Desempenho e Dimensionamento de Redes - 2018/2019

Engenharia de Computadores e Telemática

4º Guião Prático - Relatório

PERFORMANCE ESTIMATION OF PACKET SWITCHED NETWORKS

P4G2
Diego Hernandez nº 77013
Ricardo Pousa nº 80328

Simulador 1

O simulador foi construído consoante o enunciado e o seu diagrama apresentado.
O código desenvolvido para o simulador 1 foi o seguinte:

```
function out = simulator1(par)

%Events
Arrival = 0;
Departure = 1;
Terminate = 2;

%State variables
State = 0; %Link is busy or not 0 = free 1 = busy
Clock = 0; %Current clock of the system
QueueOccupation = 0; %Total number of bytes on queue
Queue = []; % Structure with arrival time and size of packet per entry 2 columns

%Statistical counters
TotalPackets = 0; %Total packets arrived to the system
LostPackets = 0; %Total number of packets dropped by the system
Delays = 0; % Sum of all delays
TransmittedPackets = 0; %Total number of sent packets
TransmittedBytes = 0; %Total number of sent bytes

%Aux variables
Instant = 0; %Time of arrival the packet being transmitted
Syze = 0; %Size of the packet being transmitted

out.AvgPacketLoss = -1;
out.AvgPacketDelay = -1;
out.TransThroughput = -1 ;

B = 64* 0.19 + 1518*0.48 + (1517+65)/2 *(1-0.19-0.48);
a = (par.r/8)/B;
time_to_next_packet = exprnd(1/a);

Events = [time_to_next_packet Arrival;par.S Terminate];
```

```
while(Events(1,1) <= par.S)
   event = Events(1,2);
   Clock = Events(1,1);
   Events(1,:) = [];

   if(event == Arrival)  %

      PacketSize = packetsize();
      TotalPackets = TotalPackets +1;
      time_to_next_packet = exprnd(1/a) + Clock;
      if(State == 0)
         State = 1;
         Instant = Clock;
         Syze = PacketSize;
         time_to_next_departure = Instant + (PacketSize*8)/1e7;
         Events = [ Events; time_to_next_departure Departure ];

      elseif(QueueOccupation + PacketSize < par.f)
         Queue = [Queue; Clock PacketSize];
         QueueOccupation = QueueOccupation + PacketSize;
      else
         LostPackets = LostPackets +1;
      end
      Events = sortrows([ Events; time_to_next_packet Arrival ]);

   elseif(event == Departure)
      Delays = Delays + (Clock-Instant);
      TransmittedBytes =  TransmittedBytes + Syze;
      TransmittedPackets = TransmittedPackets +1;
      if (QueueOccupation > 0)
         Instant= Queue(1,1);
         Syze = Queue(1,2);
         time_to_next_departure = Clock + (Syze*8)/1e7;
         Events =  sortrows([ Events; time_to_next_departure Departure ]);
         Queue(1,:) = [];
         QueueOccupation= QueueOccupation - Syze;
      else
         State = 0;
      end
   else% (event == Terminate)
      out.AvgPacketLoss = 100 * (LostPackets/TotalPackets);
      out.AvgPacketDelay = Delays/TransmittedPackets * 1000;%ms
```

```
        out.TransThroughput = (8 * TransmittedBytes)/(par.S * 1e6) ;
      end
    end
%    fprintf("LostPackets %f, TotalPackets %f\n",LostPackets,TotalPackets);
%    fprintf("Average Packet Delay %f\n",Delays/TransmittedPackets);
%    fprintf("Throughput %f\n", TransmittedBytes * 1e6/ par.S);
%    Events

end

function PacketSize = packetsize()
r = rand();
if(r < 0.19)
   PacketSize = 64;
   return
end
if(r > 0.52)
   PacketSize = 1518;
   return
end
PacketSize = randi([65 1517]);
end
```

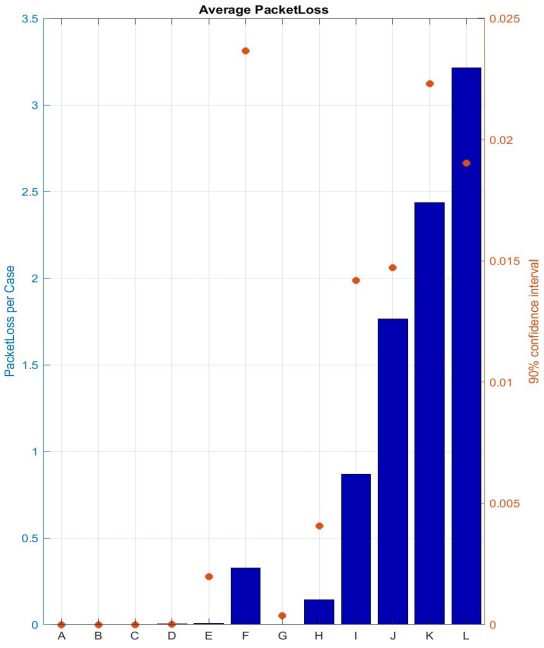
a)

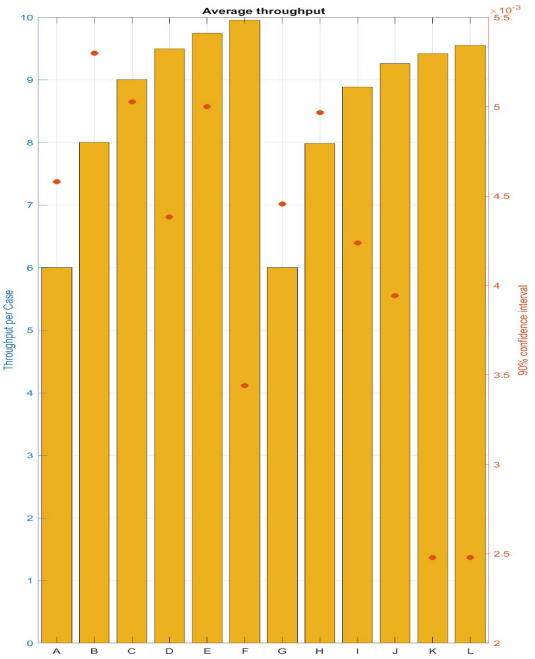
Os valores obtidos a partir da simulação foram os seguintes:

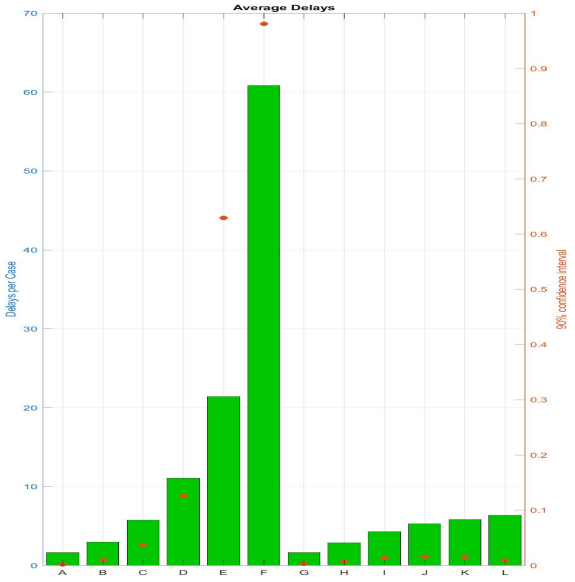
Case	r (Mbps)	f (Bytes)	Avg. Packet Loss (%)	90% Conf	Avg. Packet Delay (msec)	90% Conf	Transmitted Throughput (Mbps)	90% Conf
A	6	150000	0.0	0.000000	1.623723	0.002012	6.003503	0.004580
B	8	150000	0.0	0.000000	2.985540	0.009884	8.002095	0.005299
C	9	150000	0.0	0.000000	5.766522	0.036935	9.005573	0.005027
D	9.5	150000	0.000008	0.000014	11.062033	0.126032	9.499063	0.004383
E	9.75	150000	0.006458	0.001976	21.432391	0.629102	9.748045	0.005000
F	10.0	150000	0.326935	0.023662	60.819711	0.980454	9.956237	0.003440
G	6	15000	0.001162	0.000371	1.623549	0.003302	6.003646	0.004457
H	8	15000	0.144837	0.004065	2.873813	0.007381	7.983578	0.004967
I	9	15000	0.868799	0.014187	4.308028	0.013788	8.889465	0.004239
J	9.5	15000	1.764129	0.014714	5.288708	0.015927	9.262063	0.003944

K	9.75	15000	2.435815	0.022297	5.839905	0.015624	9.422661	0.002479
L	10	15000	3.215775	0.019025	6.366030	0.009140	9.553501	0.002944

b)







Tendo em conta os resultados obtidos na alínea a) tiramos as seguintes conclusões:

- Relativamente ao tamanho da fila de pacotes à espera de serem transmitidos, quanto maior, menor será a percentagem de pacotes perdidos, pois maior são as chances de haver espaço na fila para armazenar o pacote até chegar o seu turno de transmissão, não sendo estes perdido. Verifica-se na prática que para casos em que haja o mesmo throughput mas com

maior tamanho de fila de espera, estes sofrem pouca ou nenhuma perda de pacotes em comparação com casos que tem uma fila de espera menor.

- Para o mesmo tamanho da fila de espera de transmissão de pacotes e para a mesma capacidade de link para a transferência de pacotes, verifica-se que quanto maior for a taxa de chegada de pacotes, maior será a probabilidade de perda de pacotes para uma fila de espera finita. Isto deve-se ao facto de a capacidade de transferência do link não ser suficiente para a transmissão de todos os pacotes de modo a transmitir todos os pacotes recebidos e evitar que a fila seja totalmente ocupada consoante a taxa de chegada dos pacotes. Não sendo suficiente esta, surge a perda de pacotes quanto maior for esta taxa, devido a taxa de chegada de pacotes e a consequente armazenamento dos pacotes na fila ser grande.
- Relativamente ao atraso médio de pacotes, para o mesmo tamanho de fila de pacotes à espera de serem transmitidos, verifica-se que quanto maior for a taxa de transmissão de pacotes maior será o atraso médio dos pacotes. Isto deve-se pelo facto das filas conseguirem num espaço de tempo curto depositar um número grande de pacotes a espera de serem transmitidos, quando a taxa de chegada de pacotes for grande. Portanto quanto mais ocupada a fila de espera estiver, maior será o tempo de atraso do pacote a chegar ao sistema até ser o seu turno de ser transmitido.
- Já foi mencionado que em casos com taxas de chegada iguais, quanto menor for a fila de espera maiores são as chances de pacotes serem perdidos por causa de não haver espaço suficiente na fila. No entanto verifica-se que o atraso médio de pacotes para filas de tamanho pequeno e com grande valor na taxa de transferência de pacotes (consequentemente maior percentagem de perda de pacotes), o atraso médio de pacotes é menor em contraste para os casos em que o tamanho da fila é grande. Quando a fila de espera é grande, maior será a capacidade do link armazenar pacotes a ser transmitidos, conseguindo tratar da transmissão de maior parte do número de pacotes a serem transmitidos, se não todos, havendo poucas ou nenhuma perda de pacotes, dependendo da taxa de chegada de pacotes e do tamanho da fila. No entanto os atrasos médios aumentam pois embora as perdas sejam poucas ou nulas, quanto mais pacotes estiverem à espera na fila para serem transmitidos, maior será o tempo de atraso ou espera para um novo pacote ser transmitido. No entanto quanto menor for a fila, maior será a perda de pacotes sendo limitado o armazenamento e tempo de espera para a transmissão de pacotes no sistema. É por esta razão que o atraso médio dos pacotes transmitidos nos casos D, E e F têm um maior tempo de atraso em comparação com os casos J, K e L, respectivamente. Notamos também que nos casos K e L o atraso médio de pacotes é aproximado, devido ao limite imposto pela fila, sendo próximo do limite máximo de tempo de atraso que um novo pacote a ser transmitido no link poderá ter, tendo em conta a taxa de chegada dos pacotes, capacidade do link e tamanho da fila.
- Relativamente aos valores obtidos no Transmitted Throughput os valores aumentam à medida que a taxa de chegada aumenta. Comparando os casos cuja taxa de chegada é igual mas o tamanho da fila não o é, verificamos que o tamanho da fila não irá afetar os valores de transmitted throughput, sendo unicamente dependente da capacidade do link, tamanhos dos pacotes e capacidade do link

c)
Na seguinte tabela, foi determinado para os casos em que não se registou perdas de pacotes durante a simulação, isto é os casos {A, B, C, D}, os valores teóricos do atraso médio dos pacotes para os seguintes modelos: modelo de fila do tipo M/M/1, onde o processo de atendimento é do tipo Markoviano e o tempo de atendimento de um servidor é exponencialmente distribuído com média $1/\mu$; e para o modelo de fila M/G/1, onde processo de atendimento é do tipo determinístico e o tempo de atendimento S do servidor tem uma distribuição genérica e independente das chegadas dos clientes.

Para o cálculo do atraso médio no sistema para o modelo de fila M/M/1 utilizou-se a seguinte fórmula:

$$W = \frac{L}{\lambda} = \frac{1}{\mu - \lambda}$$

μ - taxa de transmissão de pacotes pelo link.
λ - taxa de chegada de pacotes a ser transmitidos pelo link.

Para o cálculo do atraso médio no sistema para o modelo de fila M/G/1 procedeu-se a utilização da seguinte fórmula:

$$W = \frac{\lambda E[S^2]}{2(1-\lambda E[S])} + E[S]$$

λ - taxa de chegada de pacotes a ser transmitidos pelo link.
S - tempo de atendimento de pacotes.

Os valores teóricos requisitados nesta alínea obtiveram-se da seguinte forma:

```
format shortG
AvgPkt  = 64* 0.19 + 1518*0.48 + (((1517+65)/2) *(1-0.19-0.48));%bytes
lambdas = ([6 8 9 9.5 9.75 10.0 6 8 9 9.5 9.75 10]*(1e6/8))./AvgPkt;
link_capacity = 10e6./8; %bytes/s
u = link_capacity/AvgPkt;
w = (1./(u- lambdas))*1000;

%M/G/1
e_s = ((64*8)/10e6) * 0.19 + ((1518*8)/10e6) * 0.48 + (((1517+65)*8/2)/10e6) * (1-0.19-0.48);
```

```
e_s2 = (((64*8)/10e6).^2) * 0.19 + (((1518*8)/10e6).^2) * 0.48 + ((((1517+65)*8/2)/10e6).^2) *
(1-0.19-0.48);
for i = 1:length(lambdas)
  wg(i) =  (((lambdas(i)*e_s2)./(2*(1-(lambdas(i)*e_s)))) + e_s) * 1000;
end
```

Os resultados foram os seguintes:
Nota: os valores apresentados na coluna 'Avg. Packet Delay (ms)' foram os valores obtidos a partir da simulação, são apresentados na tabela por motivos de comparação.

Case	r (Mbps)	Avg. Packet Delay (ms)	Avg. Packet Delay(ms) M/M/1	Avg. Packet Delay(ms) M/G/1
A	6	1.623723	2.00366	1.58802
B	8	2.985540	4.00732	2.89895
C	9	5.766522	8.01464	5.52081
D	9.5	11.062033	16.02928	10.76452

Os valores teóricos que mais se aproximam ao desempenho do sistema é o modelo cujo processo de atendimento é do tipo determinístico, o modelo M/G/1, enquanto que para o modelo do tipo M/M/1 os valores obtidos para os tempos de atraso médios diferem significativamente à medida que o a taxa de transmissão dos pacotes aumenta. Está mais de acordo com o modelo M/G/1 devido ao tempo de atendimento do servidor ter uma distribuição genérica e independente das chegadas dos pacotes.

d)
Um sistema M/M/1/m é caracterizado pelo facto de o tempo de atendimento de um servidor ser exponencialmente distribuído com média $1/\mu$ e por o sistema acomodar um máximo de X clientes, neste caso, a fila de espera ter uma capacidade de armazenamento de X-1 pacotes.
Portanto, tendo em conta o tamanho médio dos pacotes a ser transmitido no sistema M/M/1/X,consideramos o seguinte:

X = número máximo de pacotes que a fila de espera pode armazenar + 1;

sendo X = 150 para os casos cujo tamanho da fila equivale a 150000 bytes (i.e os casos {A,B,C,D,E,F}) e X = 15 para os casos em que o tamanho da fila é equivalente a 15000 bytes (i.e os casos {G, H, I, J, K, L}).

Para a obtenção do número médio de pacotes perdidos foi utilizada a seguinte fórmula, pela propriedade PASTA:

$$P_m = \frac{(\lambda/\mu)^m}{\sum_{i=0}^{m}(\lambda/\mu)^i}$$

, onde m = X.

Obteve-se o tempo médio de atraso dos pacotes a partir da aplicação da seguinte fórmula:

$$\frac{\sum_{i=1}^{X-1} i * P(i)}{\lambda}$$

Os valores teóricos requisitados nesta alínea obtiveram-se da seguinte forma:

```
AvgPkt  = 64* 0.19 + 1518*0.48 + (((1517+65)/2) *(1-0.19-0.48));%bytes
f = floor([150000 15000]/AvgPkt); %number of packets that can be stored in queue
lambdas = ([6 8 9 9.5 9.75 10.0]*(1e6/8))./AvgPkt; %pkts/s
link_capacity = 10e6./8; %bytes/s
u = link_capacity/AvgPkt; %pkts/s
lambda_u = lambdas./u;
for j = 1:length(lambdas)
   for i = 0:(f(1)+1)
      res_a(j,i+1) = lambda_u(j).^i;
   end
end

for j = 1:length(lambdas)
   for i = 0:(f(1)+1)
     pa(j,i+1) = res_a(j,i+1)/ sum(res_a(j,:));
   end
end

for j = 1:length(lambdas)
    AvgPktLossA(j) = (res_a(j,f(1)+1) ./ sum(res_a(j,:)))*100;
end

for j = 1:length(lambdas)
   AvgPktOnSystemA(j)=0;
   AvgPktDelayA(j) = 0;
   for i = 1:(f(1))
     AvgPktOnSystemA(j) = AvgPktOnSystemA(j) + (i*pa(j,i+1));
   end
   AvgPktDelayA(j) = (AvgPktOnSystemA(j) ./ lambdas(j)).*1000;
end
```

```
for j = 1:length(lambdas)
   for i = 0:(f(2)+1)
      res_b(j,i+1) = lambda_u(j).^i;
   end
end

for j = 1:length(lambdas)
   for i = 0:(f(2)+1)
     pb(j,i+1) = res_b(j,i+1)/ sum(res_b(j,:));
   end
end

for j = 1:length(lambdas)
    AvgPktLossB(j) = (res_b(j,f(2)+1) ./ sum(res_b(j,:)))*100;
end

for j = 1:length(lambdas)
   AvgPktOnSystemB(j)=0;
   AvgPktDelayB(j) = 0;
   for i = 1:(f(2))
      AvgPktOnSystemB(j) = AvgPktOnSystemB(j) + (i*pb(j,i+1));
   end
   AvgPktDelayB(j) = (AvgPktOnSystemB(j) ./ lambdas(j)).*1000;
end
AvgPktLoss = [AvgPktLossA AvgPktLossB];
AvgPktDelay = [AvgPktDelayA AvgPktDelayB];
```

Os resultados obtidos foram os seguintes:
Nota: os valores apresentados na coluna 'Avg. Packet Loss (%)' e 'Avg. Packet Delay (ms)' foram os valores obtidos a partir da simulação, são apresentados na tabela por motivos de comparação.

Case	r (Mbps)	f (Bytes)	Avg. Packet Loss (%)	Avg. Packet Delay (msec)	Avg. Packet Loss (%) M/M/1/X	Avg. Packet Delay (msec) M/M/1/X
A	6	150000	0.0	1.623723	3.52043e-32	2.00366
B	8	150000	0.0	2.985540	7.26839e-14	4.00732
C	9	150000	0.0	5.766522	1.52107e-06	8.01462
D	9.5	150000	0.000008	11.062033	0.00240	15.97124
E	9.75	150000	0.006458	21.432391	0.05878	29.21360
F	10.0	150000	0.326935	60.819711	0.66225	59.31364

G	6	15000	0.001162	1.623549	0.03135	1.99386
H	8	15000	0.144837	2.873813	0.90509	3.43426
I	9	15000	0.868799	4.308028	2.80800	4.43632
J	9.5	15000	1.764129	5.288708	4.35523	4.89440
K	9.75	15000	2.435815	5.839905	5.26570	5.09103
L	10	15000	3.215775	6.366030	6.25000	5.25961

Comparando os valores do atraso médio de pacotes para um modelo de fila do tipo M/M/1/X verificamos que este modelo é mais adequado para os casos I, J, K e L, devido aos valores obtidos a partir da simulação serem próximos.
Relativamente à média de pacotes perdidos não se verifica casos em que os valores obtidos no simulador se aproximem aos valores obtidos no cálculo teórico da média de pacotes perdidos para o modelo de fila do tipo M/M/1/X.

Simulador 2

O simulador foi construído consoante o enunciado.
O código desenvolvido para o simulador 2 foi o seguinte:

```
function out = sim2(par)
  %Events
  Arrival     = 0;
  Departure   = 1;
  Terminate   = 2;
  Retransmit  = 3;

  %Link State
  FREE = 0;
  OCCUPIED = 1;

  path          = par.J;
  capacityLinks   = par.C;
  queueLinkSize   = par.f;
  numLinks      = length(par.C);
  numFlows      = length(par.r);

  Events     = [];
  AvgPktSize  = 64* 0.19 + 1518*0.48 + (1517+65)/2 *(1-0.19-0.48);
  time       = (par.r./8)./AvgPktSize;
  a          = (1./time);
```

```
linkState       = zeros(1,numLinks);
QueueOccupation  = linkState;
Queue           = cell(1,numLinks);

totalSize    = zeros(1,numFlows);
totalPackets = totalSize;
lostPackets  = totalSize;
delays       = totalSize;
transmittedBytes = totalSize;
transmittedPackets  = totalSize;

instant = linkState; %arrival instant oof the packet that is being
                %transmitted on link
Size    = linkState;  %Size of the packet that is being transmitted
                %on link

out.AvgPacketLoss = totalSize;
out.AvgPacketDelay = totalSize;

for i = 1:numFlows
   Events = [Events; Arrival exprnd(a(i)) i path{i}(1) ];
end

Events = sortrows([Events; Terminate par.S 0 0]);

while(Events(1,2) <= par.S)
   event    = Events(1,1);
   Clock    = Events(1,2);
   Flow     = Events(1,3);
   Link     = Events(1,4);
   Events(1,:) = [];
   if(event == Arrival)
      packetSize = pktSize();
      totalPackets(Flow)  = totalPackets(Flow) + 1;
      totalSize(Flow)     = totalSize(Flow) + packetSize;
      if(linkState(Link) == FREE)
        linkState(Link) = OCCUPIED;
        instant(Link)   = Clock;
        Size(Link)      = packetSize;
        timeToNextPacket = Clock + (packetSize*8)/capacityLinks(Link);
        currentPath = path{Flow};
        if(currentPath(end) == Link)
```

```
        Events = [Events; Departure timeToNextPacket Flow Link];
      else
        nextLink = currentPath(find(currentPath == Link) + 1);
        Events = [Events; Retransmit timeToNextPacket Flow nextLink];
      end

    elseif(QueueOccupation(Link) + packetSize <= queueLinkSize(Link))
      Queue{Link} = [Queue{Link}; Clock packetSize Flow];
      QueueOccupation(Link) = QueueOccupation(Link) + packetSize;
    else
      lostPackets(Flow) = lostPackets(Flow) + 1;
    end
    Events = sortrows([ Events; Arrival (exprnd(a(Flow)) + Clock) Flow Link],2);

elseif(event == Retransmit)
        currentPath               = path{Flow};
   pastLink = currentPath(find(currentPath == Link) -1);
   tmp_linkInstant  = instant(pastLink);
   tmp_linkSize     = Size(pastLink);
   if(QueueOccupation(pastLink) >0)
      q_instant       = Queue{pastLink}(1,1);
      q_syze          = Queue{pastLink}(1,2);
      q_flow          = Queue{pastLink}(1,3);
      q_path          = path{q_flow};
      instant(pastLink) = q_instant;
      Size(pastLink)    = q_syze;
      timeToNextPacket = Clock + (q_syze*8)/capacityLinks(pastLink);
      if(q_path(end) == pastLink)
        Events = sortrows([Events; Departure timeToNextPacket q_flow pastLink],2);
      else
        nextLink = q_path(find(q_path == pastLink) + 1);
        Events   = sortrows([Events; Retransmit timeToNextPacket q_flow nextLink],2);
      end
      Queue{pastLink}(1,:) = [];
      QueueOccupation(pastLink) = QueueOccupation(pastLink) - q_syze;
   else
      linkState(pastLink) = FREE;
   end
   %Tratar do retransmit do link
   if(linkState(Link) == FREE)
      linkState(Link) = OCCUPIED;
      instant(Link) = tmp_linkInstant;
```

```
      Size(Link) = tmp_linkSize;
      timeToNextDeparture = Clock + (tmp_linkSize*8)/capacityLinks(Link);
      if(currentPath(end) == Link)
        Events = sortrows([ Events; Departure  timeToNextDeparture  Flow Link],2);
      else
        next_link = currentPath(find(currentPath == Link) + 1);
        Events = sortrows([Events; Retransmit timeToNextDeparture Flow next_link],2);
      end
    elseif(QueueOccupation(Link) + tmp_linkSize <= queueLinkSize(Link))
      Queue{Link} = [Queue{Link}; tmp_linkInstant tmp_linkSize Flow];
      QueueOccupation(Link) = QueueOccupation(Link) + tmp_linkSize;
    else
      lostPackets(Flow) = lostPackets(Flow) + 1;
    end

  elseif(event == Departure)
    delays(Flow) = delays(Flow) + (Clock - instant(Link));
    transmittedBytes(Flow) = transmittedBytes(Flow) + Size(Link);
    transmittedPackets(Flow) = transmittedPackets(Flow) + 1;
    if(QueueOccupation(Link) > 0)
      q_instant   = Queue{Link}(1,1);
      q_size      = Queue{Link}(1,2);
      q_flow      = Queue{Link}(1,3);
      q_path      = path{q_flow};
      instant(Link) = q_instant;
      Size(Link)    = q_size;
      timeToNextPacket = Clock + (q_size*8)/capacityLinks(Link);
      if(q_path(end) == Link)
        Events = sortrows([Events; Departure timeToNextPacket q_flow Link],2);
      else
        nextLink = q_path(find(q_path == Link) + 1);
        Events   = sortrows([Events; Retransmit timeToNextPacket q_flow nextLink],2);
      end
      Queue{Link}(1,:) = [];
      QueueOccupation(Link) = QueueOccupation(Link) - q_size;
    else
      linkState(Link) = FREE;
    end
  else
    out.AvgPacketLoss = 100 * (lostPackets./totalPackets);
    out.AvgPacketDelay = delays./transmittedPackets * 1000;%ms
  end
```

```
  end

 % fprintf("LostPackets per flow %d %d %d, TotalPackets %d %d
%d\n",lostPackets(1),lostPackets(2),lostPackets(3),totalPackets(1),totalPackets(2),totalPackets(3));
  %fprintf("Packet Delay per flow %d %d %d, TransmittedPackets %d %d
%d\n",delays(1),delays(2),delays(3),transmittedPackets(1),transmittedPackets(2),transmittedPacke
ts(3));
  %fprintf("Transmitted bytes %d %d %d, Total Bytes %d %d %d\n",
transmittedBytes(1),transmittedBytes(2),transmittedBytes(3),totalSize(1),totalSize(2),totalSize(3));
  % fprintf("c'est fini \n");

end

function PacketSize = pktSize()
r = rand();
if(r < 0.19)
   PacketSize = 64;
   return
end
if(r > 0.52)
   PacketSize = 1518;
   return
end
PacketSize = randi([65 1517]);
end
```

e)

Os valores obtidos a partir da simulação foram os seguintes:

Flow	Average Packet Loss (%)	90% Conf	Avg. Packet Delay (ms)	90% Conf
1	0.000000	0.000000	7.012268	0.026511

f)

Os valores obtidos a partir da simulação foram os seguintes:

Flow	Average Packet Loss (%)	90% Conf	Avg. Packet Delay (ms)	90% Conf
1	0.001981	0.001123	18.164663	0.351195
2	0.001707	0.001212	44.406838	0.989540
3	0.000256	0.000422	26.212739	0.692119

g)

Assumindo que as filas se comportam com o modelo do tipo M/M/1 de forma independente, obtemos o tempo de atraso médio dos pacotes com as seguintes fórmulas:

$$N_{ij} = \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}} \qquad N = \sum_{i,j} N_{ij} \qquad T = \frac{N}{\lambda}$$

$$\lambda_{ij} = \sum_{\text{P traverses link (i,j)}} X_p$$

N - Número médio de pacotes na rede.
T - Tempo médio de atraso dos pacotes na rede.
Xp - Taxa de chegada de pacotes ao longo do caminho p.
λij - Taxa de chegada dos pacotes para link (i,j).
uij - Taxa de transmissão dos pacotes para o link (i,j)

Os valores teóricos requisitados nesta alínea obtiveram-se da seguinte forma:

```
%e
AvgPkt  = 64* 0.19 + 1518*0.48 + (((1517+65)/2) *(1-0.19-0.48));%bytes
capacity_ab = 10e6/8;%bytes/s
capacity_bc = 5e6/8;%bytes/s


u_ab = capacity_ab/AvgPkt; %packets/s
u_bc = capacity_bc/AvgPkt; %packets/s
lambda = (4e6/8)/(AvgPkt); % arrival rate: pakets/s
```

```
n_ab = lambda./(u_ab-lambda); %average packet in ab
n_bc = lambda./(u_bc-lambda); %average packet in bc

n = n_ab + n_bc;    % N= Average packets in network
res = (n/lambda) * 1000;      % Average packet delay in network;

%f
r1 = 7.4e6/8; %bytes/s
r2 = 2.3e6/8; %bytes/s
r3 = 2.5e6/8; %bytes/s

capacity_ab = 10e6/8;%bytes/s
capacity_bc = 5e6/8;%bytes/s

u_ab = capacity_ab/AvgPkt; %packets/s
u_bc = capacity_bc/AvgPkt; %packets/s


lambda1 = r1./AvgPkt; %arrival_rate link 1 : packet/s
lambda2 = r2./AvgPkt; %arrival_rate link 2 : packet/s
lambda3 = r3./AvgPkt; %arrival_rate link 3 : packet/s

%%Flow1:
lambda_flow1 = lambda1 + lambda2;
res_flow1 = (1/(u_ab - lambda_flow1))*1000;

%%Flow3
lambda_flow3 = lambda2 + lambda3;
res_flow3 = (1/(u_bc - lambda_flow3))*1000;


%%Flow2:
res_flow2 = (1/(u_ab - lambda_flow1) + 1/(u_bc - lambda_flow3))*1000;
```

Nota: o valor apresentado na coluna 'Average Packet Delay (ms)' das seguintes 2 tabelas é o valor prático obtido na simulação da alínea 'e' e 'f', respectivamente. É utilizada para fins de comparação com o valor obtido a partir da aproximação de Kleinrock.

Para o caso da alínea e) os resultados obtidos foram os seguintes:

Flow	Average Packet Delay (ms)	[Kleinrock] Average Packet Delay (ms)
1	7.0123	9.3504

Verifica-se que a diferença entre o valor prático obtido a partir da simulação e o valor teórico obtido a partir da aproximação de Kleinrock é significante, sendo o resultado obtido pela aproximação de Kleinrock superior por mais de 2 unidades. Para este caso, na prática a partir da simulação deste caso, não se verificou perda de pacotes, assemelhando-se o comportamento da fila ao modelo de fila do tipo M/M/1. No entanto trata-se de uma rede com um único percurso por fluxo, logo a fonte de erro associada a aproximação de Kleinrock deve-se a correlação entre os comprimentos dos pacotes e os intervalos de chegadas dos pacotes. Portanto, verifica-se que a aproximação de Kleinrock não é uma boa aproximação para este caso.

Para os casos da alínea f) os resultados obtidos foram os seguintes:

flow	Average Packet Delay (ms)	[Kleinrock] Average Packet Delay (ms)
1	18.165	26.715
2	44.407	66.789
3	26.213	40.073

Verifica-se que a diferença entre o valores práticos obtidos a partir da simulação e os valores teóricos obtidos a partir da aproximação de Kleinrock é significante, sendo o resultado obtido pela aproximação de Kleinrock bastante superiores. Para este caso não só a correlação entre os comprimentos dos pacotes e os intervalos de chegadas afeta o erro associado à aproximação de Kleinrock. Um fator adicional de erro deve-se ao facto de esta aproximação ser necessário admitir que as ligações são modeladas por um sistema M/M/1, isto é, as filas de esperas são infinitas. Às filas para este caso são finitas, a aplicação da aproximação de Kleinrock não é adequada pois não cumpre com o requisito mencionado. Como se verificou na simulação para a alínea f) verificou-se que todos os pacotes que chegaram ao sistema não conseguiram ser transmitidos até ao seu destino, tendo sido perdidos por causa do tamanho da fila de espera de cada um dos links do sistema ser de tamanho limitado e não suficiente para armazenar os pacotes necessários. Portanto não está de acordo com o modelo M/M/1. Sendo o tamanho limitado, o tempo de atraso ou espera que um pacote demora estar na fila, se houver espaço suficiente, até o seu turno de ser transmitido chegar, também será limitado. Caso um pacote não conseguir entrar na fila, o sistema não irá transmiti-lo, logo é perdido. O valor obtido a partir da aproximação de Kleinrock tende ser bastante superior do que o obtido na prática para este caso devido a se considerar que a fila de espera é infinita e que

todos os pacotes que chegam ao sistema consigam ser colocados na fila se necessário e transmiti-los no seu turno. Quanto maior a taxa de chegada de pacotes maior será o número de pacotes a serem depositados na fila e quanto mais ocupada a fila se encontrar por vários pacotes, maior será a demora do próximo pacote que chegou ao sistema para ser transmitido no seu turno. Logo, daí os valores da aproximação de Kleinrock serem bastante superiores aos valores obtidos na simulação.

h)

Os valores obtidos na alínea f foram os seguintes:

flow	Average Packet Loss (%)	90% Conf	Avg. Packet Delay (msec)	90% Conf
1	0.001981	0.001123	18.164663	0.351195
2	0.001707	0.001212	44.406838	0.989540
3	0.000256	0.000422	26.212739	0.692119

Os valores obtidos para esta alínea foram os seguintes:

flow	Average Packet Loss (%)	90% Conf	Avg. Packet Delay (msec)	90% Conf
1	5.346815	0.018608	3.198475	0.001634
2	5.361737	0.028039	17.264694	0.244895
3	0.000000	0.000000	14.138825	0.239468

Verifica-se que os valores dos fluxos 1 e 2 os valores médio de perda de pacotes aumentam para mais de 5% devido a fila de espera de pacotes diminuir de 150000 bytes para 7500 bytes. A perda de pacotes deve-se ao tamanho da fila do link 1 não ser suficiente para o armazenamento de todos os pacotes que chegam ao link 1 para a dada taxa de chegada de pacotes a ser transmitidos para um dado flow que passa pelo link 1 e a capacidade do link 1 não ser suficiente para evitar a perda de pacotes. Sendo o tamanho da fila pequena e limitada, não sendo suficiente para depositar todos os pacotes que chegam ao link 1 a ser transmitidos, a média do tempo de demora do pacote a ser transmitido também é limitada. Daí o tempo médio de demora ser menor quando o tamanho da fila diminuir, pois se a fila estiver cheia o pacote é ignorado e não é transmitido. Enquanto que no caso de f o tamanho da fila do link é suficiente para armazenar a maior parte dos pacotes que chegam ao sistema, quanto maior for o número de pacotes a espera na fila maior será o tempo de demora do próximo pacote que chegar ao link ser transmitido, daí os valores de demora ou atraso serem maiores que nos casos desta alínea em que o tamanho da fila do link 1 é equivalente a 7500 bytes. O tempo médio de demora diminui significativamente para o terceiro flow devido ao facto de este valor ser influenciado pelas perdas dos pacotes de fluxos distintos ao flow 3 que passam pelo

link 2 antes de passar por este. Isto leva com que a ocupação da fila seja menor, logo menor perda de pacotes e menor tempo de atraso de transmissão dos pacotes pelo link 2.

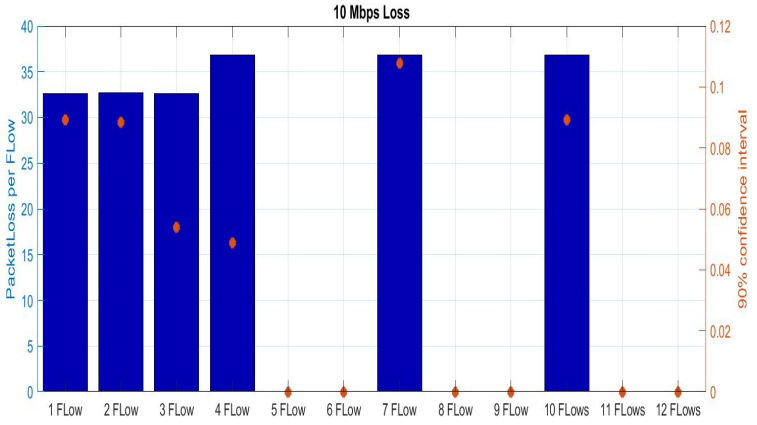
i)

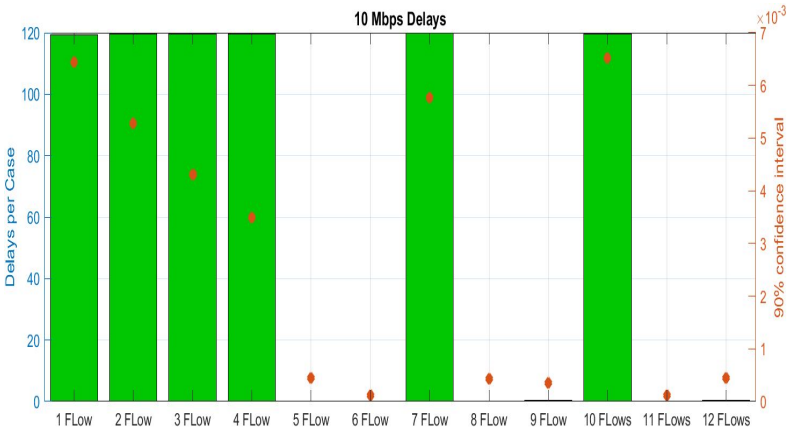
Nesta Alínea encontra-mos o valor de 30 Mbps, como a capacidade mínima necessária da ligação da rede em causa a internet e para garantir que não existem pacotes perdidos nem atrasos superiores a 1ms.
Tendo chegado a esta conclusão, independentemente da capacidade da ligação, verificámos, tal como esperado, que apenas os fluxos entre intervenientes interiores na rede se mantém incólume pelo engarrafamento no acesso à internet.
A partir do momento que a ligação tem maior capacidade (20 Mbps) tal que a soma das taxas de chegadas de pacotes dos fluxos que entram no sistema através da ligação à internet é inferior a capacidade da ligação, verificámos a inexistência de pacotes perdidos, e um decréscimo significativo no atraso dos pacotes em causa, mantendo-se as constatações dos pacotes dos fluxos "upstream" em direção à internet, uma enorme perda de pacotes (por volta dos 25%) e um atraso não passável de volta dos 28 ms, tendo em conta tratar-se de uma ligação ponto a ponto.
Desta forma, assim que a capacidade da ligação alcançou valores que contemplam tanto os fluxos da internet como para ela, foi observado a ausência de perdas e uma hegemonização dos atrasos inferiores, em todos os fluxos, a 1ms.

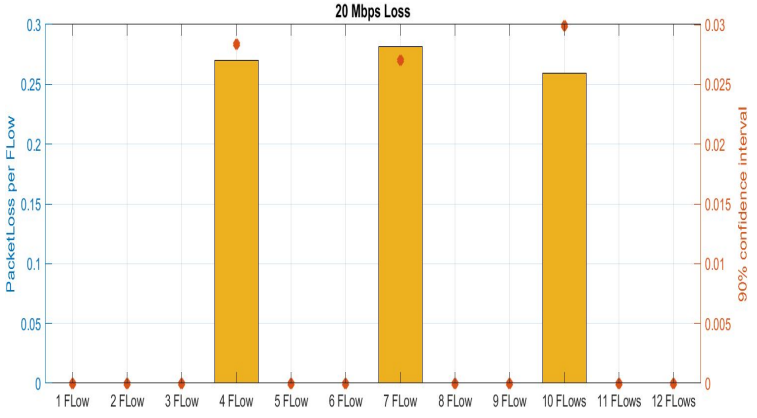
Resultados doravante referentes a alínea i) :

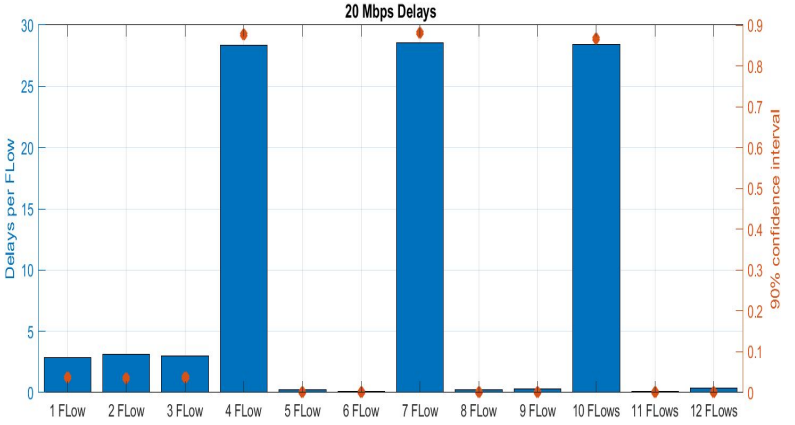
	X = 10 Mbps		X = 20 Mbps		X = 30 Mbps	
Flows	Average Packet Loss (%)	Average Packet Delay (ms)	Average Packet Loss (%)	Average Packet Delay (ms)	Average Packet Loss (%)	Average Packet Delay (ms)
From Internet to Datacenter	32.622797 ± 0.089153	19.420224 ± 0.006440	0.000000 ± 0.000000	2.864723 ± 0.037099	0.000000 ± 0.000000	0.540714 ± 0.001470
From Internet to LAN A	32.677163 ± 0.088397	119.634618 ± 0.005283	0.000000 ± 0.000000	3.100683 ± 0.036277	0.000000 ± 0.000000	0.790797 ± 0.001225
From Internet to LAN B	32.612435 ± 0.053897	119.523856 ± 0.004315	0.000000 ± 0.000000	2.972467 ± 0.038195	0.000000 ± 0.000000	0.652989 ± 0.001098
From Datacenter to Internet	36.811020 ± 0.048816	119.566005 ± 0.003490	0.270360 ± 0.028385	28.342042 ± 0.876177	0.000000 ± 0.000000	0.630998 ± 0.001111
From Datacenter to LAN A	0.000000 ± 0.000000	0.226452 ± 0.000452	0.000000 ± 0.000000	0.248304 ± 0.000474	0.000000 ± 0.000000	0.249377 ± 0.000671
From Datacenter to LAN B	0.000000 ± 0.000000	0.107003 ± 0.000110	0.000000 ± 0.000000	0.112083 ± 0.000083	0.000000 ± 0.000000	0.112382 ± 0.000093

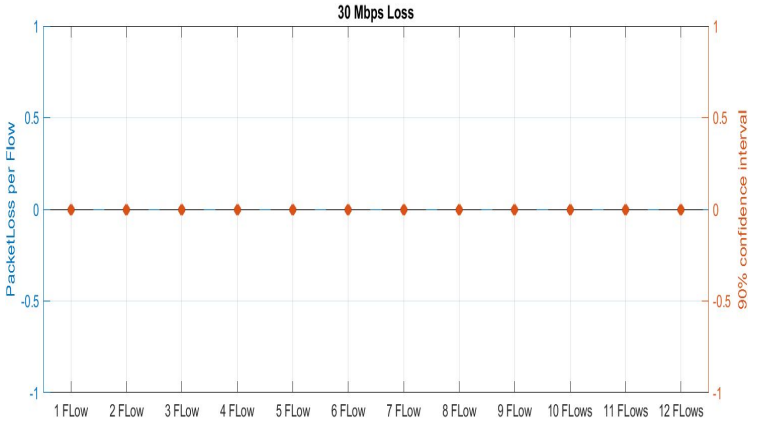
From A to Internet	36.781449 ± 0.107887	119.776369 ± 0.005767	0.281564 ± 0.026982	28.550897 ± 0.881639	0.000000 ± 0.000000	0.851666 ± 0.001495
From A to Datacenter	0.000000 ± 0.000000	0.220742 ± 0.000427	0.000000 ± 0.000000	0.221040 ± 0.000243	0.000000 ± 0.000000	0.221121 ± 0.000266
From A to B	0.000000 ± 0.000000	0.324954 ± 0.000360	0.000000 ± 0.000000	0.330524 ± 0.000304	0.000000 ± 0.000000	0.330753 ± 0.000282
From B to Internet	36.796820 ± 0.089147	119.663485 ± 0.006516	0.259422 ± 0.029931	28.438654 ± 0.866987	0.000000 ± 0.000000	0.739606 ± 0.001809
From B to Datacenter	0.000000 ± 0.000000	0.105952 ± 0.000122	0.000000 ± 0.000000	0.106019 ± 0.000071	0.000000 ± 0.000000	0.105905 ± 0.000083
From B to A	0.000000 ± 0.000000	0.335290 ± 0.000455	0.000000 ± 0.000000	0.357303 ± 0.000476	0.000000 ± 0.000000	0.358658 ± 0.000765

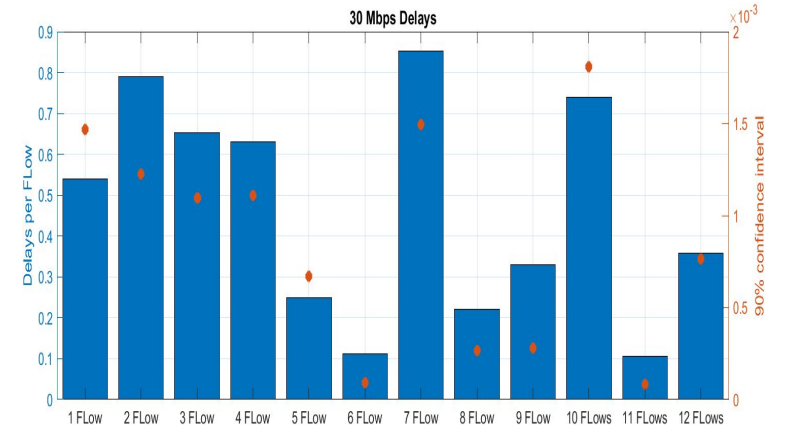

10 Mbps Loss
PacketLoss per FLow
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows


10 Mbps Delays
Delays per Case
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows


20 Mbps Loss
PacketLoss per FLow
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows


20 Mbps Delays
Delays per FLow
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows


30 Mbps Loss
PacketLoss per Flow
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows


30 Mbps Delays
Delays per FLow
90% confidence interval
1 FLow
2 FLow
3 FLow
4 FLow
5 FLow
6 FLow
7 FLow
8 FLow
9 FLow
10 FLows
11 FLows
12 FLows