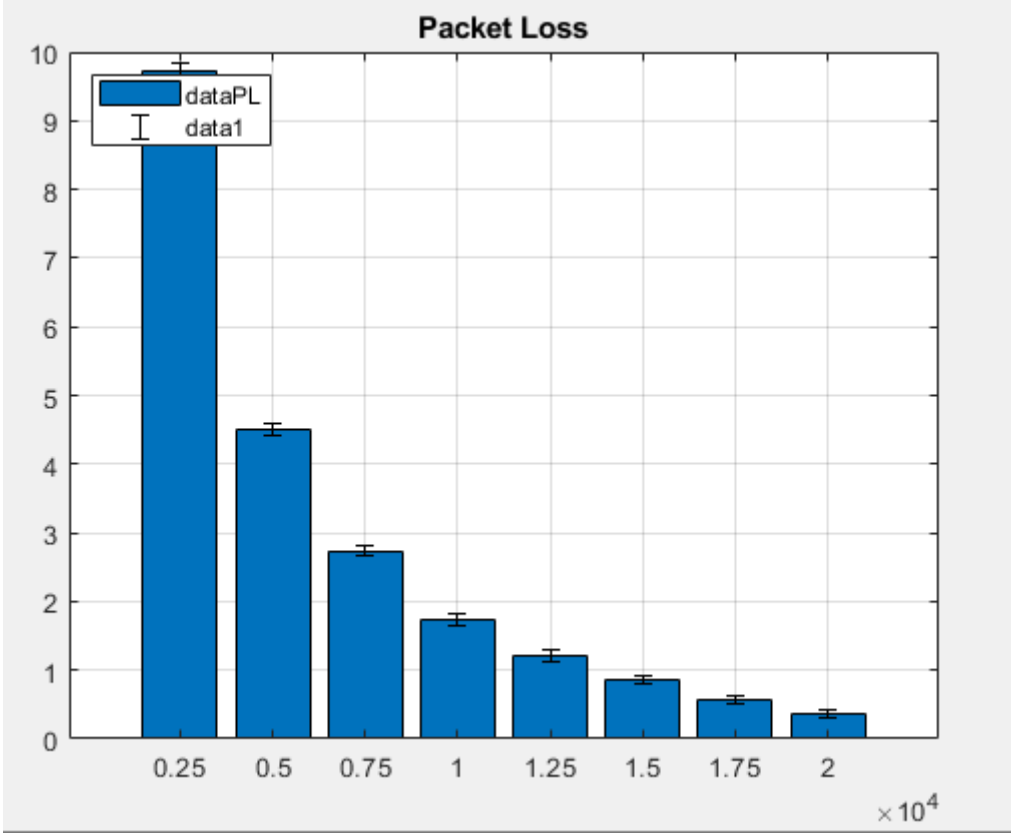

Figura 15: Packet Loss em função do tamanho da fila de espera em Bytes

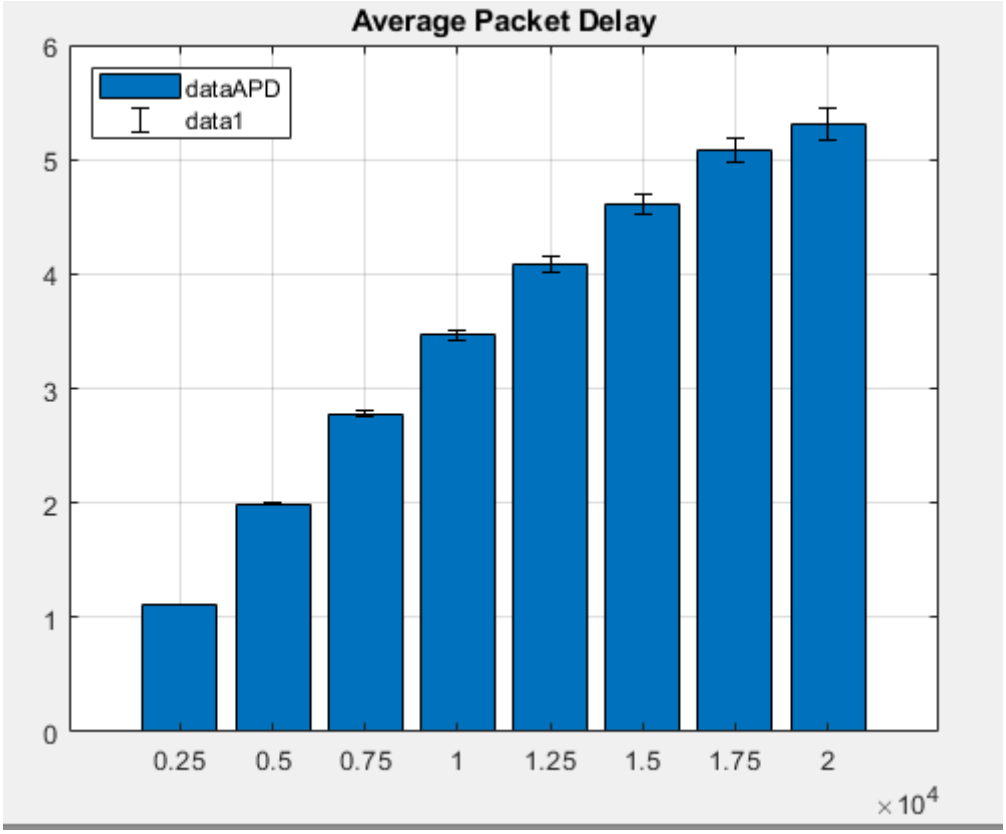

Figura 16: Average Packet Delay em função do tamanho da fila de espera em Bytes

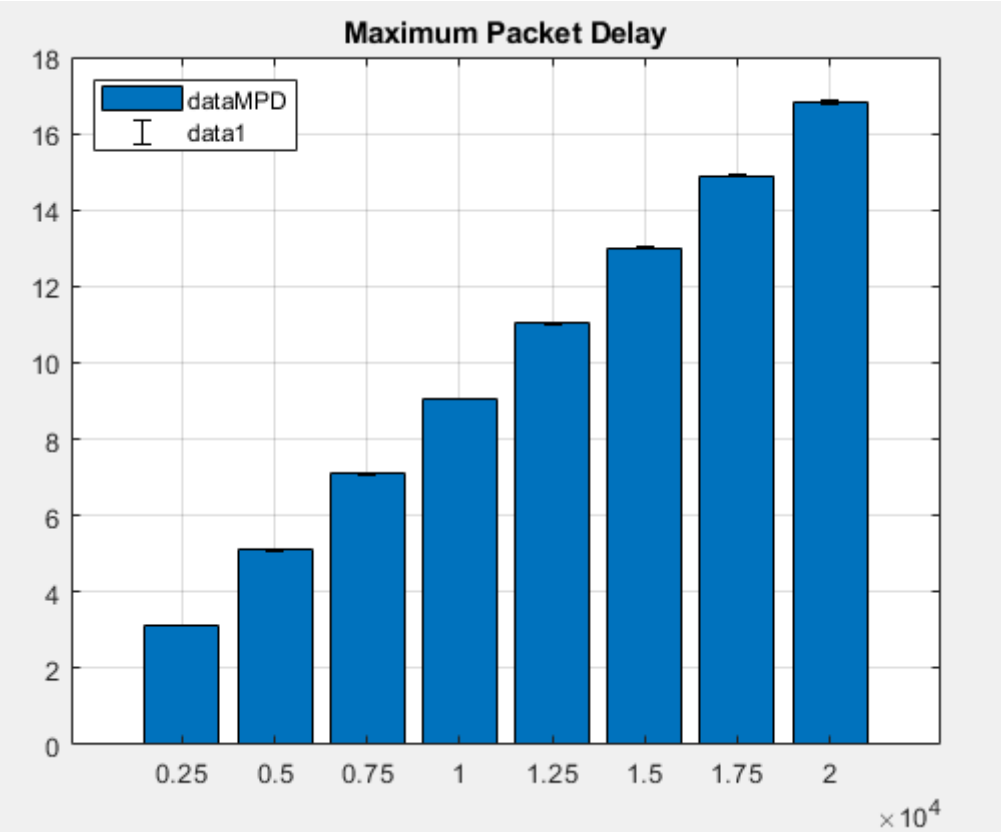

Figura 17: Maximum Packet Delay em função do tamanho da fila de espera em Bytes

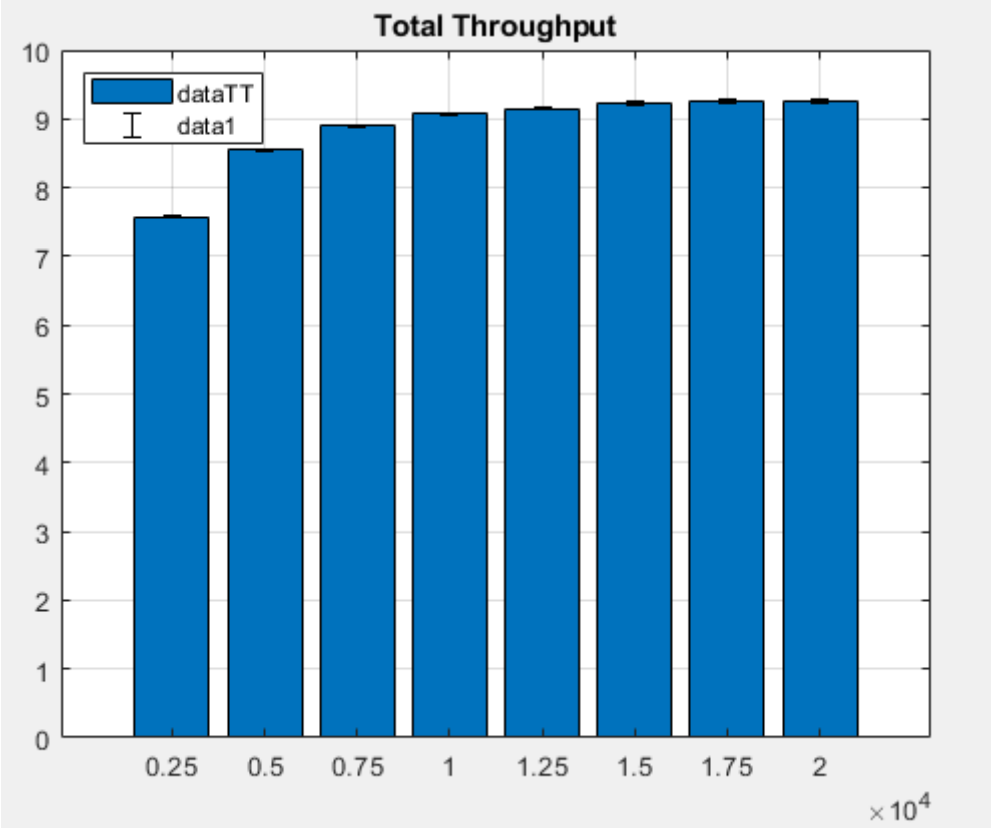

Figura 18: Total Throughput em função do tamanho da fila de espera em Bytes

Da análise dos gráficos obtidos é possível observar que o valor médio de Packet Loss, para cada valor do tamanho da fila de espera vai sendo cada vez menor à medida que o tamanho aumenta, o que é espectável pois à medida que o número de pacotes que podem ficar “armazenados” na fila de espera aumenta, menos pacotes serão perdidos por falta de espaço. Para os valores de APD e MPD também como seria expectável, à medida que mais pacotes ficam em espera na fila, maior é o tempo de espera máxima e o tempo de espera médio.

e)

```
pl_mmm1 = zeros(1,8);
w_mmm1 = zeros (1,8);
tt_mmm1 = zeros(1,8);

P = 10000;
lambda = 1800;
C = 10;
f_values = 2500:2500:20000
b = 0;
N = 40;

aux2= [65:109 111:1517];
B = 64*0.16 + 110*0.25 + 1518*0.2 + (mean(aux2))*(1-0.16-0.25-0.2);
miu = C*1e6/(B*8);
k=1
for f = f_values
    m = fix(f / B) + 1

    sum = 0;
    for j = 0:m
        sum = sum + ((lambda/miu)^j);
    end

    pl_mmm1(k) = (((lambda/miu)^m)/sum) * 100;

    sum2 = 0;
    for a = 0:m
        sum2 = sum2 + (a * (lambda/miu)^a);
    end

    L = sum2 / sum;
    w_mmm1(k) = L / ((lambda * (1 - pl_mmm1(k)/100))) * 1000;
    tt_mmm1(k) = lambda * 8 * B * 1e-6 * (1 - pl_mmm1(k)/100);
    k=k+1
end

resultsPL = zeros(1,40);
resultsMPD = zeros(1,40);
resultsAPD = zeros(1,40);
resultsTT = zeros(1,40);

dataPL = zeros(1,8);
dataMPD = zeros(1,8);
dataAPD = zeros(1,8);
dataTT = zeros(1,8);

i = 1;
```

```
51 -  for f = f_values
52 -      for it = 1:N
53 -      [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lambda, C, f, P, b);
54 -      end
55 -      dataPL(i) = mean(resultsPL);
56 -      dataAPD(i) = mean(resultsAPD);
57 -      dataMPD(i) = mean(resultsMPD);
58 -      dataTT(i) = mean(resultsTT);
59 -      i = i+1;
60 -  end
61
62 -  data_pl = [dataPL;pl_mmm1]
63 -  figure(1)
64 -  h = bar(f_values,data_pl);
65 -  hold on
66 -  grid on
67 -  title("Packet Loss mMm1")
68 -  set(h, {'DisplayName'}, {'W sim','W theoric'}')
69 -  legend('Location','northwest')
70 -  hold off
71
72 -  data_w = [dataAPD;w_mmm1];
73 -  figure(2)
74 -  h = bar(f_values,data_w);
75 -  hold on
76 -  grid on
77 -  title("Average Packet Delay mg1")
78 -  set(h, {'DisplayName'}, {'W sim','W theoric'}')
79 -  legend('Location','northwest')
80 -  hold off
81
82 -  data_tt = [dataTT;tt_mmm1];
83 -  figure(3)
84 -  h = bar(f_values,data_tt);
85 -  hold on
86 -  grid on
87 -  title("Total Throughput mg1")
88 -  set(h, {'DisplayName'}, {'TT sim','TT theoric'}')
89 -  legend('Location','northwest')
90 -  hold off
```

Figura 19: Código MatLab Task 3.e

Análise de Código:

Nas primeiras 14 linhas de código são inicializadas as variáveis conforme exposto no enunciado, de seguida entre as linhas 15 a 37 são calculados os valores teóricos aplicando as fórmulas presentes nos Slides do Módulo 2. Nas seguintes 10 linhas (39 - 49) são inicializados os arrays para armazenar os resultados do simulador. Após isto é executado o simulador 2 para os diferentes valores do tamanho da fila de espera e armazenados os resultados (linhas 51-60). Por fim, são gerados os gráficos dos resultados teóricos e de simulação obtidos, nas linhas 62-90.

Resultados e Análise:

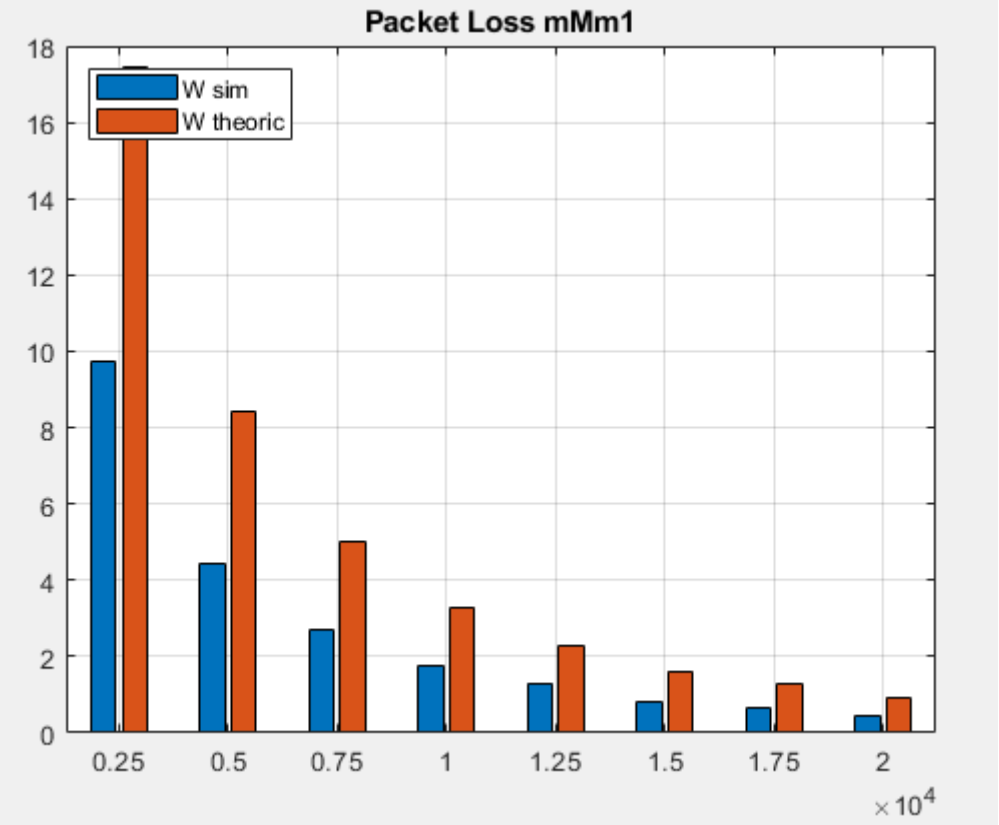

Figura 20: Packet Loss em função do tamanho da fila de espera em Bytes

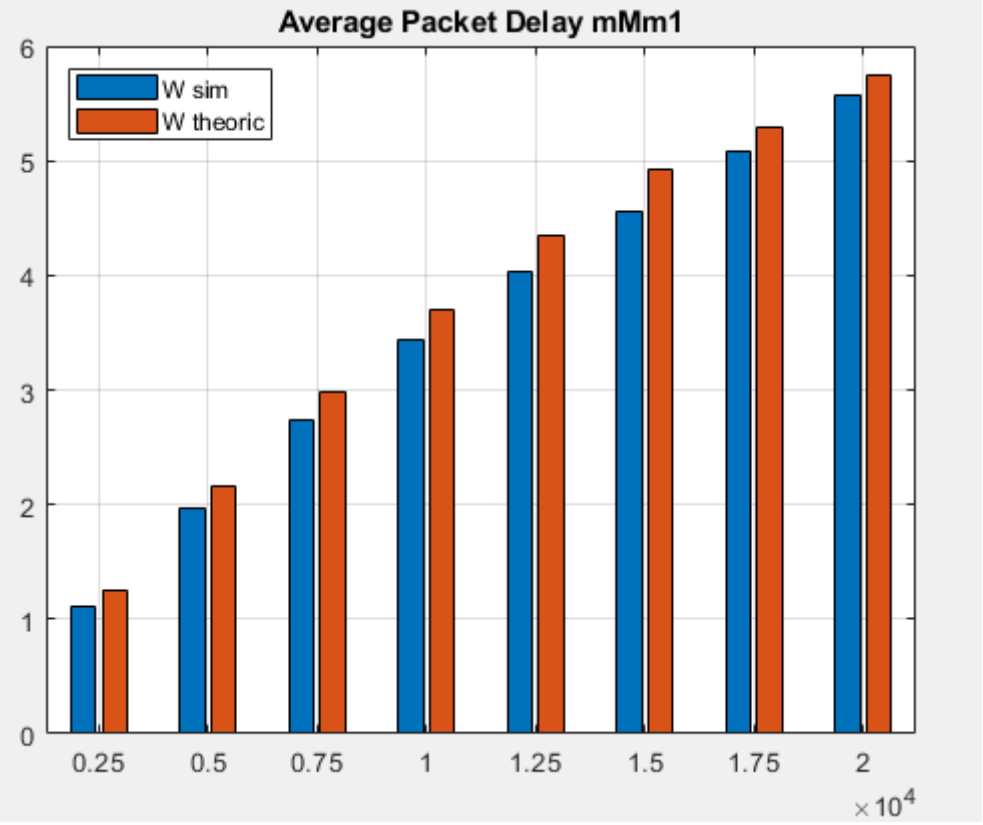

Figura 21: Average Packet Delay em função do tamanho da fila de espera em Bytes

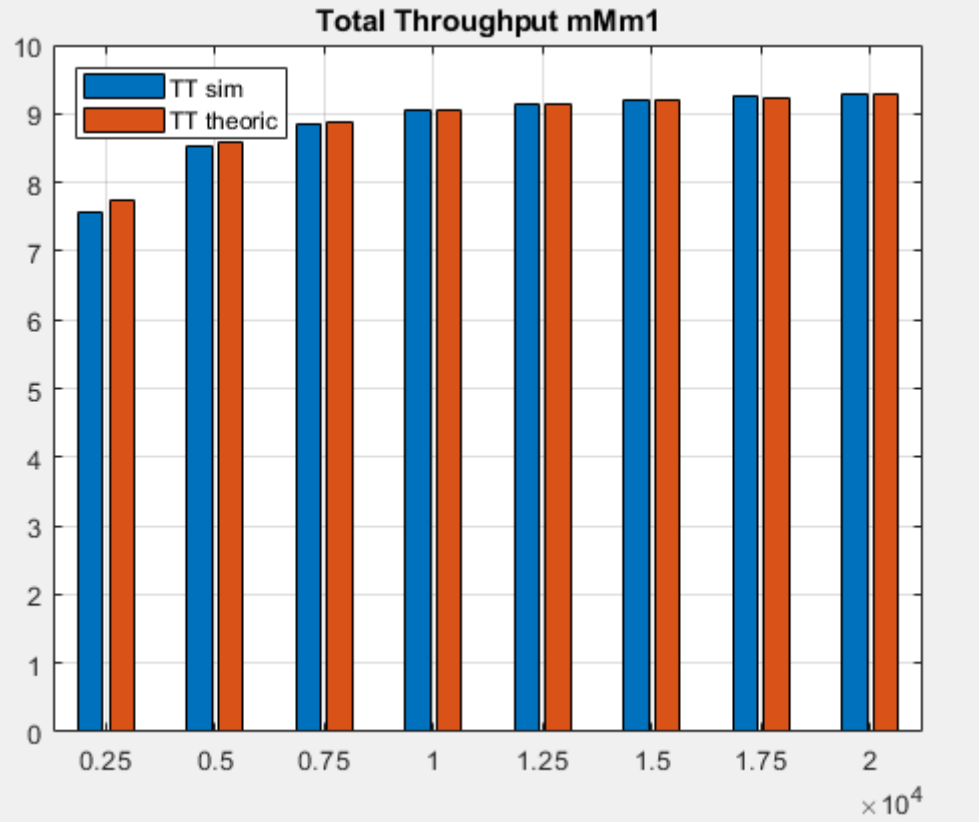

Figura 22: Total Throughput MG1 em função do tamanho da fila de espera em Bytes

Pela observação dos gráficos vemos que os valores de simulação para o PL são cerca de metade dos valores teóricos, no entanto para o APD e TT os valores de simulação são muito similares aos valores teóricos.

f)

```
P = 10000;
alfa = 0.1;
lambda = 1800;
C = 10;
f_values = 2500:2500:20000;
b = 10e-5;
N = 40;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataPL = zeros(1,5);
dataMPD = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
errhighPL = zeros(1,5);
errhighMPD = zeros(1,5);
errhighAPD = zeros(1,5);
errhighTT = zeros(1,5);
errlowPL = zeros(1,5);
errlowMPD = zeros(1,5);
errlowAPD = zeros(1,5);
errlowTT = zeros(1,5);
i = 1;

for f = f_values
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lambda, C, f, P, b);
    end
    alfa= 0.1; %90% confidence interval%

    dataPL(i) = mean(resultsPL);
    termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/N);
    errhighPL(i) = termPL;
    errlowPL(i) = - termPL;
    dataAPD(i) = mean(resultsAPD);
    termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/N);
    errhighAPD(i) = termAPD;
    errlowAPD(i) = - termAPD;
    dataMPD(i) = mean(resultsMPD);
    termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/N);
    errhighMPD(i) = termMPD;
    errlowMPD(i) = - termMPD;
    dataTT(i) = mean(resultsTT);
    termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/N);
    errhighTT(i) = termTT;
    errlowTT(i) = - termTT;
    i = i+1;
end
```

```
53 -    figure(1)
54 -    h = bar(f_values,dataPL);
55 -    hold on
56 -    grid on
57 -    title("Packet Loss")
58 -    er = errorbar(f_values,dataPL,errlowPL,errhighPL);
59 -    er.Color = [0 0 0];
60 -    er.LineStyle = 'none';
61 -    set(h, {'DisplayName'}, {'dataPL'}')
62 -    legend('Location','northwest')
63 -    hold off
64
65 -    figure(2)
66 -    h = bar(f_values,dataAPD);
67 -    hold on
68 -    grid on
69 -    title("Average Packet Delay")
70 -    er2 = errorbar(f_values,dataAPD,errlowAPD,errhighAPD);
71 -    er2.Color = [0 0 0];
72 -    er2.LineStyle = 'none';
73 -    set(h, {'DisplayName'}, {'dataAPD'}')
74 -    legend('Location','northwest')
75 -    hold off
76
77 -    figure(3)
78 -    h = bar(f_values,dataMPD);
79 -    hold on
80 -    grid on
81 -    title("Maximum Packet Delay")
82 -    er3 = errorbar(f_values,dataMPD,errlowMPD,errhighMPD);
83 -    er3.Color = [0 0 0];
84 -    er3.LineStyle = 'none';
85 -    set(h, {'DisplayName'}, {'dataMPD'}')
86 -    legend('Location','northwest')
87 -    hold off
88
89 -    figure(4)
90 -    h = bar(f_values,dataTT);
91 -    hold on
92 -    grid on
93 -    title("Total Throughput")
94 -    er4 = errorbar(f_values,dataTT,errlowTT,errhighTT);
95 -    er4.Color = [0 0 0];
96 -    er4.LineStyle = 'none';
97 -    set(h, {'DisplayName'}, {'dataTT'}')
98 -    legend('Location','northwest')
99 -    hold off
```

Figura 23: Código Matlab Task 3.f

Análise de Código:

Código similar ao da alínea 3d, onde apenas foi alterado o valor de Bit Error Rate na linha 8 para 10^-5.

Resultados e Análise:

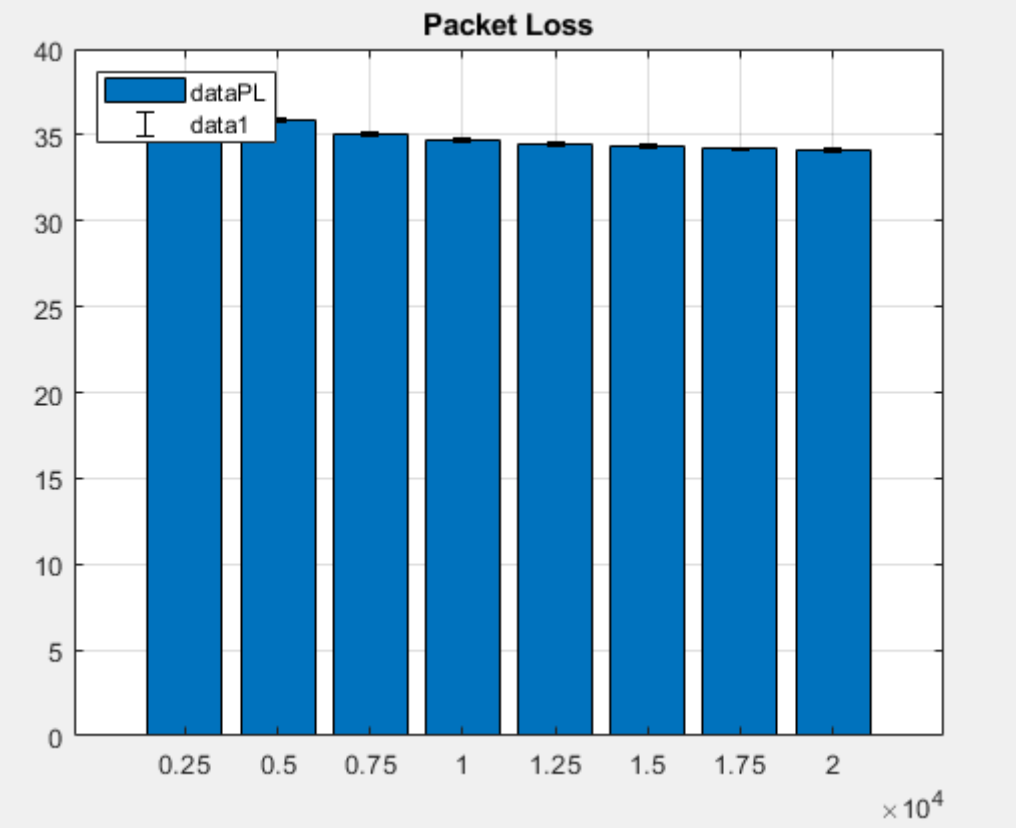

Figura 24: Packet Loss em função do tamanho da fila de espera em Bytes

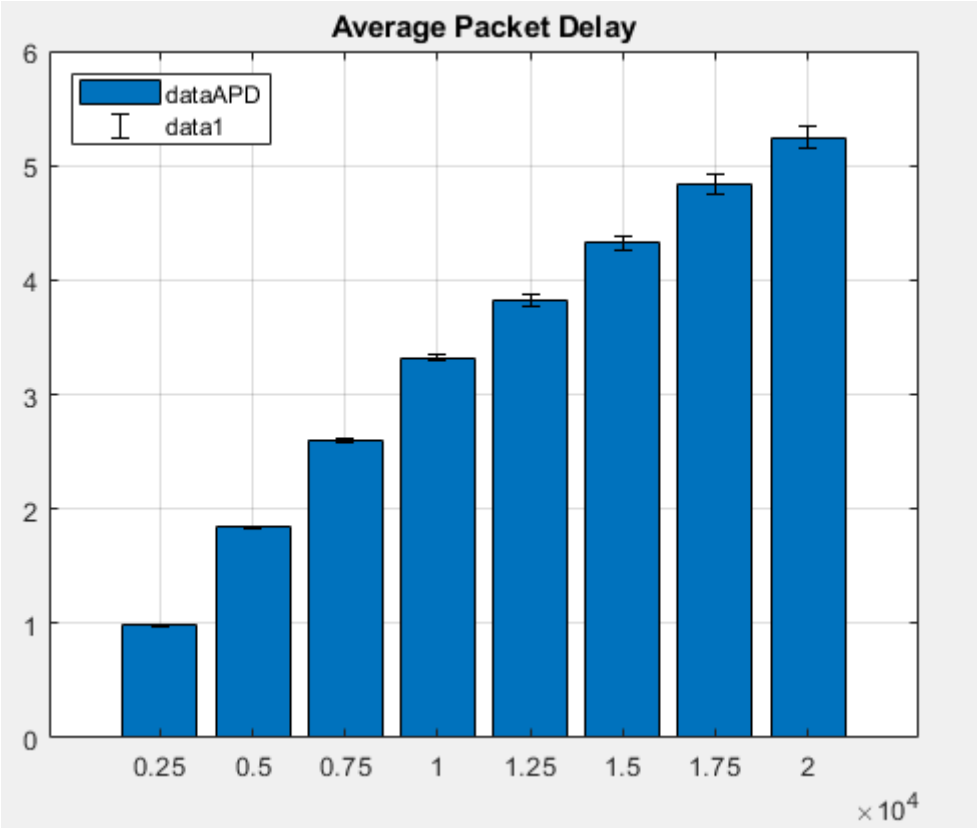

Figura 25: Average Packet Delay em função do tamanho da fila de espera em Bytes

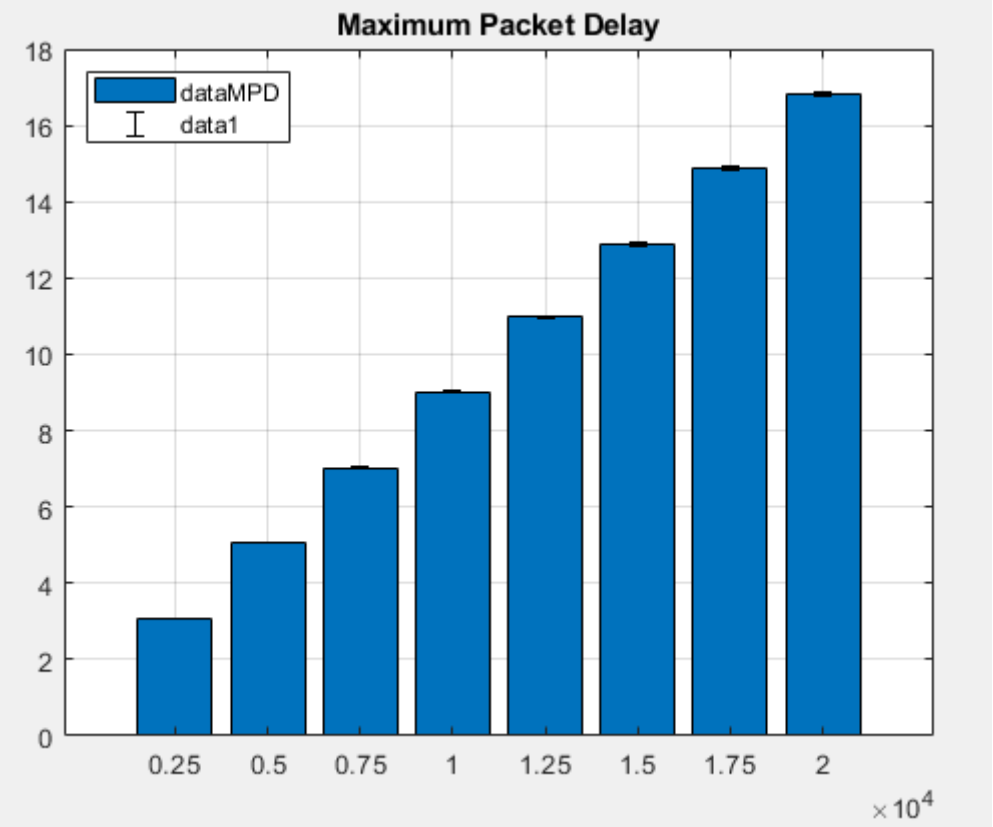

Figura 26: Average Packet Loss em função do tamanho da fila de espera em Bytes

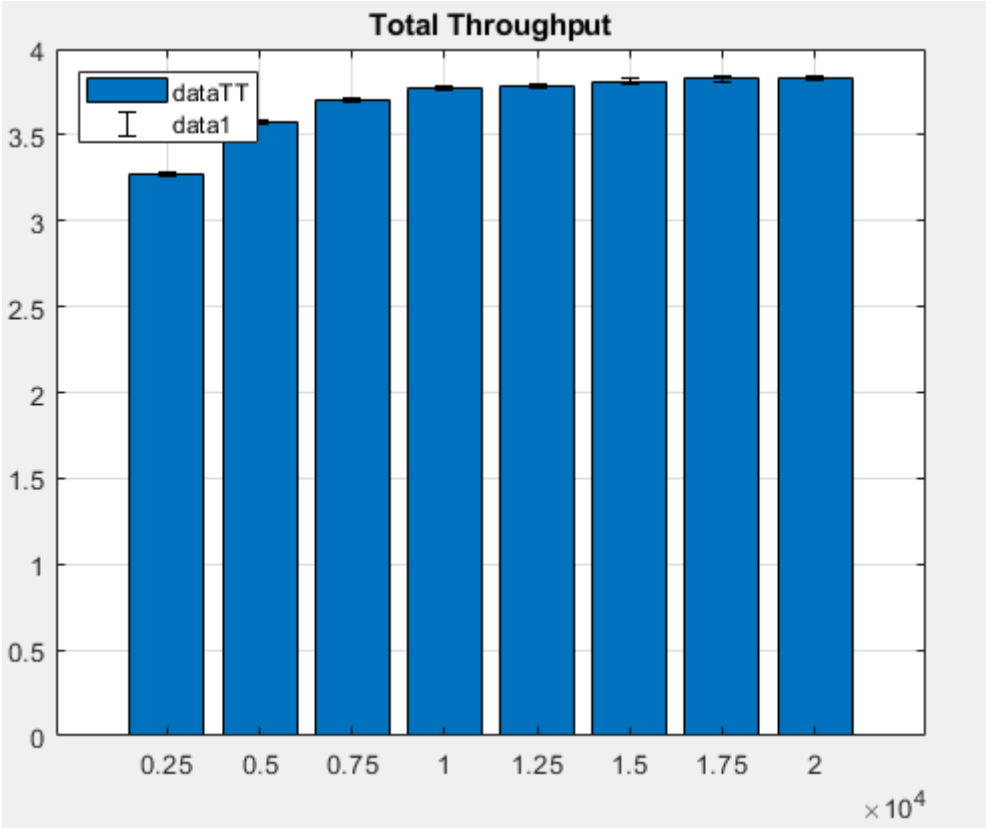

Figura 27: Total Throughput em função do tamanho da fila de espera em Bytes

Conforme o tamanho da fila de espera aumenta, a probabilidade de Packet Loss diminui ligeiramente uma vez que os pacotes são menos prováveis de ser perdidos por falta de espaço na fila, logo o Packet Loss deve se ao Bit Error Rate. Para os valores de APD e MPD seguem o mesmo comportamento na alínea d, pois há medida que mais pacotes ficam em espera na fila, maior é o tempo de espera máxima e o tempo de espera médio.

g)

```
P = 10000;
alfa = 0.1;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 10e-5;
N = 40;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataPL = zeros(1,5);
dataMPD = zeros(1,5);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
errhighPL = zeros(1,5);
errhighMPD = zeros(1,5);
errhighAPD = zeros(1,5);
errhighTT = zeros(1,5);
errlowPL = zeros(1,5);
errlowMPD = zeros(1,5);
errlowAPD = zeros(1,5);
errlowTT = zeros(1,5);
i = 1;

for lam = lambda
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lam, C, f, P, b);
    end
    alfa= 0.1; %90% confidence interval%

    dataPL(i) = mean(resultsPL);
    termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/N);
    errhighPL(i) = termPL;
    errlowPL(i) = - termPL;
    dataAPD(i) = mean(resultsAPD);
    termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/N);
    errhighAPD(i) = termAPD;
    errlowAPD(i) = - termAPD;
    dataMPD(i) = mean(resultsMPD);
    termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/N);
    errhighMPD(i) = termMPD;
    errlowMPD(i) = - termMPD;
    dataTT(i) = mean(resultsTT);
    termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/N);
    errhighTT(i) = termTT;
    errlowTT(i) = - termTT;
    i = i+1;
end
```

```
53 -    figure(1)
54 -    h = bar(lambda,dataPL);
55 -    hold on
56 -    grid on
57 -    title("Packet Loss")
58 -    er = errorbar(lambda,dataPL,errlowPL,errhighPL);
59 -    er.Color = [0 0 0];
60 -    er.LineStyle = 'none';
61 -    set(h, {'DisplayName'}, {'dataPL'}')
62 -    legend('Location','northwest')
63 -    hold off
64
65 -    figure(2)
66 -    h = bar(lambda,dataAPD);
67 -    hold on
68 -    grid on
69 -    title("Average Packet Delay")
70 -    er2 = errorbar(lambda,dataAPD,errlowAPD,errhighAPD);
71 -    er2.Color = [0 0 0];
72 -    er2.LineStyle = 'none';
73 -    set(h, {'DisplayName'}, {'dataAPD'}')
74 -    legend('Location','northwest')
75 -    hold off
76
77 -    figure(3)
78 -    h = bar(lambda,dataMPD);
79 -    hold on
80 -    grid on
81 -    title("Maximum Packet Delay")
82 -    er3 = errorbar(lambda,dataMPD,errlowMPD,errhighMPD);
83 -    er3.Color = [0 0 0];
84 -    er3.LineStyle = 'none';
85 -    set(h, {'DisplayName'}, {'dataMPD'}')
86 -    legend('Location','northwest')
87 -    hold off
88
89 -    figure(4)
90 -    h = bar(lambda,dataTT);
91 -    hold on
92 -    grid on
93 -    title("Total Throughput")
94 -    er4 = errorbar(lambda,dataTT,errlowTT,errhighTT);
95 -    er4.Color = [0 0 0];
96 -    er4.LineStyle = 'none';
97 -    set(h, {'DisplayName'}, {'dataTT'}')
98 -    legend('Location','northwest')
99 -    hold off
```

Figura 28: Código Matlab Task 3.g

Análise de Código:

Código similar ao da alínea 3.b sendo que a única alteração relevante se encontra na linha 8, onde foi alterado o valor de bit error rate para 10^-5.

Resultados e Análise:

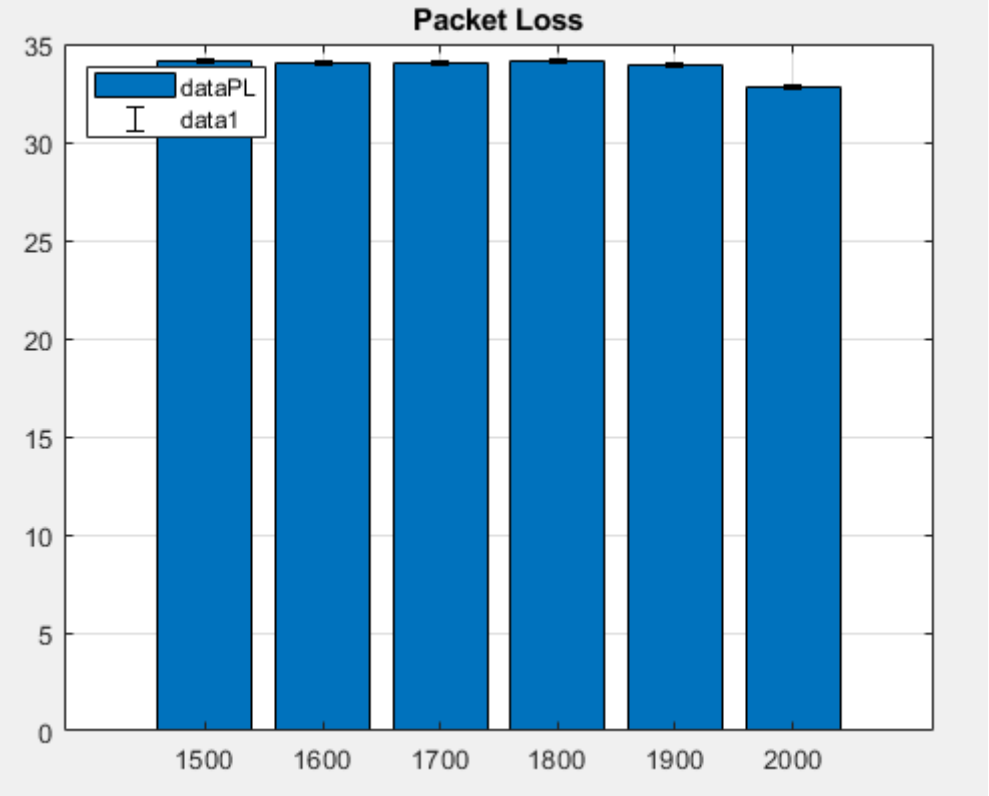

Figura 29: Packet Loss em função do número de pacotes por segundo

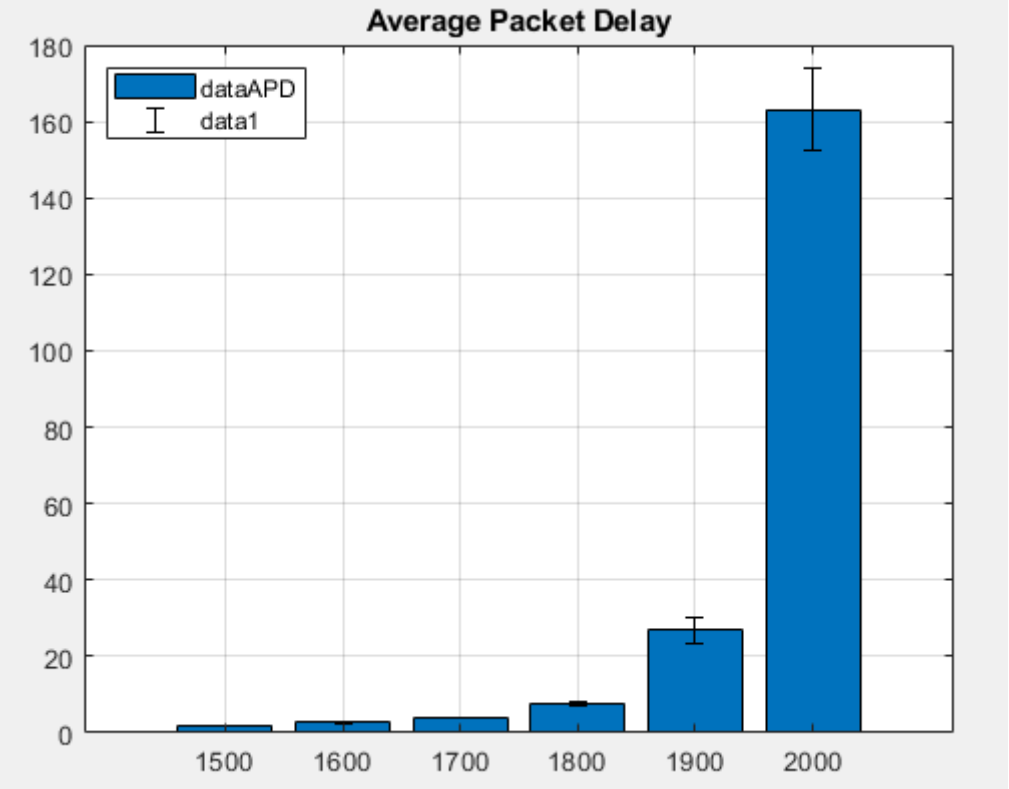

Figura 30: Average Packet Delay em função do número de pacotes por segundo

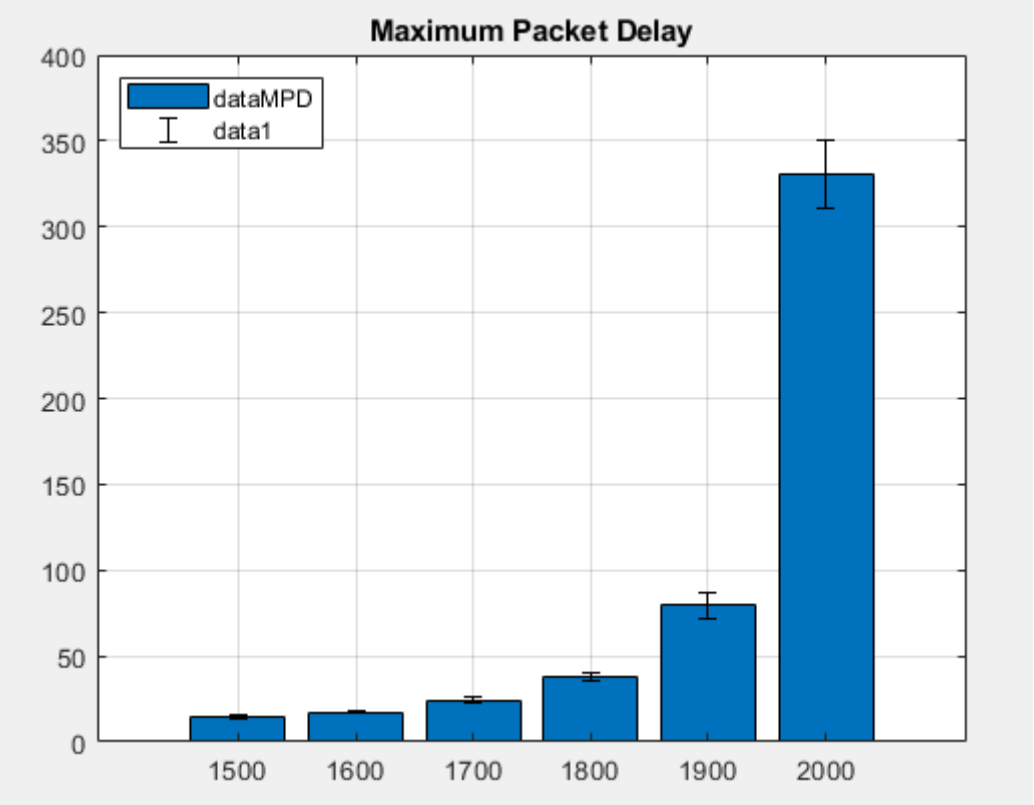

Figura 31: Maximum Packet Delay em função do número de pacotes por segundo

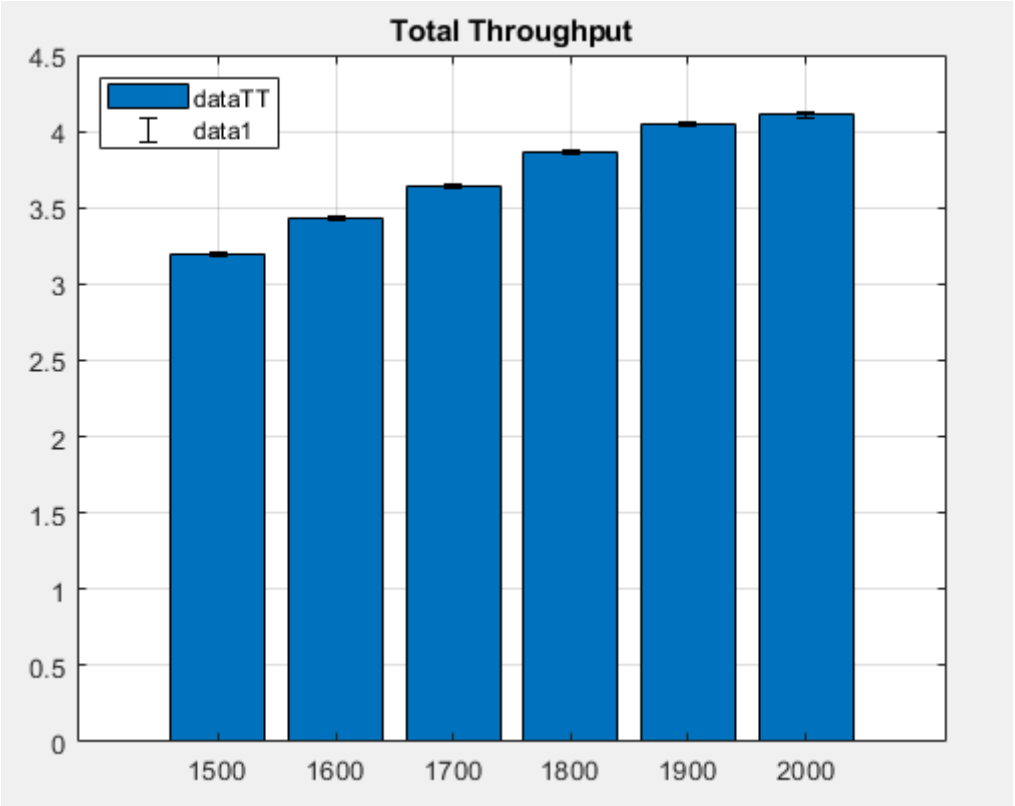

Figura 32: Total Throughput em função do número de pacotes por segundo

Resultados similares aos da alínea b, à exceção do Packet Loss e do Total Throughput, o Packet Loss agora deixou de ser 0, uma vez que existe agora probabilidade de os pacotes

terem erros com o Bit Error Rate sendo 10^-5. O valor de Total Throughput, diminuiu comparativamente ao valor da alínea b, uma vez que, como alguns pacotes são perdidos devido a erro, não são contabilizados para o throughput do simulador.

h)

```
P = 10000;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 10e-5;
N = 40;
resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);
dataAPD = zeros(1,5);
dataTT = zeros(1,5);
i = 1;

for lam = lambda
    for it = 1:N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator2(lam, C, f, P, b);
    end
    dataAPD(i) = mean(resultsAPD);
    dataTT(i) = mean(resultsTT);
    i = i+1;
end

% MG1 WAITING QUEUE
sizes = [65:1:109 111:1:1517];
rest = 1 - 0.16 - 0.2 - 0.25; % probability of all sizes in sizes
B = (64 * 0.16 + 1518 * 0.20 + 110 * 0.25 + mean(sizes) * rest) * 8; % Average Packet Size
w_mg1 = zeros (1,5);
tt_mg1 = zeros(1,5);
S1 = (64 * 8) /10e6; %C*1e6;
S2 = (110 * 8) / 10e6; %C*1e6;
S3 = (1518 * 8) / 10e6; %C*1e6;
S4 = ((mean(sizes)) * 8) / 10e6; %C*1e6;
S42 = 0;

for i = [65:109 111:1517]
    S42 = S42 + ((i*8)/10e6)^2;
end
S42 = S42 / length(sizes);
ES = 0.16 * S1 + 0.25 * S2 + 0.2 * S3 + (1-0.16-0.25-0.2) * S4;
ES2 = 0.16 * S1^2 + 0.25 * S2^2 + 0.2 * S3^2 + (1-0.16-0.25-0.2) * S42;
i=1;
for lam = lambda
    w_mg1(i) = ((lam * ES2) / (2 * (1 - lam * ES)) + ES) * 1000;
    tt_mg1(i) = lam * B * 1e-6 ;
    i=i+1;
end
```

```
51 -    data1 = [dataAPD;w_mg1];
52 -    data2 = [dataTT;tt_mg1];
53
54 -    figure(1)
55 -    h = bar(lambda,data1);
56 -    hold on
57 -    grid on
58 -    title("Average Packet Delay")
59 -    set(h, {'DisplayName'}, {'APD SIM','APD MG1'}')
60 -    legend('Location','northwest')
61 -    hold off
62
63 -    figure(2)
64 -    h = bar(lambda,data2);
65 -    hold on
66 -    grid on
67 -    title("Total Throughput")
68 -    set(h, {'DisplayName'}, {'TT SIM','TT MG1'}')
69 -    legend('Location','northwest')
70 -    hold off
```

Figura 33: Código Matlab Task 3.h

Análise de Código:

Nas primeiras 24 linhas podemos observar um código semelhante ao código da alínea 3g, não contendo no entanto o cálculo das margens de erro. De seguida, das linhas 27 a 49 encontra-se o cálculo dos valores teóricos para as condições descritas no enunciado, para uma fila do tipo M/G/1. Nas restantes linhas é feito o plot dos valores obtidos.

Resultados e Análise:

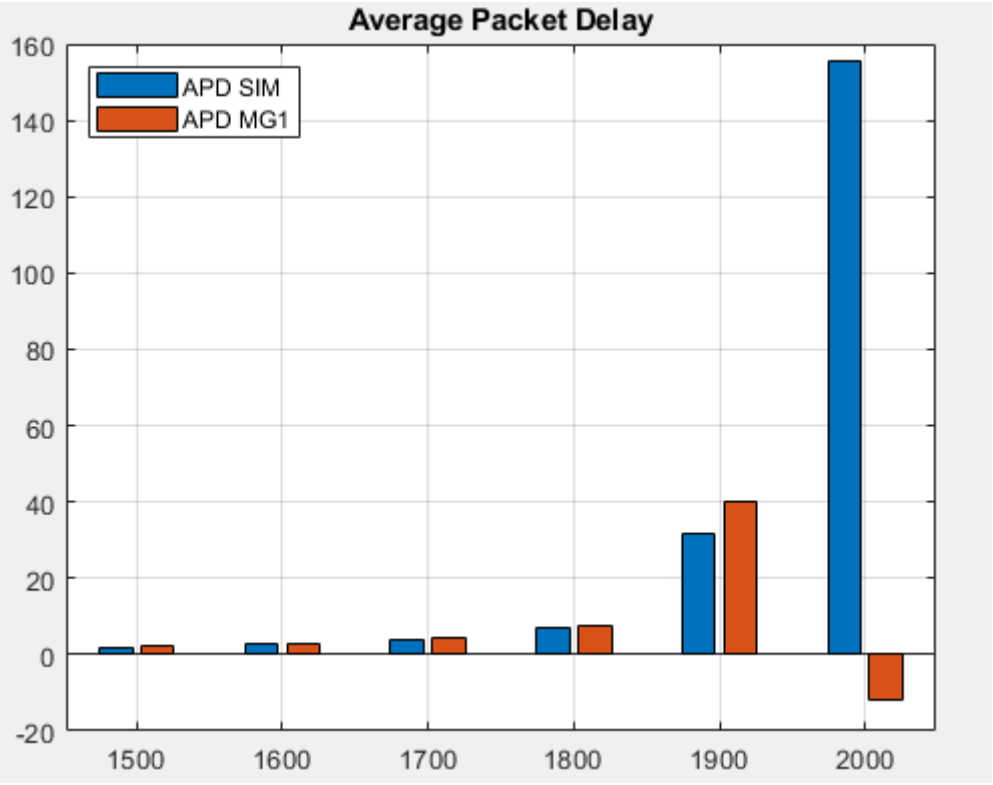

Figura 34: Average Packet Delay em função do número de pacotes por segundo

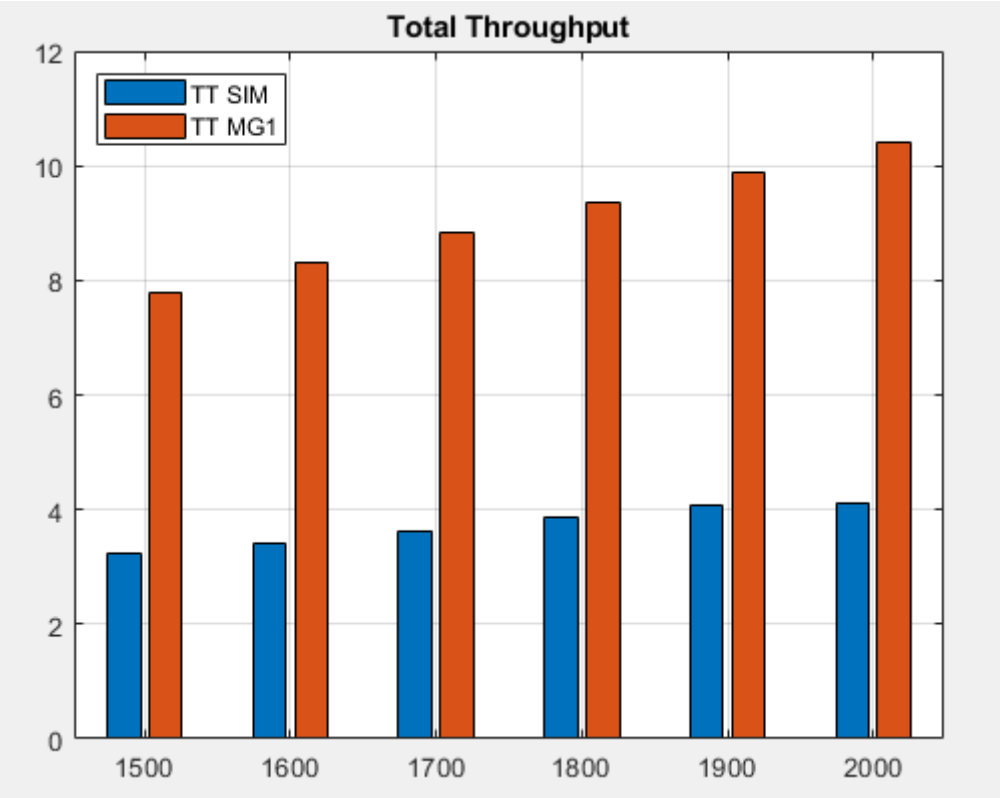

Figura 35: Total Throughput em função do número de pacotes por segundo

No gráfico obtido de tempo médio de atraso é possível verificar que os valores da simulação e os valores teóricos são sempre similares, à exceção de quando lambda é 2000, e neste caso o valor do tempo fica negativo. Isto ocorre uma vez que *lambda*ES* é maior que 1, resultando num valor negativo.

No que toca ao total throughput, podemos observar que os valores da simulação são sempre mais ou menos metade dos valores teóricos mantendo na mesma uma tendência de crescimento à medida que o número de pacotes por segundo aumenta..

Task 4:

a)

```
1   % Task 4a
2
3   P = 100000;
4   alfa = 0.1;
5   lambda = 1800;
6   C = 10;
7   f = 1000000;
8   b = 0;
9   N = 1:1:10;
10
11  resultsPL = zeros(1,10);
12  resultsMPD = zeros(1,10);
13  resultsAPD = zeros(1,10);
14  resultsTT = zeros(1,10);
15
16  for it = N
17      [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator3(lambda, C, f, P, b);
18  end
19
20  termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/10);
21  termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/10);
22  termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/10);
23  termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/10);
24  PL = mean(resultsPL);
25  APD = mean(resultsAPD);
26  MPD = mean(resultsMPD);
27  TT = mean(resultsTT);
28
29  fprintf('PL = %0.2e +- %0.2e \n', PL, termPL);
30  fprintf('APD= %0.2e +- %0.2e \n', APD,termAPD);
31  fprintf('MPD= %0.2e +- %0.2e \n', MPD, termMPD);
32  fprintf('TT = %0.2e +- %0.2e \n', TT, termTT);
```

Figura 36: Código MatLab Task 4a

Análise de Código:

As primeiras 14 linhas de código inicializam todas as variáveis necessárias à simulação e de acordo com o cenário descrito no enunciado. No ciclo *for* presente nas linhas 16 a 18 são recolhidos os valores da simulação, utilizando o *simulator3.* Por fim, são calculados os valores médios para cada parâmetro de saída, bem como as suas margens de erro, e são impressos os resultados.

Resultados e Análise:

PL = 5.17e-02 +- 5.45e-02
APD= 1.50e+02 +- 2.88e+01
MPD= 6.67e+02 +- 7.87e+01
TT = 9.38e+00 +- 7.82e-02

b)

```
% Task 4b

P = 100000;
alfa = 0.1;
lambda = 1800;
C = 10;
f = 10000;
b = 10e-5;
N = 1:1:10;

resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);

for it = N
    [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator3(lambda, C, f, P, b);
end

termPL = norminv(1-alfa/2)*sqrt(var(resultsPL)/10);
termAPD = norminv(1-alfa/2)*sqrt(var(resultsAPD)/10);
termMPD = norminv(1-alfa/2)*sqrt(var(resultsMPD)/10);
termTT = norminv(1-alfa/2)*sqrt(var(resultsTT)/10);
PL = mean(resultsPL);
APD = mean(resultsAPD);
MPD = mean(resultsMPD);
TT = mean(resultsTT);


fprintf('PL sim3 = %0.2e +- %0.2e \n', PL, termPL);
fprintf('APD sim = %0.2e +- %0.2e \n', APD,termAPD);
fprintf('MPD sim3 = %0.2e +- %0.2e \n', MPD, termMPD);
fprintf('TT sim2 = %0.2e +- %0.2e \n', TT, termTT);
```

Figura 37: Código MatLab Task 4b

Análise de Código:

Código similar ao código da alínea anterior, sendo a única diferença no valor do Bit Error Rate (*b*).

Resultados e Análise:

PL sim3 = 3.65e+01 +- 1.11e-01
APD sim = 4.13e+00 +- 4.46e-02
MPD sim3 = 9.16e+00 +- 1.34e-02
TT sim2 = 3.48e+00 +- 1.88e-02

c)

```
% Task 4c

P = 100000;
alfa = 0.1;
lambda = 1500:100:2000;
C = 10;
f = 10000000;
b = 0;
N = 1:1:10;

resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);

resultsPL2 = zeros(1,10);
resultsMPD2 = zeros(1,10);
resultsAPD2 = zeros(1,10);
resultsTT2 = zeros(1,10);

PL = zeros(1,5);
MPD = zeros(1,5);
APD = zeros(1,5);
TT = zeros(1,5);

PL2 = zeros(1,5);
MPD2 = zeros(1,5);
APD2 = zeros(1,5);
TT2 = zeros(1,5);

i= 1;
for lam = lambda

    for it = N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator3(lam, C, f, P, b);
        [resultsPL2(it),resultsAPD2(it),resultsMPD2(it),resultsTT2(it)] = simulator2(lam, C, f, P, b);
    end

    PL(i) = mean(resultsPL);
    APD(i) = mean(resultsAPD);
    MPD(i) = mean(resultsMPD);
    TT(i) = mean(resultsTT);

    PL2(i) = mean(resultsPL2);
    APD2(i) = mean(resultsAPD2);
    MPD2(i) = mean(resultsMPD2);
    TT2(i) = mean(resultsTT2);
    i= i+1;
end
```

```
51 -    dataPL = [PL;PL2];
52 -    figure(1)
53 -    h = bar(lambda,dataPL);
54 -    hold on
55 -    grid on
56 -    title("Packet Loss")
57 -    set(h, {'DisplayName'}, {'PL SIM3','PL SIM2'}')
58 -    legend('Location','northwest')
59 -    hold off
60
61 -    dataAPD = [APD;APD2];
62 -    figure(2)
63 -    h = bar(lambda,dataAPD);
64 -    hold on
65 -    grid on
66 -    title("Average Packet Delay")
67 -    set(h, {'DisplayName'}, {'APD SIM3','APD SIM2'}')
68 -    legend('Location','northwest')
69 -    hold off
70
71 -    dataMPD = [MPD;MPD2];
72 -    figure(3)
73 -    h = bar(lambda,dataMPD);
74 -    hold on
75 -    grid on
76 -    title("Maximum Packet Delay")
77 -    set(h, {'DisplayName'}, {'MPD SIM3','MPD SIM2'}')
78 -    legend('Location','northwest')
79 -    hold off
80
81 -    dataTT = [TT;TT2];
82 -    figure(4)
83 -    h = bar(lambda,dataTT);
84 -    hold on
85 -    grid on
86 -    title("Total Throughput")
87 -    set(h, {'DisplayName'}, {'TT SIM3','TT SIM2'}')
88 -    legend('Location','northwest')
89 -    hold off
```

Figura 38: Código MatLab Task 4c

Análise de Código:

Nas primeiras 31 linhas de código são apenas criadas e inicializadas as variáveis de acordo com o cenário descrito no enunciado. Das linhas 32 a 49 são executados os simuladores 2 e 3 e armazenados os resultados para cada um. Por fim, das linhas 51 a 89 são gerados os gráficos correspondentes.

Resultados e Análise:

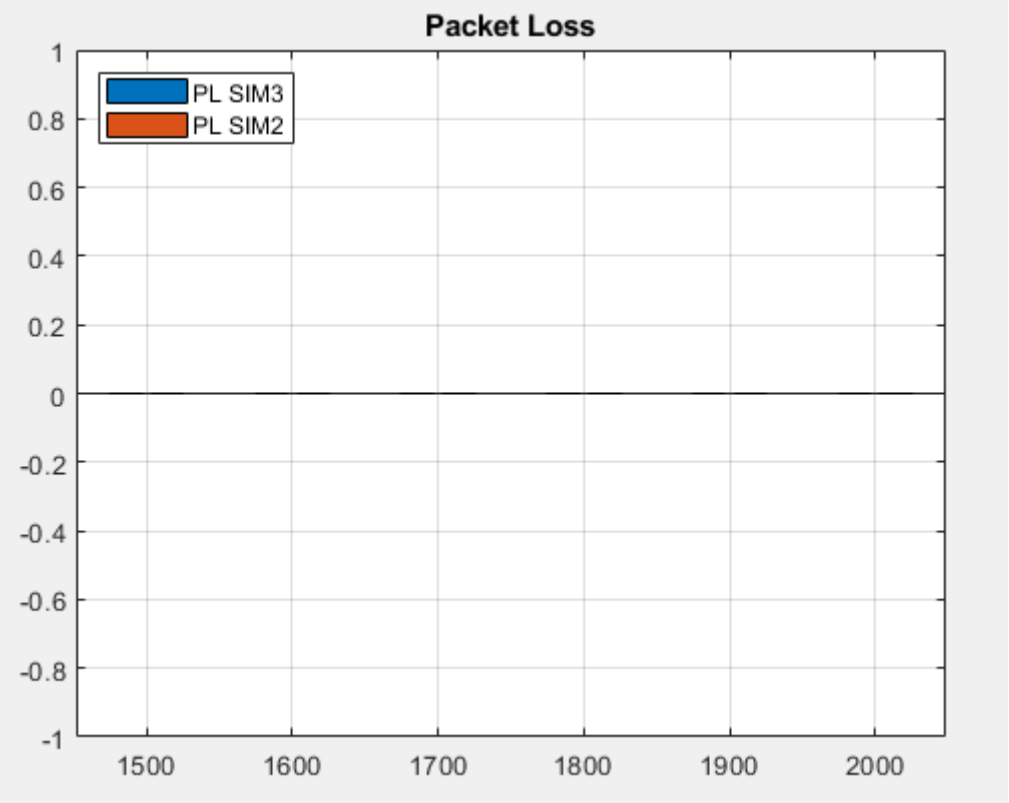

Figura 39: Packet Loss em função do número de pacotes por segundo

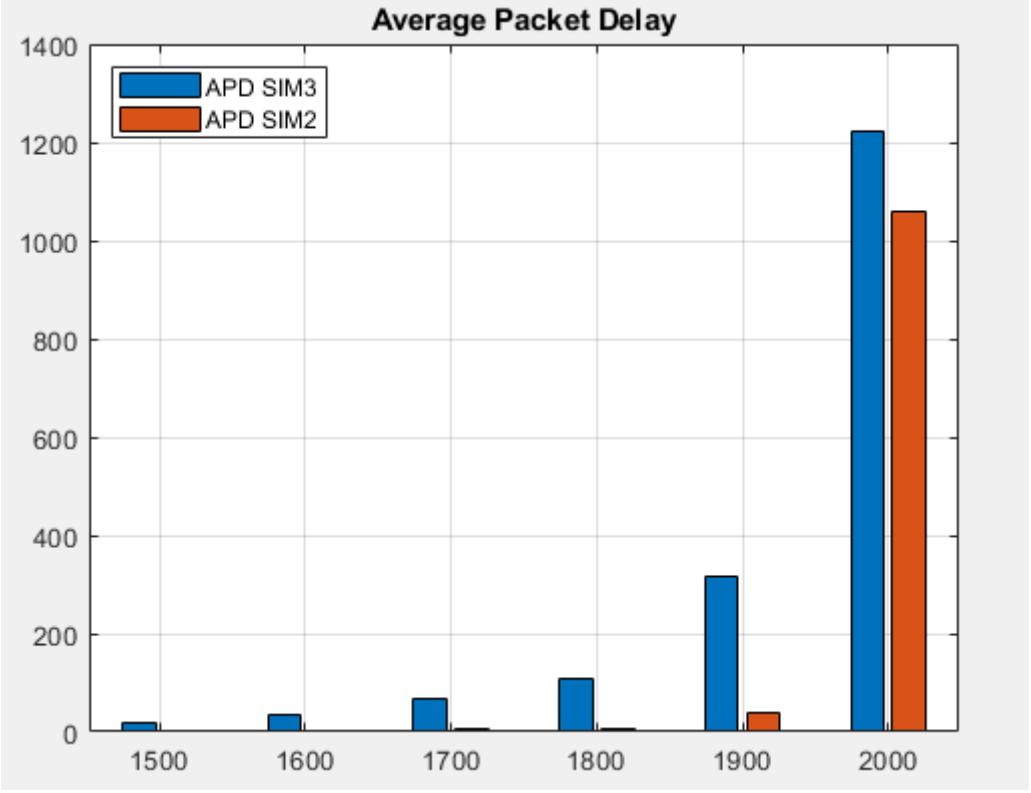

Figura 40: Average Packet Delay em função do número de pacotes por segundo

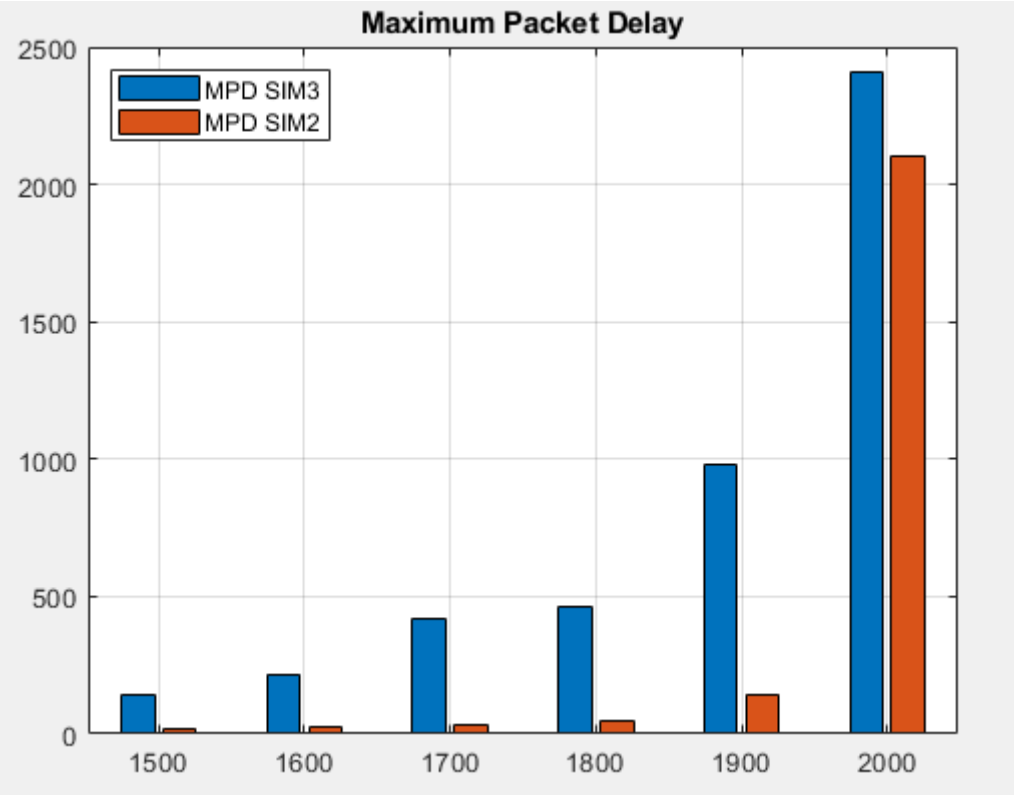

Figura 41: Maximum Packet Delay em função do número de pacotes por segundo

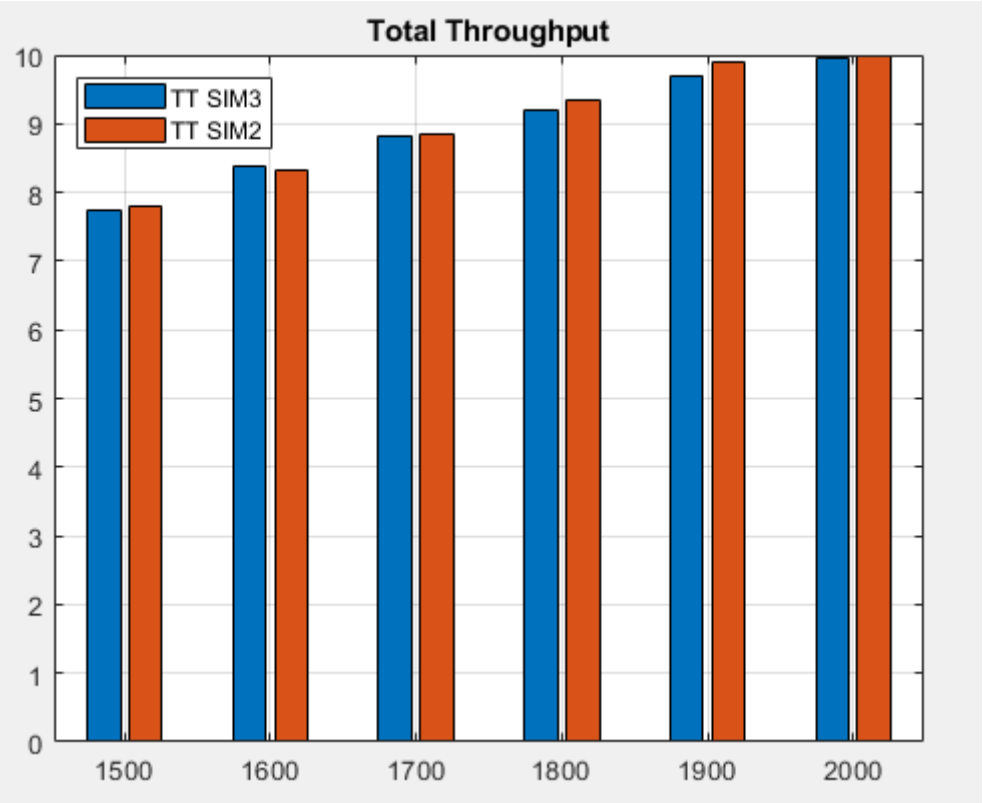

Figura 42: Total Throughput em função do número de pacotes por segundo

O Packet Loss é 0 pois o bit error rate é 0 e a fila de espera possui espaço suficiente para todos os pacotes. Em ambos os simuladores os valores de MPD e APD seguem o mesmo

comportamento mas para o simulador 3 os resultados são significativamente maiores, uma vez que para o simulador 3 o tempo de processamento de um pacote depende agora do estado da cadeia de Markov em que o simulador se encontra, logo os pacotes ficam mais tempo em espera quando chegam ao serviço. É possível verificar nos gráficos que a performance do simulador 2 foi superior à do simulador 3, principalmente a nível de Average Packet Delay e Maximum Packet Delay obtendo tempos de espera menores.

d)

```
% Task 4d

P = 100000;
alfa = 0.1;
lambda = 1800;
C = 10;
f_sizes = 2500:2500:20000;
b = 0;
N = 1:1:10;

resultsPL = zeros(1,10);
resultsMPD = zeros(1,10);
resultsAPD = zeros(1,10);
resultsTT = zeros(1,10);

resultsPL2 = zeros(1,10);
resultsMPD2 = zeros(1,10);
resultsAPD2 = zeros(1,10);
resultsTT2 = zeros(1,10);

PL = zeros(1,5);
MPD = zeros(1,5);
APD = zeros(1,5);
TT = zeros(1,5);

PL2 = zeros(1,5);
MPD2 = zeros(1,5);
APD2 = zeros(1,5);
TT2 = zeros(1,5);

i= 1;
for f = f_sizes

    for it = N
        [resultsPL(it),resultsAPD(it),resultsMPD(it),resultsTT(it)] = simulator3(lambda, C, f, P, b);
        [resultsPL2(it),resultsAPD2(it),resultsMPD2(it),resultsTT2(it)] = simulator2(lambda, C, f, P, b);
    end

    PL(i) = mean(resultsPL);
    APD(i) = mean(resultsAPD);
    MPD(i) = mean(resultsMPD);
    TT(i) = mean(resultsTT);

    PL2(i) = mean(resultsPL2);
    APD2(i) = mean(resultsAPD2);
    MPD2(i) = mean(resultsMPD2);
    TT2(i) = mean(resultsTT2);
    i= i+1;
end
```

```
51 -    dataPL = [PL;PL2];
52 -    figure(1)
53 -    h = bar(f_sizes,dataPL);
54 -    hold on
55 -    grid on
56 -    title("Packet Loss")
57 -    set(h, {'DisplayName'}, {'PL SIM3','PL SIM2'}')
58 -    legend('Location','northwest')
59 -    hold off
60
61 -    dataAPD = [APD;APD2];
62 -    figure(2)
63 -    h = bar(f_sizes,dataAPD);
64 -    hold on
65 -    grid on
66 -    title("Average Packet Delay")
67 -    set(h, {'DisplayName'}, {'APD SIM3','APD SIM2'}')
68 -    legend('Location','northwest')
69 -    hold off
70
71 -    dataMPD = [MPD;MPD2];
72 -    figure(3)
73 -    h = bar(f_sizes,dataMPD);
74 -    hold on
75 -    grid on
76 -    title("Maximum Packet Delay")
77 -    set(h, {'DisplayName'}, {'MPD SIM3','MPD SIM2'}')
78 -    legend('Location','northwest')
79 -    hold off
80
81 -    dataTT = [TT;TT2];
82 -    figure(4)
83 -    h = bar(f_sizes,dataTT);
84 -    hold on
85 -    grid on
86 -    title("Total Throughput")
87 -    set(h, {'DisplayName'}, {'TT SIM3','TT SIM2'}')
88 -    legend('Location','northwest')
89 -    hold off
```

Figura 43: Código MatLab Task 4d

Análise de Código:

Código similar ao da alínea c, onde em vez de correr os simuladores para vários valores de número de pacotes por segundo, são corridos para diversos valores do tamanho da fila.

Resultados e Análise:

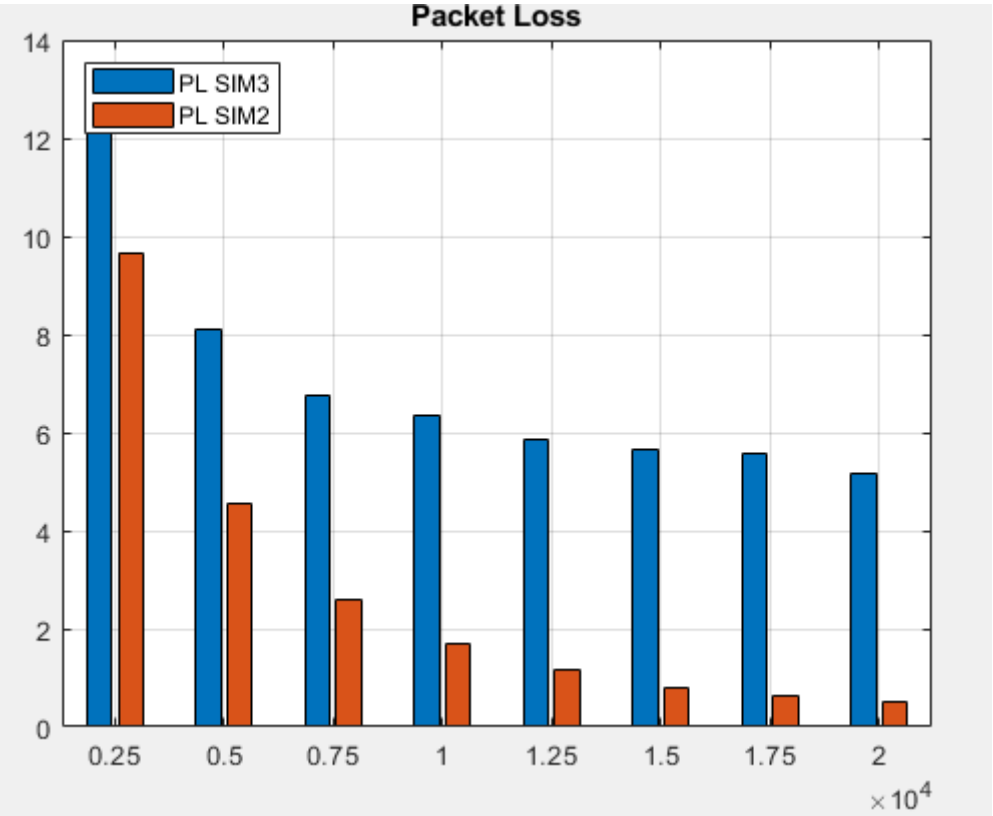

Figura 44: Packet Loss em função do tamanho da fila de espera

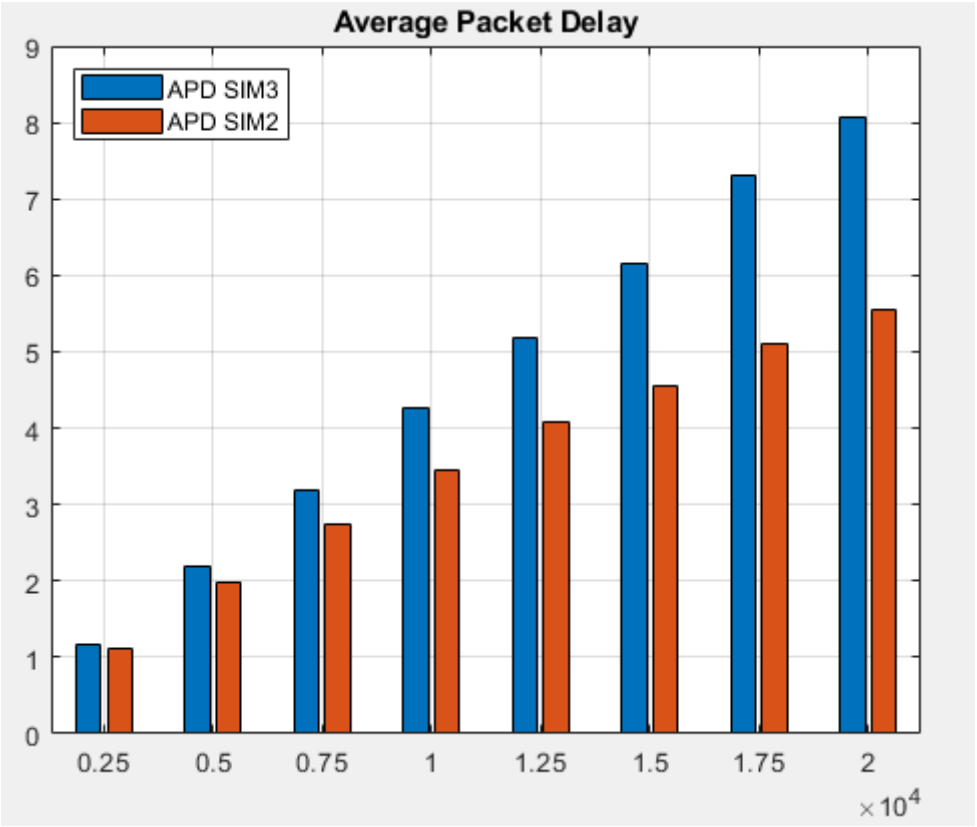

Figura 45: Packet Loss em função do tamanho da fila de espera

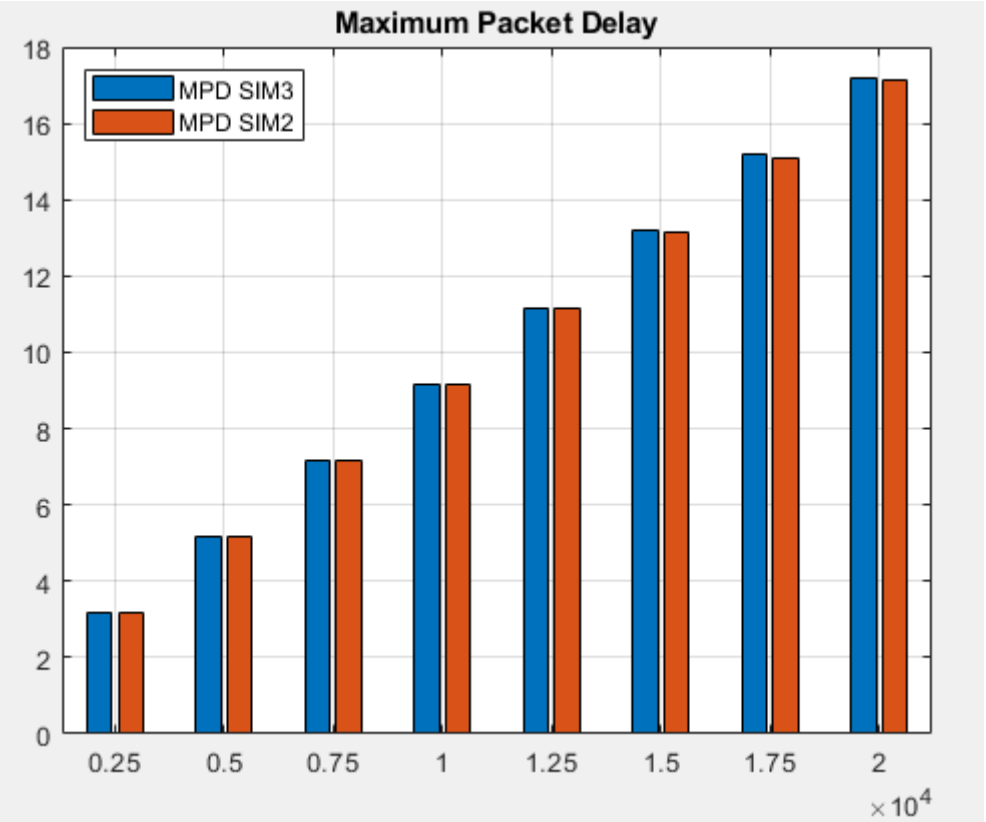

Figura 46: Maximum Packet Delay em função do tamanho da fila de espera

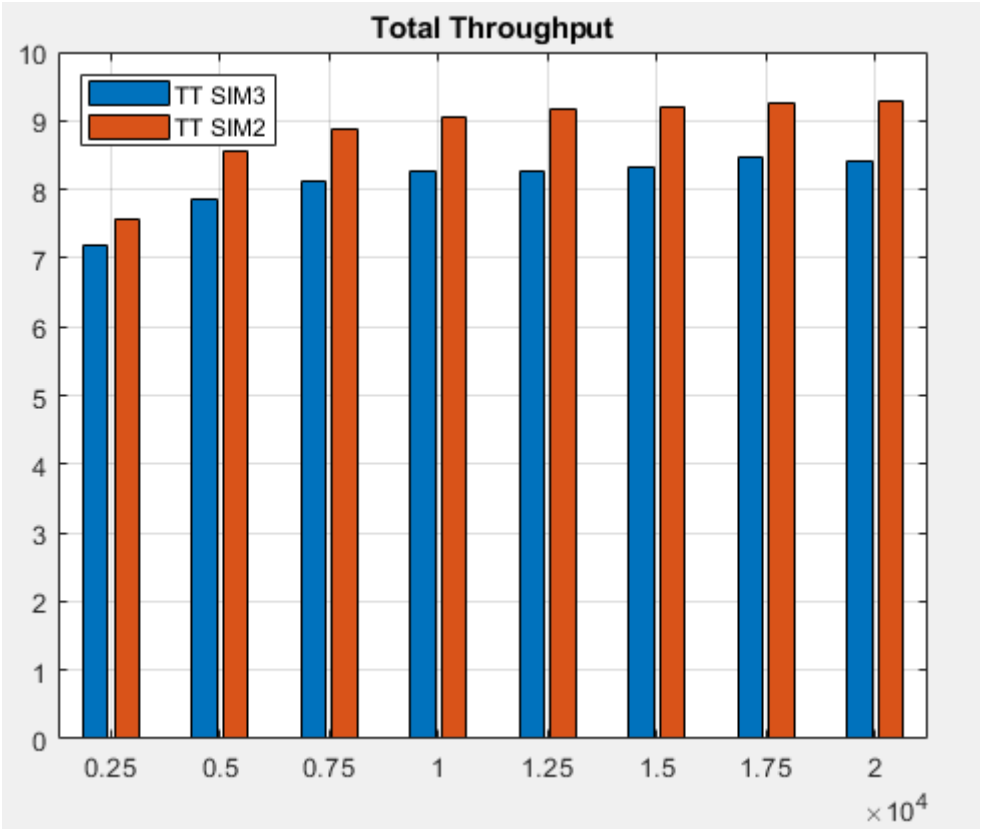

Figura 47: Total Throughput em função do tamanho da fila de espera

Como se pode observar a performance do simulador 2 foi superior à do simulador 3, principalmente a nível de Total Throughput, Packet Loss e Average Packet Delay, onde a diferença entre os resultados PL e APD é mais evidente e significativa. Algo que é explicado pelo facto de uma vez que o para o simulador 3 o tempo de processamento dos pacotes é dependente do estado da cadeia de Markov representativa em que se encontra, logo quanto mais tempo for necessário para processar pedidos, mais pedidos ficarão na fila de espera e por consequência mais pacotes serão perdidos, assim como mais tempo os pacotes na fila terão em média de esperar.

NOTA: Não será apresentado o código para alíneas 4.e) e 4.f) pois a única alteração entre o código de 4.c) - 4.e) e 4.d) - 4.f) encontra-se na linha 8 onde é alterado o valor da variável *b* para 10^-5.

e)

Resultados e Análise:

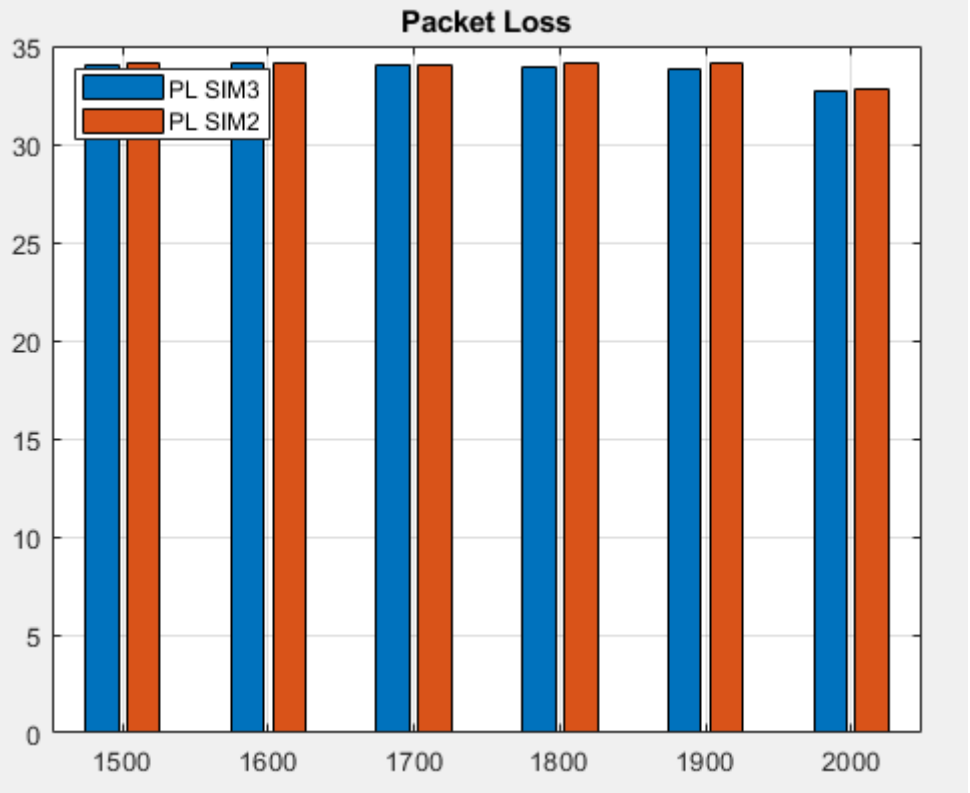

Figura 48: Packet Loss em função do número de pacotes por segundo

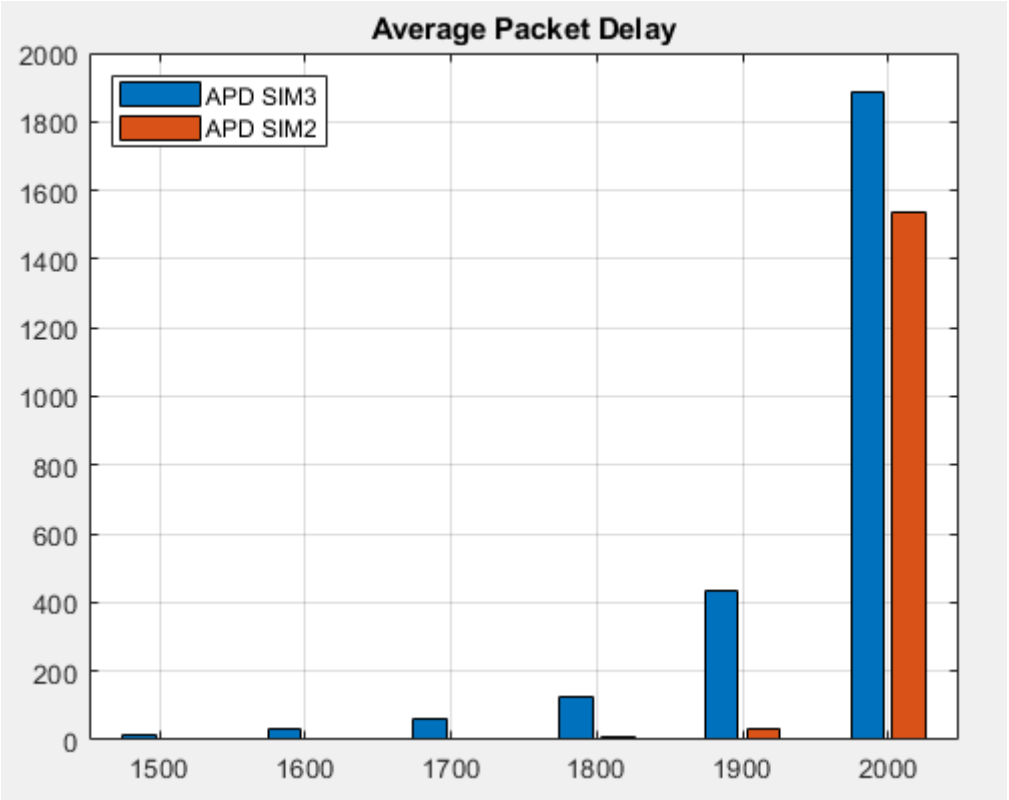

Figura 49: Average Packet Delay em função do número de pacotes por segundo

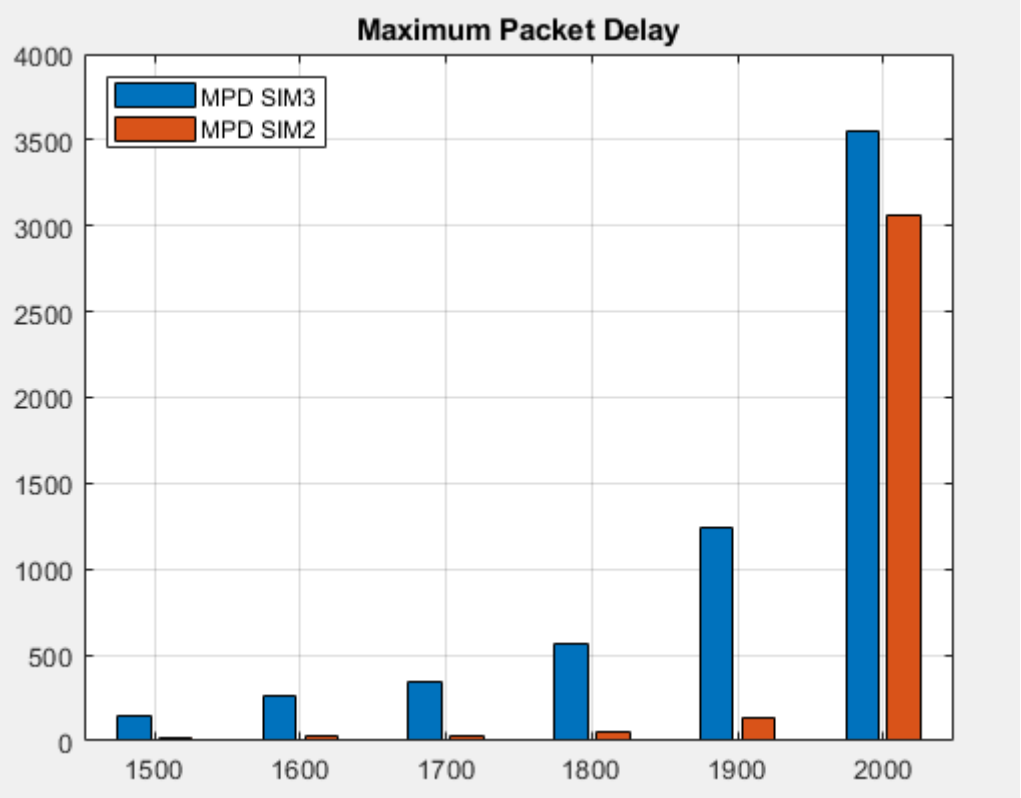

Figura 50: Maximum Packet Delay em função do número de pacotes por segundo

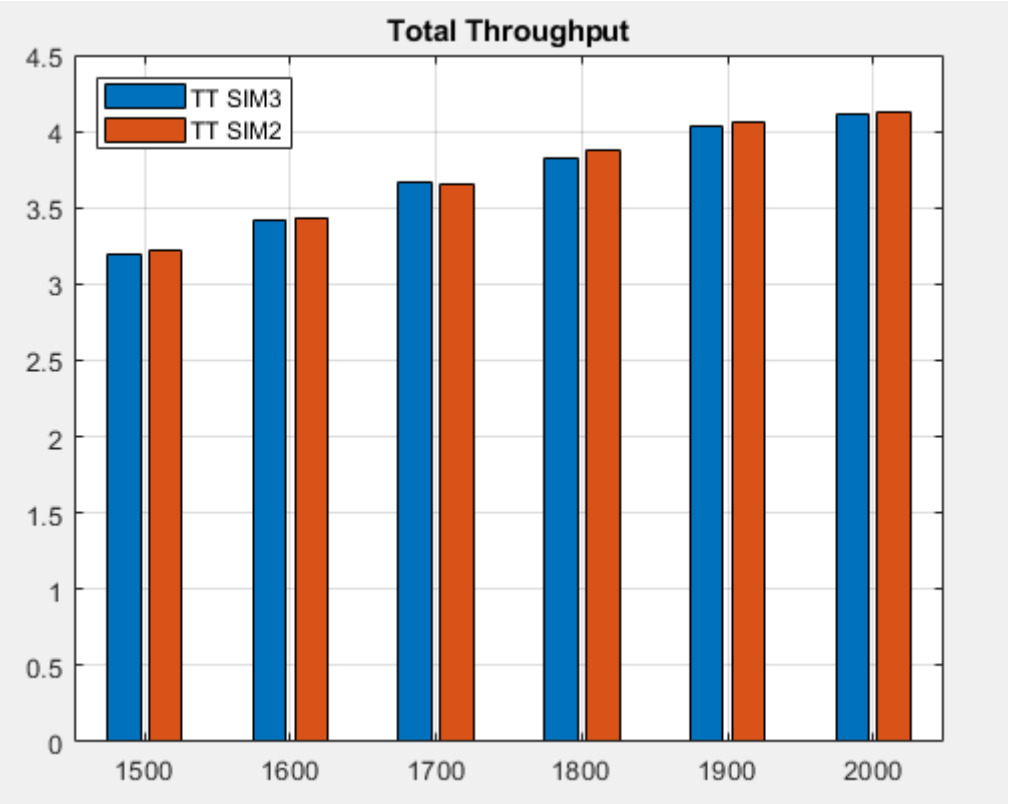

Figura 51: Total Throughput em função do número de pacotes por segundo

Neste caso, o Packet Loss já não foi nulo como na alínea 4.c devido ao facto de o bit error rate já não ser 0, o que levou a resultados semelhantes de Packet Loss para ambos os simuladores. Os resultados de Total Throughput foram menores que os obtidos na alínea 4.c uma vez que foram perdidos alguns pacotes, e os resultados de Average Packet Delay e Maximum Packet Delay foram superiores comparativamente aos da alínea 4.c.

f)

Resultados e Análise:

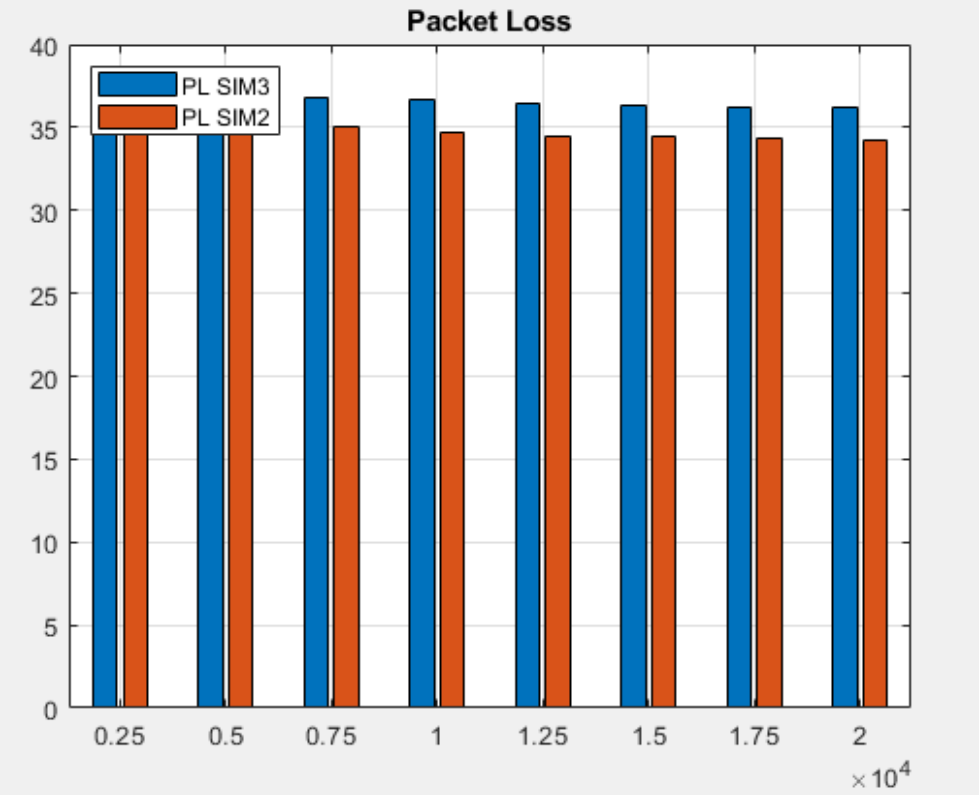

Figura 52: Packet Loss em função do número de tamanho da fila de espera

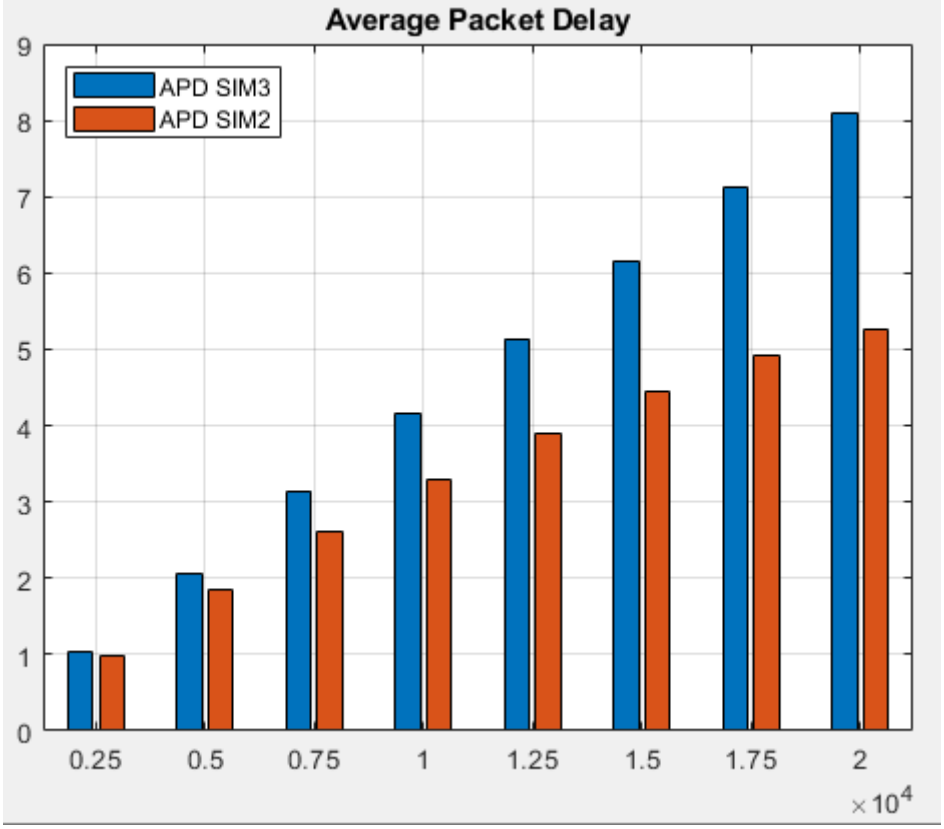

Figura 53: Average Packet Delay em função do número de tamanho da fila de espera

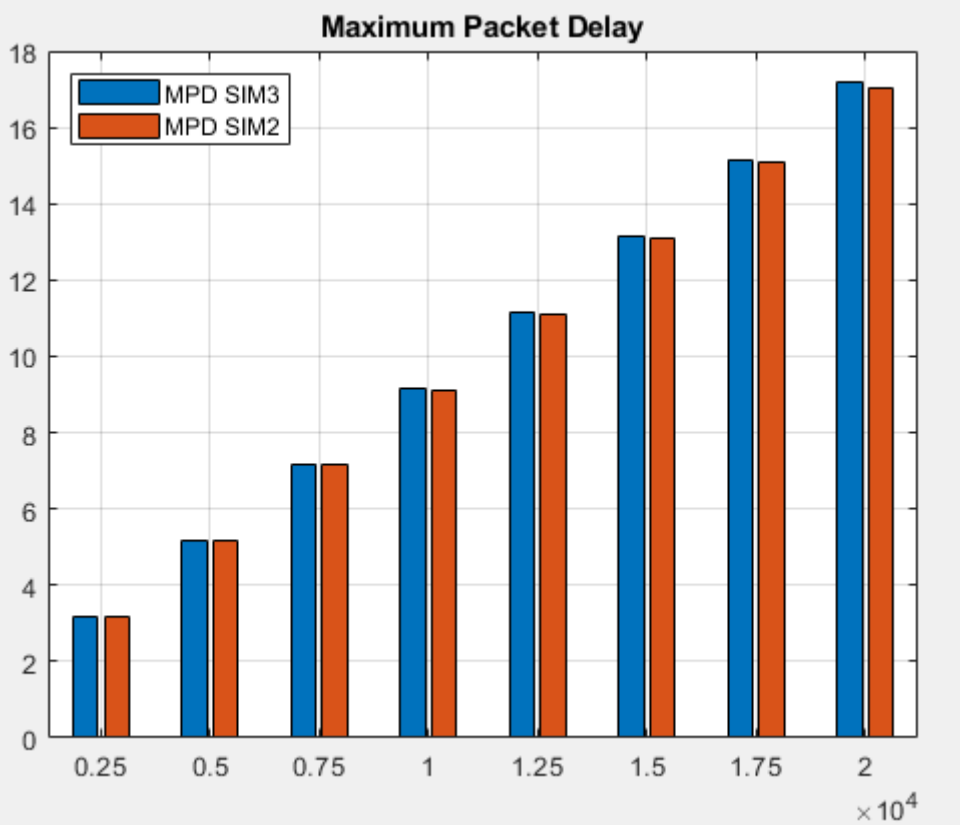

Figura 54: Maximum Packet Delay em função do número de tamanho da fila de espera

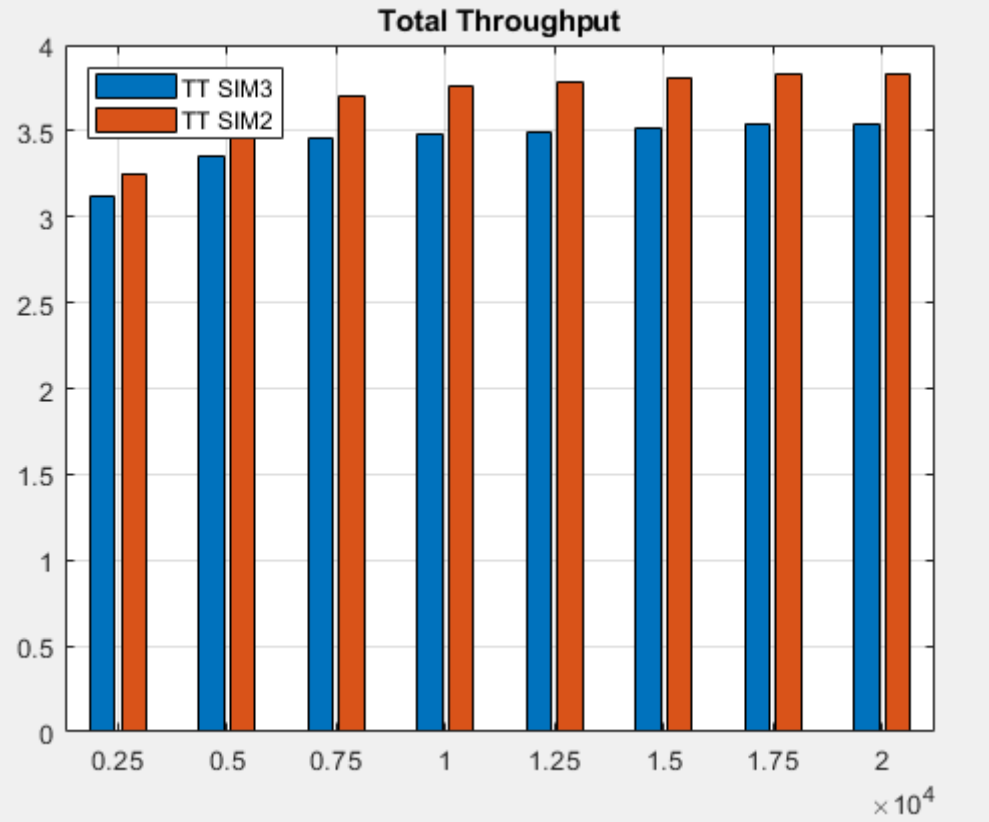

Figura 55: Total Throughput em função do número de tamanho da fila de espera

Podemos observar que o Packet Loss em ambos os simuladores foi similar e com valores muito superiores comparativamente aos resultados obtidos na alínea 4.d, devido ao facto de

o bit error rate já não ser 0. Os resultados de Total Throughput foram menores que os obtidos na alínea 4.d uma vez que foram perdidos alguns pacotes, e os resultados de Average Packet Delay e Maximum Packet Delay mantiveram-se relativamente iguais aos previamente obtidos na alínea 4.d com o simulador 2 a conseguir melhores resultados e com isso uma melhor performance que o simulador 3.

Apêndice A:

Simulator2:

```
61 -        case DEPARTURE                    % If first event is a DEPARTURE
62 -
63 -            prob_err = (1-b) ^ (8 * PacketSize);
64 -            if(rand(1)>prob_err)
65 -                PACKETS_WITH_ERRORS = PACKETS_WITH_ERRORS + 1;
66 -                LOSTPACKETS = LOSTPACKETS + 1;
67 -            else
68 -                TRANSMITTEDBYTES= TRANSMITTEDBYTES + PacketSize;
69 -                DELAYS= DELAYS + (Clock - ArrivalInstant);
70 -                if Clock - ArrivalInstant > MAXDELAY
71 -                    MAXDELAY= Clock - ArrivalInstant;
72 -                end
73 -                TRANSMITTEDPACKETS= TRANSMITTEDPACKETS + 1;
74 -            end
75 -            if QUEUEOCCUPATION > 0
76 -                EventList = [EventList; DEPARTURE, Clock + 8*QUEUE(1,1)/(C*10^6), QUEUE(1,1), QUEUE(1,2)];
77 -                QUEUEOCCUPATION= QUEUEOCCUPATION - QUEUE(1,1);
78 -                QUEUE(1,:)= [];
79 -            else
80 -                STATE= 0;
81 -            end
```

Figura 56: Alteração relativa ao Simulador 1

Para o simulador 2, foi necessário adicionar uma condição ao simulador 1, que verifica se um pacote chegou ou não com erro, tendo em conta a probabilidade de tal acontecer. Para isso é calculada a probabilidade de o pacote ter chegado sem nenhum bit errado (linha 63) e de seguida, caso seja gerado um valor aleatório superior a essa probabilidade, esse pacote vai ser considerado como perdido caso contrário segue a simulação tal como no simulador 1.

Apêndice B:

Simulator3:

```
14      %Events:
15 -    ARRIVAL= 0;        % Arrival of a packet
16 -    DEPARTURE= 1;      % Departure of a packet
17 -    TRANSITION = 2;
```

Figura 57: Alteração relativa ao Simulador 2

Adição de um novo evento, evento TRANSITION que significa a mudança de estado de um pacote que chega à cadeia de Markov.

```
19      %State variables:
20 -    STATE = 0;          % 0 - connection free; 1 - connection bysy
21 -    QUEUEOCCUPATION= 0; % Occupation of the queue (in Bytes)
22 -    QUEUE= [];          % Size and arriving time instant of each packet in the queue
23 -    FLOWSTATE = 0;
```

Figura 58: Alteração relativa ao Simulador 2

Adição de uma variável de estado FLOWSTATE, que simboliza o estado atual de um pacote, numa cadeia de Markov, em que os valores possíveis são 1, 2 e 3.

```
39 -    F1 = 1 / (1 + 10/5 + 10/5*5/10);
40 -    F2 = (10/5)  / (1 + 10/5 + 10/5*5/10);
41 -    F3 = (10/5*5/10)  / (1 + 10/5 + 10/5*5/10);
42 -    T_perm = 1/10;
43
44 -    aux_rand= rand();
45 -    if aux_rand <= F1
46 -        rate=0.5*lambda;
47 -        FLOWSTATE = 1;
48 -    elseif aux_rand <= F1+F2
49 -        rate = lambda;
50 -        FLOWSTATE = 2;
51 -    else
52 -        rate = 1.5*lambda;
53 -        FLOWSTATE = 3;
54 -    end
55 -    EventList = [ARRIVAL, Clock + exprnd(1/rate), GeneratePacketSize(), 0];
56 -    EventList = [EventList; TRANSITION, Clock + exprnd(T_perm), 0, 0];
```

Figura 59: Alteração relativa ao Simulador 2

Nas linhas 39 a 41 são calculadas as probabilidades de cada estado, bem como o tempo de permanência em cada um na linha 42, que é igual para todos. De seguida é inicializado o estado inicial da cadeia de Markov com base nas probabilidades calculadas anteriormente.

```
103 -           case TRANSITION
104 -               if FLOWSTATE ~= 2
105 -                   FLOWSTATE = 2;
106 -                   rate = lambda;
107 -               else
108 -                   if rand() <= 0.5
109 -                       FLOWSTATE = 1;
110 -                       rate = 0.5*lambda;
111 -                   else
112 -                       FLOWSTATE = 3;
113 -                       rate = 1.5*lambda;
114 -                   end
115
116 -               end
117 -               EventList = [EventList; TRANSITION, Clock + exprnd(T_perm), 0, 0];
```

Figura 60: Alteração relativa ao Simulador 2

Adição do comportamento para um evento do tipo TRANSITION (linhas 103-116) no qual, é alterado com base no estado atual, nas probabilidades e nas possibilidades de transição entre estados, o próximo estado da cadeia de Markov que representa o simulador 3. É também atualizado o *rate* do processamento de pacotes com base no estado para o qual transita. Por fim é agendada uma nova transição de estado do simulador (linha 117).