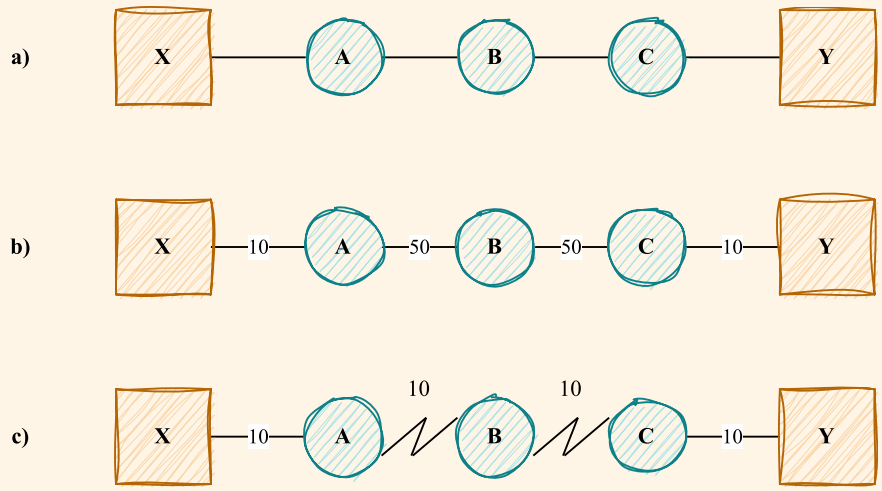
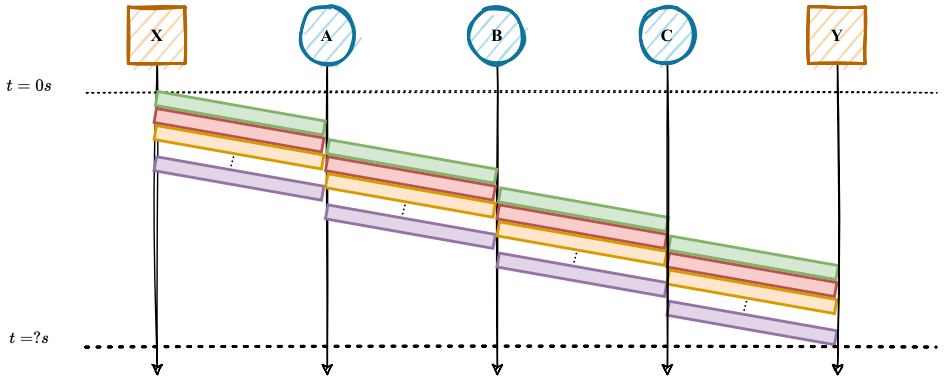
1. Considere um sistema terminal X que pretende enviar um ficheiro com 100 K_iB a um sistema terminal Y através de um caminho com quatro ligações como ilustrado nas figuras abaixo. É usada uma tecnologia de comutação de pacotes, contendo cada pacote 5 K_iB. Os atrasos de propagação nas ligações, bem como os atrasos de processamento nos nós, são desprezáveis.



(a) Na topologia da Figura a), cada uma das quatro ligações é cablada com ritmo de transmissão 10 $Mbps$. Desenhe um diagrama espaço-tempo ilustrando a transferência do ficheiro e calcule o atraso na entrega do mesmo, desde a transmissão do primeiro bit por X até à receção do último bit por Y.

Em primeiro lugar, é relevante realçar que estas transmissões utilizam o mecanismo de *Store-and-forward*: cada pacote só pode avançar quando todo o seu conteúdo chega ao *hop* atual. Assim sendo, considerando três *hops* intermédios entre os sistemas terminais, teremos o seguinte:



Descobrir a quantidade de tempo que demora a transferir todo o ficheiro entre sistemas terminais é relativamente simples. É particularmente relevante considerar que apenas se sentem atrasos "no que ao primeiro pacote diz respeito": é o único com espera "inútil", onde o canal à frente está livre mas é obrigado a esperar. O atraso corresponde precisamente ao produto entre o número de *hops* e o tempo de transmissão de um pacote, sendo que este é calculado através da expressão L/R, onde L é a quantidade de bits a transmitir e R é o ritmo de transmissão. Considerando que temos $5K_iB = 5 \cdot 2^{10} \cdot 2^3$ bits a transmitir, temos:

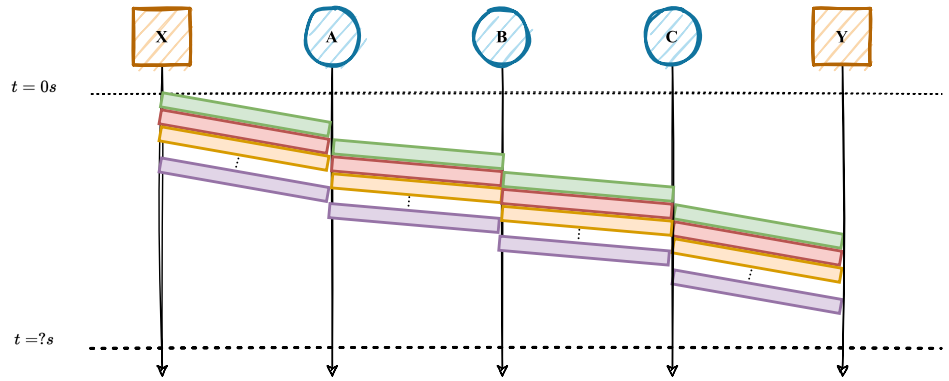
$$\begin{aligned}\text{Atraso} &= 3 \cdot \frac{L}{R} \\ &= 3 \cdot \frac{5 \cdot 2^{13}}{10 \cdot 10^6} \text{ s}\end{aligned}$$

Assim sendo, o tempo total de transmissão corresponderá à soma entre o atraso e o tempo de transmissão de todos os pacotes sem atraso, ou seja:

$$\begin{aligned}\text{T} &= \text{Atraso} + N \cdot \frac{L}{R} \\ &= (3 + 20) \cdot \frac{5 \cdot 2^{13}}{10 \cdot 10^6} \text{ s} \\ &= 0.094208 \text{ s}\end{aligned}$$

(b) Na topologia da Figura b), as ligações (A, B) e (B, C), e apenas estas, passaram a um ritmo de transmissão de 50 $Mbps$. Repita a alínea anterior para este caso.

A alteração não é particularmente significativa, afetando apenas a transmissão dos pacotes intra-*hops*. Neste sentido, o diagrama espaço-tempo será o seguinte (note-se que a diferença centra-se no tempo de transmissão de cada pacote):



Neste caso, o atraso é calculado de forma ligeiramente diferente, uma vez que o tempo de transmissão de cada pacote é agora variável - tanto podemos considerar $\frac{L}{R_1}$, com $R_1 = 10\ Mbps$, como $\frac{L}{R_2}$, com $R_2 = 50\ Mbps$. Assim sendo, e atendendo ao diagrama exposto acima (onde o primeiro pacote *espera* duas vezes num canal com ritmo R_2 e uma num canal com ritmo R_1), temos:

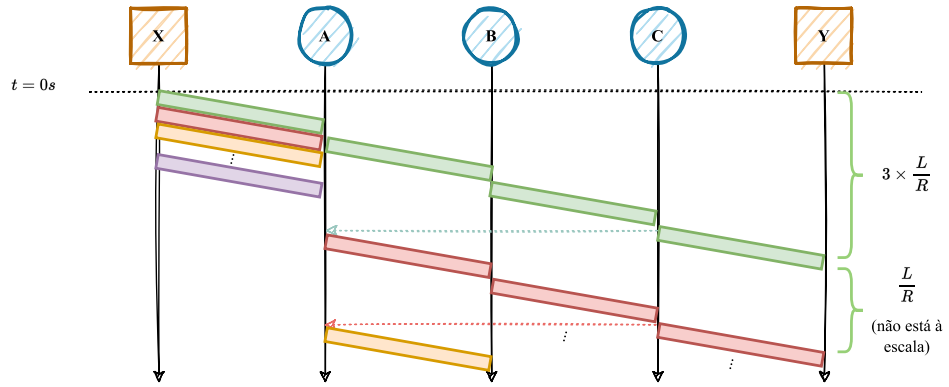
$$\begin{aligned}\text{Atraso} &= \frac{L}{R_1} + 2 \cdot \frac{L}{R_2} \\ &= L \cdot \left(\frac{1}{R_1} + 2 \cdot \frac{1}{R_2} \right)\end{aligned}$$

Para além do atraso, temos ainda o tempo de transmissão de todos os pacotes, que é calculado da mesma forma que na alínea anterior:

$$\text{T} = \text{Atraso} + N \cdot \frac{L}{R_1} = L \cdot \left(\frac{1}{R_1} + 2 \cdot \frac{1}{R_2} + \frac{20}{R_1} \right) = 0.087654 \text{ s}$$

(c) Na topologia da Figura c), as ligações (A, B) e (B, C) passaram a ser ligações sem-fios a 10 $Mbps$. As antenas usadas são omnidirecionais (o nó B não pode receber e transmitir simultaneamente) e assume-se que os nós A e C não conseguem escutar as transmissões um do outro. Desenhe um diagrama espaço-tempo ilustrando a transferência do ficheiro e calcule o atraso mínimo na entrega do mesmo, desde a transmissão do primeiro bit por X até à receção do último bit por Y.

Neste caso, o diagrama espaço-tempo é o seguinte:



Note-se que o atraso inicial é exatamente igual ao da primeira alínea - o primeiro pacote não vê alterações no seu comportamento. Contudo, os pacotes seguintes passam a ter comportamento "alternado", passando a poder avançar apenas quando B fica completamente desocupado (e não quando $A - B$ fica desocupado). Assim sendo, o atraso adicional será, claro, maior (o dobro):

$$\begin{aligned} \text{T} &= \text{Atraso} + 2N \cdot \frac{L}{R} \\ &= (3 + 2 \cdot 20) \cdot \frac{5 \cdot 2^{13}}{10 \cdot 10^6} \text{ s} \\ &= 0.176128 \text{ s} \end{aligned}$$

2. Os sistemas terminais A e B estão ligados a um mesmo comutador de pacotes. O sistema A gera pacotes de dados com 480 bytes cada a intervalos regulares de T segundos, sendo o primeiro desses pacotes transmitido no instante $t = 0ms$. A ligação de A ao comutador tem débito 1 $Mbps$. O sistema B gera pacotes de voz codificada a 64 $kbit/s$ a intervalos regulares de 20 ms, sendo o primeiro desses pacotes transmitido no instante $t = 3ms$. A ligação de B ao comutador tem também débito 1 $Mbps$. A linha de saída do comutador tem débito 256 $kbit/s$. Desprezam-se os atrasos de processamento e de propagação.

(a) Determine a gama de valores de T para os quais o sistema é estável, i.e., tal que o número de pacotes à espera de transmissão à saída do comutador não aumenta indefinidamente no tempo.

O sistema é estável caso o ritmo a que os pacotes de dados chegam ao comutador seja menor ou igual ao ritmo a que os pacotes de dados são transmitidos na linha de saída do mesmo. Note-se que, tal como

na alínea anterior, estamos a considerar um mecanismo de *Store and Forward*, pelo que o pacote terá de ser completamente recebido no comutador antes de ser transmitido na linha de saída. Assim sendo:

$$\text{output rate} \geq \text{input rate} \leftrightarrow 256 \text{ kbit/s} \geq 64 \text{ kbit/s} + \text{rate de A}$$

Neste sentido, o ritmo de saída de A não poderá exceder 192 $kbit/s$ - com pacotes de 480 bytes, isto quer dizer que só devemos enviar pacotes de A a intervalos de $T \geq \frac{480\times2^3}{192\times10^3} = 20ms$.

(b) Assumindo que T é tal que o sistema é estável e que a linha de saída do comutador transmite pacotes na ordem de chegada, independentemente de serem de dados ou de voz, diga qual o atraso máximo na transmissão dos pacotes de voz, medido desde o momento em que um bit está disponível à saída do codificador de voz até que é transmitido na linha de saída do comutador.

Nota: exercício atualmente errado, de acordo com as soluções.

Sendo o sistema estável, temos como pior cenário o caso em que os pacotes de dados e voz chegam ao mesmo tempo ao comutador, e em que o comutador transmite primeiro o pacote de dados. Neste caso, a quantidade de tempo pedida corresponde ao tempo de transmissão de um pacote de voz entre B e o comutador, somado ao tempo de transmissão de um pacote de dados entre o comutador e o exterior.

Em primeiro lugar, é importante notar que não é dado o tamanho de um pacote de voz. Sabemos, contudo, que é criado um pacote a cada $20ms$, a um ritmo de 64 $kbit/s$. Assim sendo, são criados 50 pacotes por secundo, pelo que cada um deles terá $\frac{64\times10^3}{50} = 1280$ bits. O tempo de transmissão de um pacote de voz entre B e o comutador é, portanto, dado por:

$$t_{B-com} = 1.28 \times 10^{-3} \text{ s} = 1.28 \text{ ms}$$

O tempo de transmissão de um pacote de dados entre o comutador e o exterior é dado pela seguinte expressão:

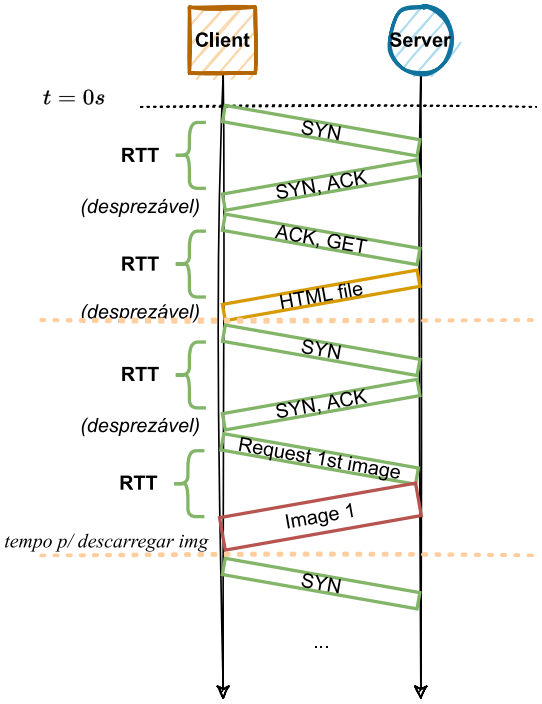
$$t_{com-out} = \frac{480 \times 2^3}{256 \times 10^3} = 15 \text{ ms}$$

Assim sendo, o atraso máximo na transmissão dos pacotes de voz é dado por $0.128 + 15 = 16.28\ ms$.

3. Pretende-se estimar o tempo necessário à descarga de um documento da Web. O documento é constituído por um objeto HTML base que referencia três imagens de tamanho 5000 byte cada. A dimensão do objeto base bem como de todos os pacotes de controlo é desprezável. O cliente liga-se ao servidor através de uma sessão de débito constante igual 1 $Mbps$ e tempo de ida-e-volta (RTT) igual a 20 ms. Qual o tempo necessário para descarregar o documento em cada uma das condições seguintes:

(a) O browser usa HTTP não-persistente sem sessões TCP paralelas.

Utilizando HTTP-não persistente (e sem paralelismo de sessões), o browser terá forçosamente de abrir uma nova sessão TCP para cada pedido HTTP. Note-se que haverá um pedido inicial para o objeto base e três pedidos adicionais para as imagens. O diagrama espaço-tempo abaixo ilustra o comportamento em causa:

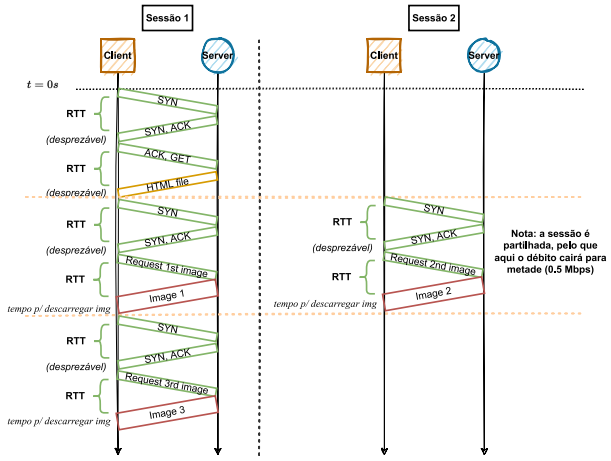


Vão ser realizados 4 pares ”SYN - SYN-ACK” e 4 pares ”ACK - Resposta”, um por pedido HTTP; assim sendo, teremos no mínimo 8 RTTs para estabelecer as sessões TCP. Considerando o tempo de descarga do ficheiro HTML como desprezável, resta adicionar o tempo de descarga das imagens (igual para cada uma):

$$
\begin{aligned}
T_{total} &= 8 \times RTT + 3 \times \frac{5000 \times 2^3}{1 \times 10^6} \\
&= 8 \times 20 \times 10^{-3} + 3 \times \frac{5000 \times 2^3}{1 \times 10^6} \\
&= 0.16 + 0.12 \\
&= 0.28 \text{ s}
\end{aligned}
$$

(b) O browser usa HTTP não-persistente, permitindo, no máximo, duas sessões TCP paralelas.

Neste cenário, o diagrama ficará como se segue:

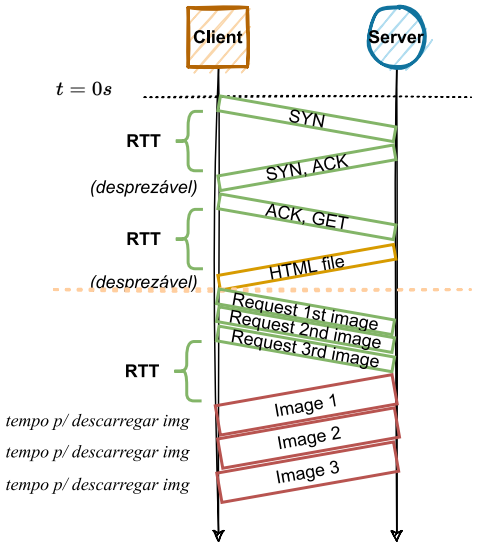


Assim sendo, passamos de 8 para 6 RTTs para estabelecer as sessões TCP e fazer os pedidos HTTP. O tempo de descarga do ficheiro HTML é desprezável. Duas das imagens são descarregadas em simultâneo, a metade do ritmo habitual, e a terceira imagem é descarregada após o término das duas primeiras (que ocorrem em simultâneo). Assim sendo, a quantidade de tempo pretendida é dada por:

$$\begin{aligned} T_{total} &= 6 \times RTT + 2 \times \frac{5000 \times 2^3}{0.5 \times 10^6} + \frac{5000 \times 2^3}{1 \times 10^6} \\ &= 6 \times 20 \times 10^{-3} + 2 \times \frac{5000 \times 2^3}{0.5 \times 10^6} + \frac{5000 \times 2^3}{1 \times 10^6} \\ &= 0.12 + 0.08 + 0.04 \\ &= 0.24 \text{ s} \end{aligned}$$

(c) O browser usa HTTP persistente com pipelining (sem sessões paralelas).

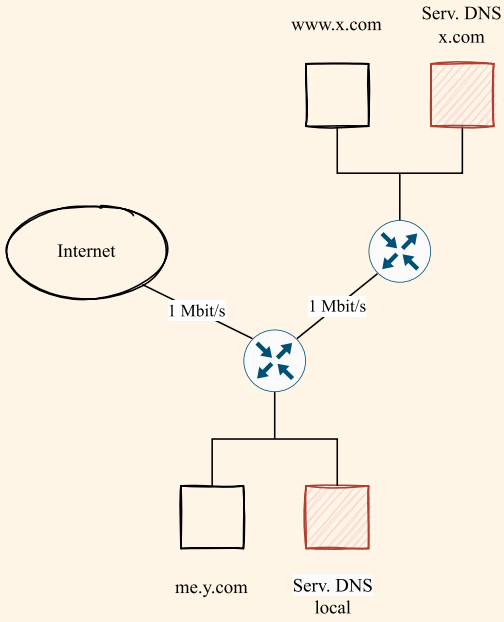
Com HTTP persistente e pipelining, o diagrama fica como se segue:



Note-se que as diferenças quanto à primeira alínea centram-se no facto de que neste caso, o browser não precisa de abrir novas sessões TCP para cada pedido HTTP, e ainda em haver pipelining, permitindo a descarga das três imagens num único pedido. Assim sendo, o número de RTTs para estabelecer as sessões TCP passa para 3: o inicial para estabelecer a sessão TCP e os 4 pedidos HTTP. Com contas relativamente semelhantes, podemos assim chegar ao seguinte resultado:

$$\begin{aligned} T_{total} &= 3 \times RTT + 3 \times \frac{5000 \times 2^3}{1 \times 10^6} \\ &= 3 \times 20 \times 10^{-3} + 3 \times \frac{5000 \times 2^3}{1 \times 10^6} \\ &= 0.06 + 0.12 \\ &= 0.18 \text{ s} \end{aligned}$$

4. Considere as redes da figura, na qual Serv. DNS local é o servidor de nomes configurado em `me.y.com` e Serv. DNS `x.com` é o servidor de nomes idóneo (Authoritative Name Server) para o domínio `x.com`. O tempo de propagação a separar um encaminhador do outro é 100 *ms* e o tempo de propagação entre as máquinas da rede `y.com` e a Internet é de 500 *ms* (num sentido apenas). Inicialmente, o Serv. DNS local tem a sua cache de nomes vazia.



(a) Suponha que o utilizador de um browser em `me.y.com` digita a URL `www.x.com/index.html`. Apresente a sequência ordenada de todas as mensagens DNS e HTTP trocadas até que o ficheiro `index.html` seja recebido em `me.y.com`. Relembra-se que a pesquisa de nomes a partir de Serv. DNS local é iterativa. Para cada mensagem indique a sua origem, o seu destino e o tipo de mensagem. (Seja tão preciso quanto possível relativamente ao tipo da mensagem).

A tabela seguinte corresponde a uma listagem ordenada das mensagens DNS e HTTP trocadas.

	Tipo	**Origem**	**Destino**	**Comentário**
1	DNS Query	me.y.com	Serv. DNS local	-
2	DNS Query	Serv. DNS local	Root Name Server	-
3	DNS Response	Root Name Server	Serv. DNS local	Root não sabe, aponta p/ TLD
4	DNS Query	Serv. DNS local	TLD Server	-
5	DNS Response	TLD Server	Serv. DNS local	TLD não sabe, aponta p/ Serv. DNS de x.com
6	DNS Query	Serv. DNS local	Serv. DNS x.com	-
7	DNS Response	Serv. DNS x.com	Serv. DNS local	Serv. DNS de x.com indica o IP da máquina
8	DNS Response	Serv. DNS local	me.y.com	me.y.com terá agora o IP de x.com
9	GET HTTP/1.1	me.y.com	www.x.com	Pedido index.html
10	HTTP 200 OK	www.x.com	me.y.com	Ficheiro enviado

(b) Sabendo que o ficheiro `index.html` residente na máquina `www.x.com` tem 1 Mbit, determine o atraso desde o momento em que o utilizador digita a URL em `me.y.com` até que o ficheiro é recebido na totalidade. Considere que o mecanismo de arranque lento não está ativo e despreze o tempo de transmissão de todos os segmentos exceto aqueles que contêm dados do ficheiro `index.html`.

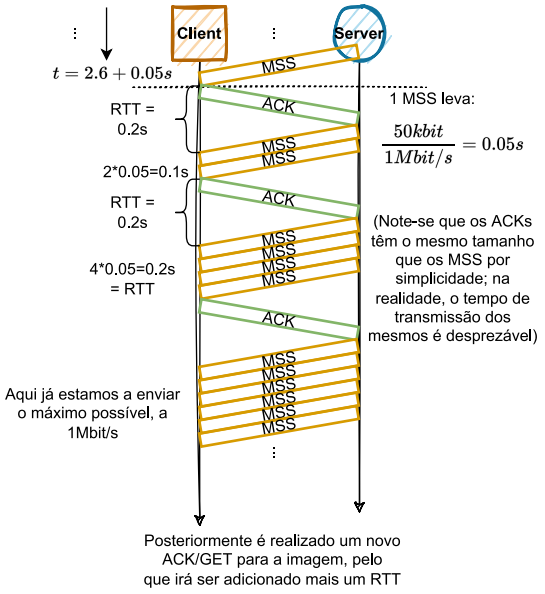
Para calcular o atraso pretendido, faz sentido suportar o nosso pensamento na tabela anterior. Note-se que o tempo de propagação entre `me.y.com` e o respetivo servidor de nomes é considerado desprezável.

- As mensagens 2 e 3, entre Serv. DNS local e Root Name Server, levam $2 \cdot 500\ ms$, já que o tempo de propagação num só sentido é de $500\ ms$.
- As mensagens 4 e 5, entre Serv. DNS local e TLD Server, levam $2 \cdot 500\ ms$.
- As mensagens 6 e 7, entre Serv. DNS local e Serv. DNS x.com, levam $2 \cdot 100\ ms$, já que aqui não vamos à *black-box* Internet, passando diretamente entre comutadores.
- As mensagens 9 e 10 levam $2 \cdot 2 \cdot 100\ ms$, contando com o par SYN/ACK e o par ACK GET/transmissão do ficheiro.

Assim sendo, o atraso total é de $2 \cdot 500 + 2 \cdot 500 + 2 \cdot 100 + 2 \cdot 2 \cdot 100 + 1000 = 3600\ ms$.

(c) Assuma agora que: (i) o ficheiro `index.html` referencia uma imagem com 5 $Mbits$; (ii) o browser usa sessões HTTP persistentes; (iii) o mecanismo de arranque lento está ativo; (iv) e o MSS é 50 $kbit$. Determine o atraso na receção da página, constituída pelo ficheiro mais a imagem.

Em primeiro lugar, é relevante realçar que é mencionada a existência de sessões HTTP persistentes, mas nada é referido no que a pipelining diz respeito. Neste sentido, essa noção será aqui descartada. Mais ainda, é referido o mecanismo de arranque lento, com MSS de 50 $kbit$. Note-se que esta alteração não afeta os tempos de propagação até ao envio do primeiro segmento de *index.html* (inclusive), pelo que a *baseline* temporal será de 2600 ms.
As mensagens seguintes verão um diagrama espaço-tempo como o que se segue:



Assim sendo, aos 2600 ms da *baseline* acrescem os seguintes tempos:

- 0.05 segundos para a transferência do primeiro segmento;
- 0.2 + 0.1 segundos para a transferência dos dois segmentos seguintes;
- 0.2 + 0.2 segundos para a transferência dos quatro segmentos seguintes. Note-se que aqui os segmentos levam precisamente o equivalente a um RTT a ser transferidos, pelo que a partir daqui iremos transferir todos os segmentos de forma contínua.
- $0.2 + x$ segundos para a transferência dos restantes segmentos - aqui, após o arranque lento, estamos finalmente num cenário de transferência de dados contínuo. Tendo já sido transferidos três segmentos, resta-nos transferir $N - 3$ segmentos, onde N corresponde ao número de segmentos necessários para transmitir o ficheiro `html` e respetiva imagem.

$$N = \frac{5 \cdot 10^6 + 1 \cdot 10^6}{50 \cdot 10^3} = 120$$

Restando transferir $N-7 = 113$ segmentos, temos que o tempo de transferência dos restantes segmentos será dado por $x = 0.05 \cdot 113 = 5.65$ segundos. Note-se que será ainda adicionado um RTT, 200 ms, para a transferência do ficheiro de imagem.

Assim, temos que no total, a interação em questão leva $2600+50+200+100+200+200+5650+200 = 9200\ ms$.

5. Considere três estações que pretendem recuperar um ficheiro com 100 MiB de um servidor. A largura-de banda uplink do servidor, do servidor à Internet, é de 1 $Mbit/s$. A largura de banda downlink de cada estação, da Internet à estação, é muito elevada (superior a 1 $Mbit/s$) e a largura de banda uplink de cada estação, da estação à Internet, é 200 $kbit/s$. Despreze os atrasos de propagação.

(a) Com uma aplicação cliente servidor, qual o tempo mínimo necessário para que o ficheiro seja distribuído a todas as estações?

O servidor terá de distribuir o envio do ficheiro por três clientes, pelo que a largura de banda uplink do mesmo será dividida por três. Assim sendo, e considerando que a largura de banda downlink de cada estação é muito elevada (pelo que este não será o *bottleneck*), o tempo mínimo necessário para que o ficheiro seja distribuído a todas as estações é dado por:

$$T_{min} = 3 \times \frac{100 \text{ MiB}}{1 \text{ Mbit/s}} = 3 \times \frac{100 \times 2^{20} \times 2^3}{1 \times 10^6} = 2516.5824 \text{ s}$$

(b) Considere agora uma aplicação peer-to-peer em que as estações usam as suas larguras de banda uplink para ajudar a distribuir o ficheiro. O servidor envia dados a cada uma das estações continuamente. Para além disso, cada estação vai redistribuir a cada uma das outras duas estações os dados que recebe do servidor. Encontre uma boa estratégia para distribuir o ficheiro por todas as estações, e de acordo com ela diga qual o tempo mínimo necessário para a distribuição do ficheiro?

O ficheiro é distribuído por todas as estações, sendo dividido em N (com $N = 3$) partes distintas, uma por estação. Cada estação recebe uma parte do ficheiro e envia a cada uma das outras duas estações

a sua parte. Assim sendo, a largura de banda uplink de cada estação será dividida por $N - 1$. Assim sendo:

$$T_{upload-servidor} = 3 \times \frac{\frac{100}{3} \times 2^{23}}{1 \times 10^6} = 838.8608 \text{ s}$$

Resta ainda calcular o tempo de upload de cada estação para as outras duas estações.

$$T_{upload-estacao} = \frac{N \times L}{u_{servidor} + \sum u_{estacao}} = \frac{3 \times 100 \times 2^{23}}{10^6 + 3 \times 200 \times 10^3} = 1572.864 \text{ s}$$

Note-se que o tempo mínimo necessário para a distribuição do ficheiro é dado (considerando que o *download* não é o *bottleneck*) por:

$$T_{min} = \max\{T_{upload-servidor}, T_{upload-estacao}\} = 1572.864 \text{ s}$$

6. Considere um caminho de 5000 km de comprimento com atraso de propagação igual a 5 $\mu s/km$, e sobre o qual consegue transmitir a um débito máximo de 100 Mbit/s. Suponha que usa um algoritmo de janela deslizante para controlo de erros e controlo de fluxo. Cada pacote tem 1000 bits. Se usar Go-Back-N qual a dimensão mínima da janela, em número de pacotes, que garante uma eficiência de utilização do caminho de 100%? E se usar Selective-Repeat?

Garantimos eficiência de utilização de 100% caso nunca se esteja *idle* - isto é, estamos sempre a transmitir ou receber dados. Para tal, é necessário que a janela seja grande o suficiente para que, assim que o último pacote de um conjunto de N pacotes é enviado, começamos a receber o primeiro pacote desse mesmo conjunto.

Ora, em primeiro lugar será relevante calcular o RTT: neste caso, corresponde a duas vezes o atraso de propagação sobre todo o comprimento do caminho, ou seja, $2 \times 5000 \times 5 \times 10^{-6} = 0.05\ s$.

Ora, o nosso objetivo é o de encontrar um valor N_{window} tal que enviar N_{window} pacotes seja **igual** (já que queremos eficiência de utilização de 100%) ao tempo que leva a enviar um pacote e o respetivo ACK chegar. Neste sentido, o que vamos querer é o seguinte:

$$N_{window} \times \frac{L}{R} = \frac{L}{R} + RTT$$

Assim sendo, isto será o mesmo que ter o seguinte:

$$N_{window} = 1 + \frac{RTT}{\frac{L}{R}} = 1 + \frac{0.05}{\frac{1000}{100 \times 10^6}} = 1 + 5000 = 5001$$

A lógica é em tudo igual para o Selective-Repeat, já que o mecanismo aqui relevante é o de janela deslizante, não sendo relevante a ”tipologia” do mesmo.

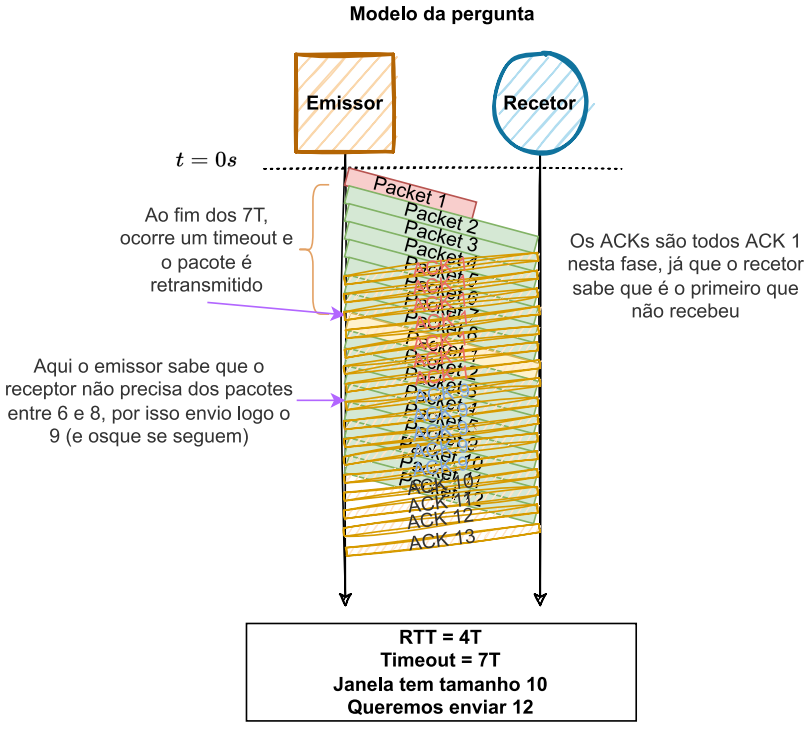
7. Considere o algoritmo de janela deslizante descrito a seguir que, não sendo nem Go-Back-N nem Selective-Repeat, tem características de ambos, e é um algoritmo muito próximo do empregue no TCP. Neste algoritmo, tanto o emissor quanto o receptor têm uma janela de dimensão N_w pacotes, $N_w > 1$, como no Selective-Repeat. No entanto, os ACKs são cumulativos como em Go-Back-N: isto é, se o recetor não recebeu o i-ésimo mas recebeu o j-ésimo pacote, com $j > i$, ele guarda-o na janela de receção, mas devolve um ACK i porque é do i-ésimo pacote que ele continua à espera. O emissor lança um temporizador por cada pacote enviado. Se este temporizador expira, o pacote associado, e apenas este, é imediatamente retransmitido.

(a) Assumindo que o canal pode perder os pacotes mas não os reordena, qual o módulo mínimo para a sua numeração que assegura a operação correta do protocolo?

Sabemos, por definição, que em algoritmos de janela deslizante o módulo mínimo para a numeração dos pacotes corresponde a $N_{emissor} + N_{receptor}$; neste sentido, temos que aqui o módulo mínimo é $2N_w$.

(b) Seja T o tempo de transmissão de um pacote. Assuma que: (i) a janela N_w tem dimensão 10 pacotes; (ii) o atraso de ida-e-volta é $4T$; (iii) e o tempo de espera para a retransmissão de um pacote é $7T$. O emissor tem 12 pacotes para transmitir. O primeiro destes pacotes, e só ele, é perdido. Faça um diagrama espaço-tempo que mostre a evolução do algoritmo até que todos os pacotes sejam recebidos com sucesso. Em face deste diagrama, conclua sobre o desempenho do algoritmo face aos algoritmos de Go-Back-N e Selective-Repeat, nestas circunstâncias.

Apresenta-se abaixo o diagrama espaço-tempo do algoritmo em questão.



As diferenças para os algoritmos Go-Back-N e Selective-Repeat são relativamente sucintas: no caso do

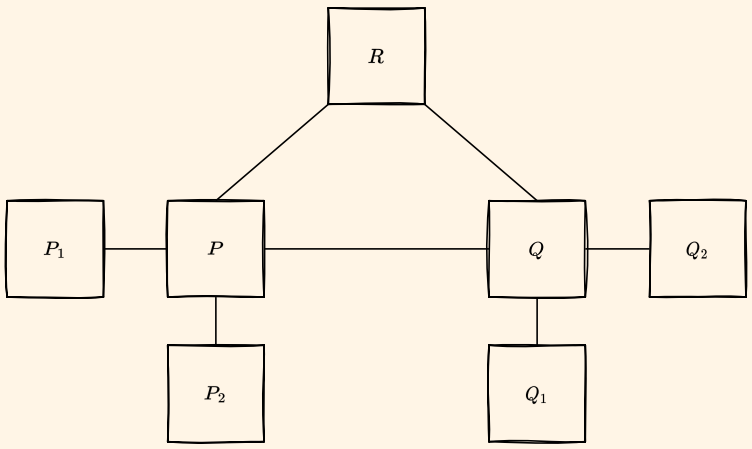
primeiro, toda a janela teria de ser retransmitida, o que implicaria a transmissão de 2 segmentos adicionais (7 e 8) - pior performance; com Selective Repeat, por outro lado, teria apenas de se retransmitir o primeiro segmento (já que os restantes ACKs seriam enviados com sucesso), o que levaria a melhor performance.

(c) Mostre, também através de diagramas espaço-tempo, uma circunstância em que este algoritmo tem melhor desempenho do que Selective-Repeat.

Caso haja perda de ACKs, o algoritmo em questão tem melhor desempenho do que Selective-Repeat. Considere-se o cenário em que são enviados 7 pacotes, e em que os primeiros 6 ACKs são perdidos, mas o último é recebido com sucesso.

- No caso do algoritmo proposto neste exercício, com ACKs cumulativos, o emissor sabe, ao receber ACK 8, que tudo até ao 7º pacote foi recebido com sucesso, pelo que pode continuar a enviar pacotes.
- No caso de Selective Repeat, o recetor envia ACK 7 (não ACK 8), e sem ter recebido nada sobre os anteriores terá de retransmitir todos os outros pacotes, o que levaria a uma performance inferior.

8. Considere os fornecedores de serviços Internet (ISPs) P, Q e R interligados da forma ilustrada na figura e em que nenhum deles deverá transportar tráfego de trânsito entre os outros dois. Aos ISPs P, Q e R foram atribuídos os blocos de endereços `193.32.0.0/11`, `193.64.0.0/11` e `193.0.0.0/12`, respetivamente. As regiões autónomas (ASes) P_1 e P_2 são clientes do ISP P e foram-lhes atribuídos os blocos de endereços `193.50.0.0/16` e `193.49.0.0/16`, respetivamente. As ASes Q_1 e Q_2 são clientes do ISP Q. A Q_1 foi atribuído o bloco de endereços `193.64.0.0/18`.



(a) Atribua um bloco de endereços a Q_2, sabendo que esta AS precisa de 8000 endereços e escolhendo o endereço base mais baixo possível.

Q_2 precisa de 8000 endereços - neste sentido, nunca será possível atribuir um bloco de endereços com sufixo superior a $/(32 - \lceil \log_2 8000 \rceil) = /19$. Mais ainda, foi atribuído a Q_1 o bloco `193.64.0.0/18`, pelo que todos os endereços no intervalo `193.64.0.0` a `193.64.63.255` estão também ocupados: em binário, é o equivalente a ter os endereços no seguinte intervalo ocupados:

[11000001.01000000.00000000.00000000, 11000001.01000000.00111111.11111111]

Assim sendo, o endereço mais baixo possível para Q_2 é 193.64.64.0, já que todos os anteriores estão ocupados. Alocando um bloco de endereços com $\lceil \log_2 8000 \rceil = 13$ bits, temos que o bloco a atribuir a Q_2 é 193.64.0.0/19:

[11000001.01000000.01000000.00000000, 11000001.01000000.01011111.11111111]

Note-se, claro, que nenhum destes endereços se encontra no bloco de endereços atribuído a Q_1.

(b) Apresente as tabelas de expedição de um encaminhador em P, outro em Q, e um terceiro em R, com pares (prefixo, AS).

	Subnet	Subnet mask	AS
1	193.50.0.0	255.255.0.0 (/16)	P_1
2	193.49.0.0	255.255.0.0 (/16)	P_2
3	193.64.0.0	255.224.0.0 (/11)	Q
4	193.0.0.0	255.240.0.0 (/12)	R
5	193.32.0.0	255.224.0.0 (/11)	P

Table 1: Tabela de encaminhamento para um encaminhador em P

	Subnet	Subnet mask	AS
1	193.64.0.0	255.255.192.0 (/18)	Q_1
2	193.64.64.0	255.255.224.0 (/19)	Q_2
3	193.32.0.0	255.224.0.0 (/11)	P
4	193.0.0.0	255.240.0.0 (/12)	R
5	193.64.0.0	255.224.0.0 (/11)	Q

Table 2: Tabela de encaminhamento para um encaminhador em Q

	Subnet	Subnet mask	AS
1	193.32.0.0	255.224.0.0 (/11)	P
2	193.64.0.0	255.224.0.0 (/11)	Q
3	193.0.0.0	255.240.0.0 (/12)	R

Table 3: Tabela de encaminhamento para um encaminhador em R

(c) Suponha que Q_1 mudou o seu ISP de Q para P. Apresente as novas tabelas de expedição tendo em conta que os endereços atribuídos a Q1 não se alteram.

As alterações às tabelas são mínimas: em P, é inserida a rota para Q_1; em Q a rota para Q_1 é alterada, com o AS da mesma a passar para P (em vez de Q_1 diretamente); é adicionada uma rota ao encaminhador em R para Q_1 (mas com referência ao AS P em vez de Q_1).

(d) Relativamente à alínea anterior, apresente em pseudocódigo um algoritmo para a expedição de datagramas usado num encaminhador do ISP R.

Seria algo como:

i. Se `IP_destino` & 255.224.0.0 = 193.32.0.0 OU `IP_destino` & 255.255.192.0 = 193.64.0.0, enviar para P.
ii. Caso contrário, se `IP_destino` & 255.224.0.0 = 193.64.0.0, enviar para Q.
iii. Caso contrário, se `IP_destino` & 255.240.0.0 = 193.0.0.0, enviar para R.

Note-se que não existe aqui (porque não é especificada no enunciado) a *default route*, com endereço a comparar e máscara ambos a zeros.

9. Suponha que um segmento TCP tem 2048 bytes de dados e 20 bytes de cabeçalho. Este segmento tem que atravessar duas ligações para chegar ao destino. A primeira ligação tem um MTU de 1024 bytes e a segunda um MTU de 512 bytes. Indique o comprimento e offset de todos os fragmentos entregues à camada de rede do destino. O cabeçalho de qualquer datagrama IP tem 20 bytes.

A resolução deste exercício tem algumas particularidades que o tornam menos trivial. O primeiro pensamento seria provavelmente dividir o segmento TCP (com 2068 bytes) em três datagramas IP, dois com 1004 + 20 bytes e um com $2 \times 20 + 20 + 20$ bytes. Ora, isto não é possível, já que juntamente com o datagrama IP é também enviado um offset ($\frac{\text{data length}}{8}$) inteiro. Neste caso, $\frac{1004}{8}$ não é um inteiro, pelo que seremos obrigados a enviar, no máximo 1000 bytes de dados. Assim sendo, de TCP para IP ficamos com três segmentos:

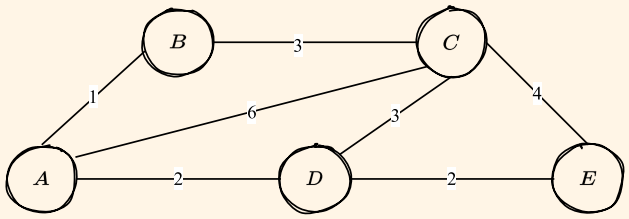
- Dois segmentos de tamanho 1020 bytes, com 1000 bytes de dados;
- Um segmento com os restantes 68 bytes de dados (isto inclui os 20 bytes do cabeçalho TCP), mais 20 bytes de cabeçalho. Note-se que, sendo o último elemento, a necessidade de data length divisível por 8 não se aplica.

Por fim, podemos então olhar para a segunda ligação IP. Como o MTU é de 512 bytes, em cada datagrama IP podemos enviar 492 bytes de dados. Mais uma vez, 492 não é divisível por 8 - na verdade, podemos enviar no máximo, por segmento, 488 bytes de dados. Assim sendo, temos:

	Datagram size	Data size	Offset (8-byte blocks)
1	508	488	0
2	508	488	61
3	44	24	122
4	508	488	125
5	508	488	186
6	44	24	247
7	88	68	250

Note-se que o offset vai sendo incrementando em $\frac{\text{data size}}{8}$ blocos por datagrama, claro. Mais ainda, notar que o último datagrama não teve necessidade de ser truncado, pelo que a sua estrutura manteve-se intacta.

10. Considere a rede da figura em que o protocolo de encaminhamento usado é por estado-da-ligação.



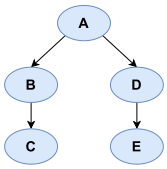
(a) Preencha uma tabela ilustrando a execução do algoritmo de Dijkstra a partir do nó A.

(A aplicação do algoritmo já foi abordada em ASA, pelo que não será descrita em pormenor nesta resposta.)

	Nós vistos	d_A, π_A	d_B, π_B	d_C, π_C	d_D, π_D	d_E, π_E
1	$\{A\}$	0, -	1, A	6, A	2, A	∞
2	$\{A, B\}$	0, -	1, A	4, B	2, A	∞
3	$\{A, B, D\}$	0, -	1, A	4, B	2, A	4, D
4	$\{A, B, D, C\}$	0, -	1, A	4, B	2, A	4, D
5	$\{A, B, D, C, E\}$	0, -	1, A	4, B	2, A	4, D

(b) Apresente o pseudocódigo genérico que cada nó tem que executar para a partir dos cálculos efetuados com o algoritmo de Dijkstra popular a sua tabela de expedição. Preencha a tabela de expedição do nó A.

O pseudocódigo é relativamente simples: para cada nó, após a aplicação de Dijkstra, temos o respetivo pai; basta então percorrer a árvore de caminhos mais curtos a partir do nó de destino, até ao nó de origem, por forma a obter o nó que a ele vai dar desde a origem - na tabela de expedição, o custo será o calculado pelo algoritmo de Dijkstra, e o nó de encaminhamento será o nó ligado à origem na árvore de caminhos mais curtos que liga o nó de destino ao nó de origem. Para o caso da alínea anterior, teríamos uma árvore como a que se segue:



A tabela de expedição em questão é a seguinte:

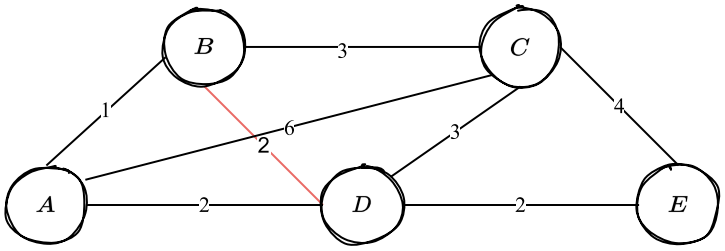
	Destino	Custo	Encaminhar p/
1	B	1	A
2	C	4	B
3	D	2	A
4	E	4	D
5	A	0	local

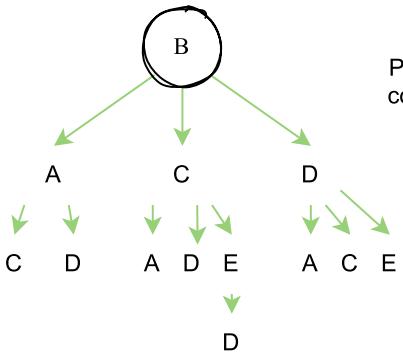
(c) Suponha que, devido a atrasos na difusão de um LSP, o nó A não tem conhecimento da ligação $D - E$, e só dela não tem conhecimento. Todos os outros nós têm conhecimento completo da topologia da rede. O que acontece aos pacotes enviados pelo nó A com destino ao nó E? Conclua sobre o regime transitório de um protocolo estado-da-ligação.

Na tabela, a quarta linha é alterada para o tuplo `(E, 8, C)`; os pacotes serão enviados para B, que sabe que o caminho mais curto até E é $BADE$, pelo que os pacotes serão encaminhados de volta a A. Como A não tem conhecimento da ligação $D - E$, os pacotes serão enviados novamente para B, ficando assim num *loop* infinito.

(d) Suponha que é estabelecida um nova ligação entre os nós D e B com comprimento 2. No total, quantos LSPs é que vão viajar pela rede?

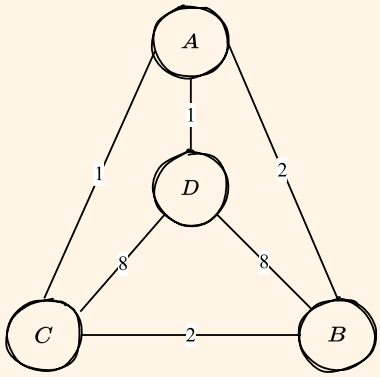
B e D apercebem-se da alteração, claro, e serão eles a fazer o primeiro *broadcast* dos LSPs. B informa A, C, e D, que por sua vez informam os seus filhos, etc. Abaixo encontra-se uma ilustração do processo:





Podemos construir uma árvore análoga com raiz em D, ficando assim com 2*12 = 24 LSPs enviados pela rede

11. Considere a rede da figura sobre a qual opera um protocolo vetor-distância.



(a) No instante t_0 o protocolo está estável, com todos os nós a saber os comprimentos dos caminhos mais curtos para alcançar cada um dos outros nós. Apresente as entradas das tabelas de encaminhamento no que diz respeito ao nó destino D.

No instante inicial:

- A sabe que o caminho mais curto para D é direto, com custo 1.
- B sabe que o caminho mais curto para D é $B - A - D$, com custo 3.
- C sabe que o caminho mais curto para D é $C - A - D$, com custo 2.
- D para D é direto, com custo 0.

(b) No instante t_1 a ligação $A - D$ falha. Assumindo que os nós trocam mensagens sincronamente em instantes bem definidos, $t_1, t_2, t_3, ...$, mostre a evolução das tabelas de encaminhamento até que o protocolo volte a estabilizar.

Inicialmente, todas as tabelas de custo tinham o seguinte aspeto (para todos os nós):

From/Cost to	A	B	C	D
A	0	2	1	1
B	2	0	2	3
C	1	2	0	2
D	1	3	2	0

Após a ligação $A - D$ falhar, no instante t_1, as tabelas de A e D são atualizadas:

From/Cost to	A	B	C	D
A	0	2	1	$\min\{2+3, 1+2\} = 3$
B	2	0	2	3
C	1	2	0	2
D	1	3	2	$\min\{3+2, 2+1\} = 3$

…(acabo depois)

(c) Repita a alínea anterior, mas assumindo agora que o protocolo utilizado é vetor-caminho. Neste protocolo os nós trocam entre si não apenas a distância para alcançar cada destino mas também todo o caminho (sequência de nós) associado a essa distância.

12. Pretende-se comparar duas abordagens para providenciar multicast na camada de aplicação: (1) emulação com encaminhamento unicast; (2) encaminhamento multicast. Considere uma fonte e 32 destinos. A fonte está interligada com os destinos através de uma árvore binária de encaminhadores, tendo a fonte com raiz. O custo de uma abordagem multicast é o número de datagramas que têm que ser enviados nessa abordagem por forma a alcançar todos os destinos.

(a) Determine os custos das duas abordagens para multicast.

Considerando uma abordagem via árvore binária, com 32 destinos possíveis, vamos ter, claro, $\log_2 32 = 5$ níveis de encaminhadores; neste sentido:

- Para a primeira abordagem, emulação com encaminhamento unicast, o custo corresponderá ao produto entre o número de níveis na árvore e o número de destinos possíveis, já que cada mensagem irá seguir um caminho específico por destino; existirá um datagrama por cada vez que um encaminhador *encaminha* um datagrama para um destino;
- Para a segunda abordagem, encaminhamento multicast, o custo corresponderá ao número de arestas na árvore, já que cada encaminhador irá fazer *broadcast* de um datagrama para todos os seus filhos.

Assim sendo, podemos dizer:

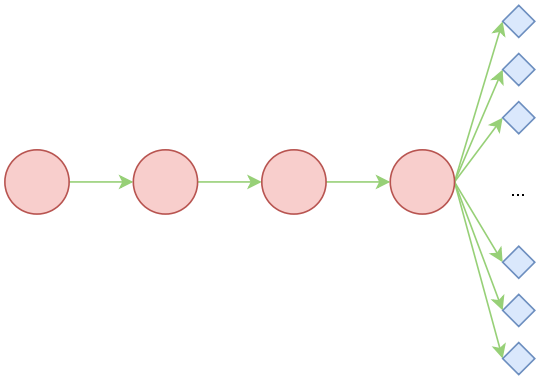
- Primeira abordagem: $D \times \log_2 D = 32 \times 5 = 160$;
- Segunda abordagem: $\sum_{i=1}^{\log_2 D} 2^i = 62$.

(b) Encontre a topologia de rede, incluindo a fonte, os destinos, e tantos encaminhadores quanto quiser, que maximiza a razão entre o custo da abordagem por emulação com encaminhamento unicast e o custo da abordagem por encaminhamento multicast na camada.

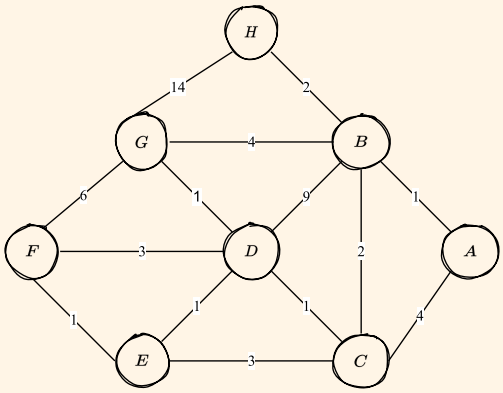
Queremos, portanto, maximizar $\frac{\text{custo unicast}}{\text{custo multicast}}$. Note-se que abordagens multicast são mais eficientes em topologias completamente esguias até ao último nível, já que não terá de haver divisão de datagramas por vários caminhos; por outro lado, e apesar de também ser mais eficiente do que abordagens binárias no caso de unicast, o caminho terá também de ser percorrido 1 vez por destino. Assim sendo, o cenário que maximiza o rácio pretendido é o seguinte:

Em abordagens por emulação com encaminhamento unicast, o custo será, considerando N níveis, $N \times D$; por outro lado, com encaminhamento multicast, o custo será $(N-1)+D$. Considerando $D = 32$, ou seja, introduzindo o contexto do problema, a razão pretendida tenderá então para 32:

$$\frac{32N}{N-1+32} \to 32$$

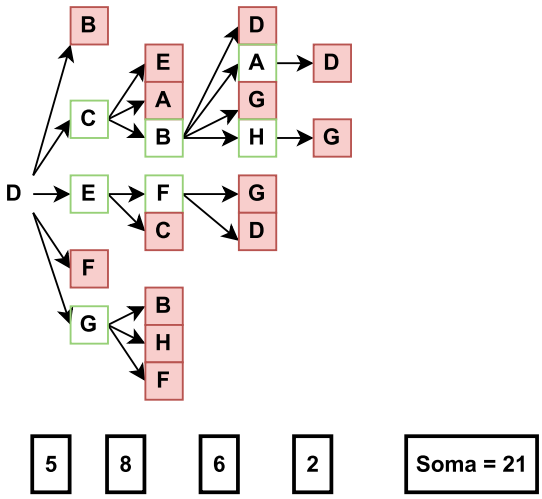


13. Considere a rede da figura. O nó D envia um datagrama que é difundido por toda a rede usando expedição por caminho inverso (RPF). Indique quantas cópias do datagrama é que atravessam as ligações da rede.



A lógica do algoritmo em questão é relativamente simples: corresponde a uma ”variação” do algoritmo de Dijkstra, em que sempre que olhamos para um novo nó, fazemos *broadcast* do datagrama para todos os seus vizinhos (exceto o pai).

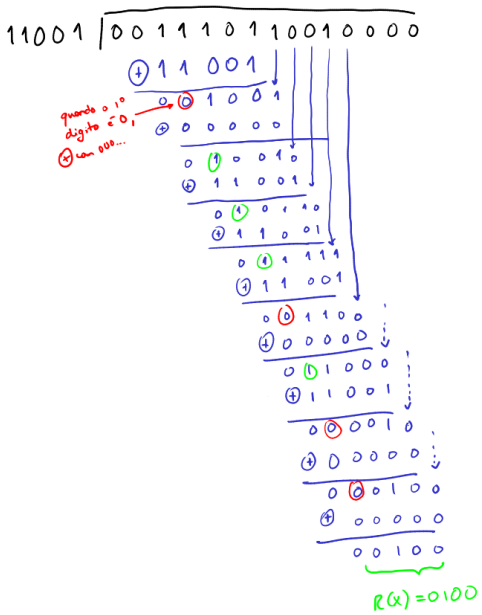
	Nós Vistos	d_A, π_A	d_B, π_B	d_C, π_C	d_D, π_D	d_E, π_E	d_F, π_F	d_G, π_G	d_H, π_H
1	$\{D\}$	∞	9	1	0	1	3	1	∞
2	$\{D, C\}$	5	3	1	0	1	3	1	∞
3	$\{D, C, E\}$	5	3	1	0	1	2	1	∞
4	$\{D, C, E, G\}$	5	3	1	0	1	2	1	15
5	$\{D, C, E, G, F\}$	5	3	1	0	1	2	1	15
6	$\{D, C, E, G, F, B\}$	4	3	1	0	1	2	1	5
7	$\{D, C, E, G, F, B, A\}$	4	3	1	0	1	2	1	5
8	$\{D, C, E, G, F, B, A, H\}$	4	3	1	0	1	2	1	5



14. Suponha que um determinado protocolo da camada da ligação de dados usa um código cíclico de verificação, CRC, gerado pelo polinómio gerador $G(x) = x^4 + x^3 + 1$.

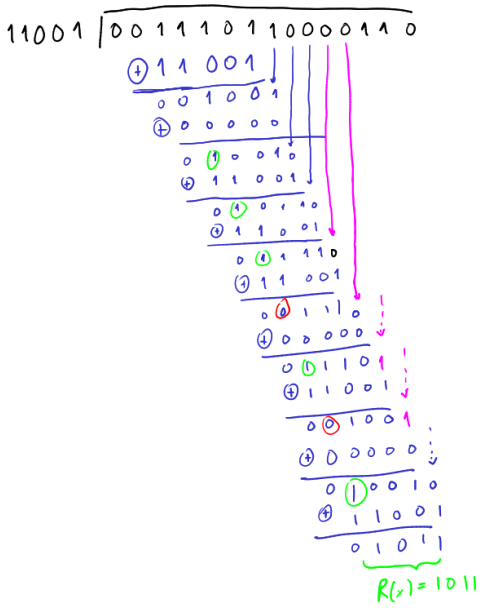
(a) Determine os bits de CRC do bloco de dados 00111011001.

Com um polinómio gerador do tipo $x^4 + x^3 + 1$, sabemos que o divisor pretendido será, em binário, 11001. O dividendo, 00111011001, vê concatenados quatro zeros à direita (são sempre concatenados $N - 1$ zeros, onde N é o número de bits do divisor). A divisão será feita por subtrações sucessivas, e o resultado será o resto da divisão. A divisão é feita da seguinte forma:



(b) Suponha que o emissor forma uma trama com o bloco de dados e os bits de CRC da alínea anterior. A trama é enviada do emissor para o recetor, é corrompida no canal que os une, e é recebida pelo recetor na forma 001110110000110. Os erros são detetados no recetor?

Um erro será detetado na mensagem caso o que seja recebido não for divisível pelo polinómio gerador.



Podemos, assim, concluir que o erro é detetado, já que $R(x) \neq 0$.

15. Suponha duas estações, A e B, ligadas a extremos opostos de um cabo de $900m$, e cada uma delas com uma trama de 1000 bits para transmitir. As duas estações começam a transmitir em $t = 0$. Suponha que há 4 repetidores entre A e B, cada um inserindo um atraso de 20 bits. Considere que a taxa de transmissão é $10Mbit/s$ e que se utiliza CSMA/CD com intervalos de retransmissão múltiplos de 512 bits. Depois da primeira colisão, A escolhe $K_A = 0$ e B escolhe $K_B = 1$ na execução do algoritmo de recuo binário exponencial. Ignore o sinal de reforço de colisão de 32 bits e a espera de 96 bits para o acesso ao canal.

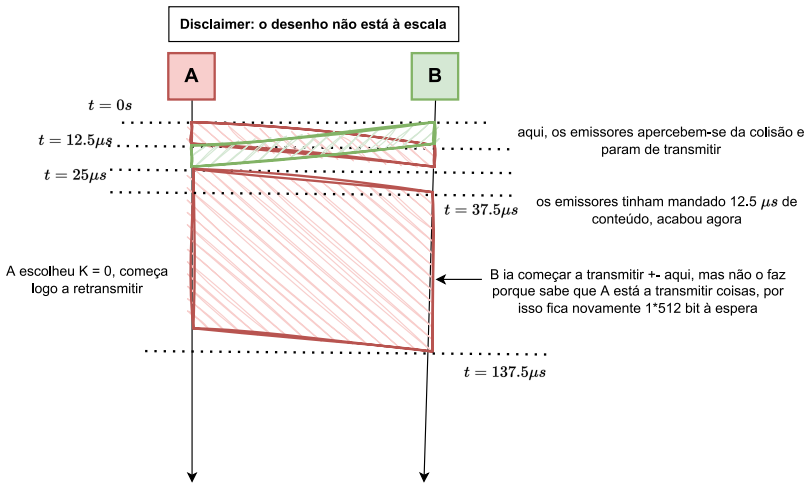
(a) Qual o atraso de propagação, incluindo atrasos introduzidos nos repetidores de A para B? Suponha uma velocidade de propagação no cabo de $2 \times 10^8 m/s$.

O atraso de propagação pretendido corresponderá à soma entre o tempo de propagação normal, sem repetidores, e o tempo de propagação introduzido pelos mesmos.

No cabo, a velocidade de propagação é de $2 \times 10^8 m/s$, pelo que, percorrendo $900m$, o tempo de propagação será de $4.5 \times 10^{-6} s$. Mais ainda, é introduzido um atraso de 20 bits em cada repetidor - podendo transmitir $10Mbit/s = 10^7 bit/s$, cada repetidor introduzirá, assim, um atraso de $2 \times 10^{-6} s$. Assim sendo, no total, o atraso de propagação será de $4.5 \times 10^{-6} s + 4 \times 2 \times 10^{-6} s = 1.25 \times 10^{-5} s$.

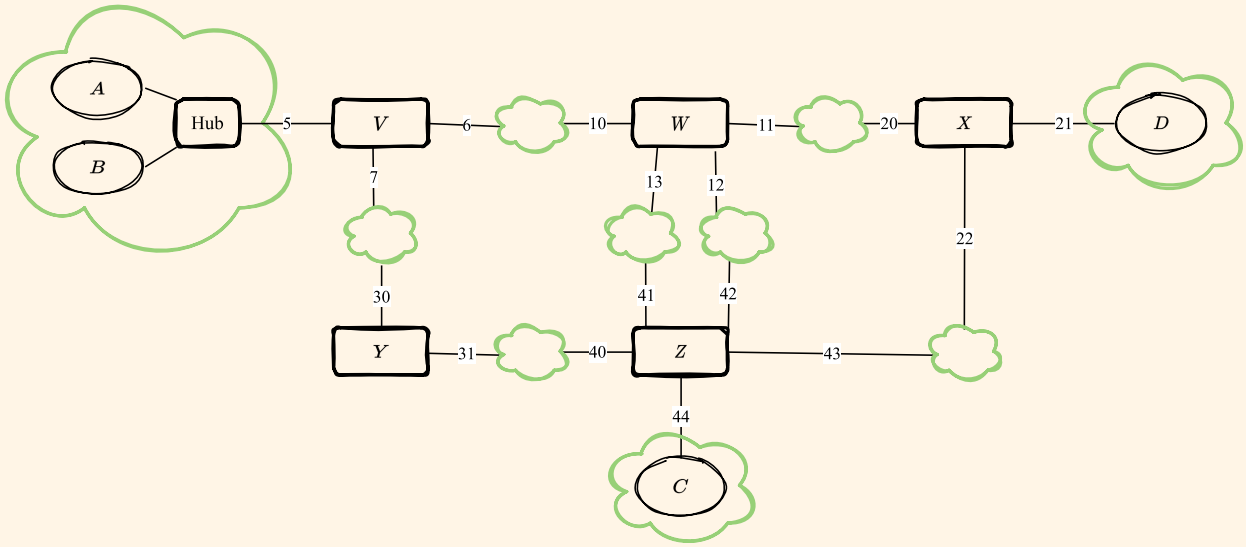
(b) Em que instante de tempo é que o pacote de A é completamente recebido em B?

O diagrama espaço-tempo da interação em questão é o que se segue:



Passados $1.25 \times 10^{-6}s$ desde o início da transmissão, A e B apercebem-se da colisão, pelo que param de transmitir - a transmissão em si só chega completamente ao outro lado $1.25 \times 10^{-6}s$ depois, pelo que só aqui já se passaram $2.5 \times 10^{-6}s$ desde o início da transmissão. De seguida, com A a escolher $K = 0$, pode logo voltar a transmitir; por outro lado, B tem de esperar 512 bits, o que corresponde a $5.12 \times 10^{-5}s$. Note-se que a transmissão de 1000 bits leva $100 \times 10^{-6}s$, pelo que B até tenta começar a transmitir sensivelmente a meio da transmissão de A, mas apercebendo-se que este está a transmitir, não o faz, voltando a tentar depois de $5.12 \times 10^{-5}s$ (e aqui a transmissão de A já terminou). Ao todo, passaram $3 \times 12.5 \times 10^{-6} + 100 \times 10^{-6} = 137.5\mu s$ desde o início da transmissão, até ao pacote de A ser completamente recebido em B.

16. Considere a rede da figura, composta por cinco comutadores Ethernet (Bridges): V, W, X, Y, Z, um hub e quatro estações: A, B, C, D. Considera-se que o identificador de cada comutador é igual ao menor dos endereços MAC das suas interfaces (indicados simplificadamente junto a cada ligação) - por exemplo, o identificador do comutador W será 10. Todas as ligações têm custo unitário. As tabelas estão inicialmente vazias.



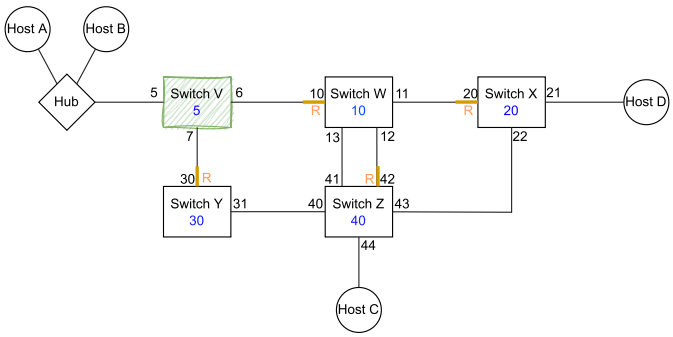
Disclaimer: resolução da autoria do Pedro Chaparro, retirada daqui.

(a) Usando o algoritmo spanning tree classifique as interfaces de cada um dos comutadores em raiz, designada, ou bloqueada, e indique as BPDUs enviadas por cada comutador em cada uma das suas interfaces quando em regime estacionário.

Em primeiro lugar, é relevante realçar que o comutador raiz será, claro, V, pois é o comutador com menor identificador (5). Igualmente relevante é recordar que o formato das BPDU é o seguinte:

`(Root Bridge ID, Root Path Cost, Bridge ID de quem enviou, Port ID de quem enviou)`

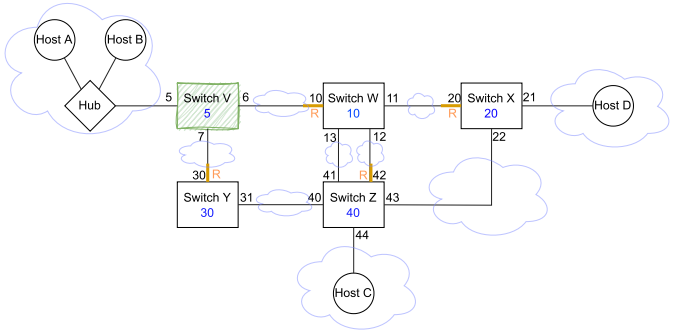
De seguida, vamos procurar calcular as *root ports* em cada comutador, que representam os portos pelo qual se consegue, com menor custo, chegar à raiz. Tanto as interfaces vizinhas do comutador como as em que não há empate são triviais, estando representadas na figura abaixo:



Note-se, contudo, que há cenários em que há empate no custo. Nestes casos, escolhe-se a interface com menores valores nos dois outros identificadores da BPDU (sendo estes os que têm maior prioridade). Quanto ao que à imagem acima diz respeito, só nos resta determinar a root port em Z: para tal, examinam-se as BPDUs como supra-indicado:

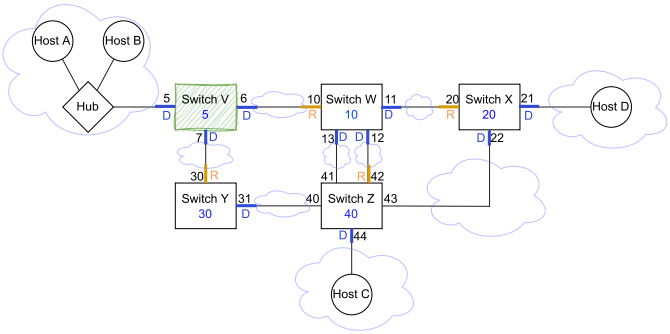
$$X \to Z : (5, 2, 20, 22), \quad Y \to Z : (5, 1, 30, 31), \quad W \to Z : (5, 1, 10, 12), \quad W \to Z : (5, 1, 10, 13)$$

X é descartado por ter um custo maior, Y por ter um identificador de bridge maior, e por fim escolhemos o porto 12 de W por ter um identificador de porto menor. Assim sendo, temos atualmente as *root ports* representadas na figura abaixo:

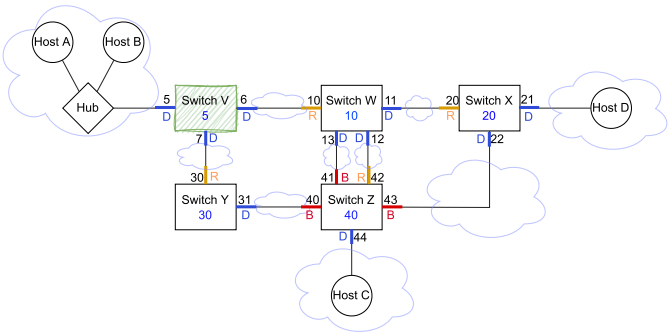


De seguida, vamos descobrir as *designated ports*: para segmento de rede (as *nuvens* na figura), qual é o caminho mais barato para a raiz? A análise é feita de forma em tudo análoga à anterior, analisando as

BPDUs, e ficamos com as seguintes *designated ports* (note-se que todos os portos do comutador raiz são, por definição, *designated ports*, a não ser que não estejam ligados a nenhum segmento de rede):

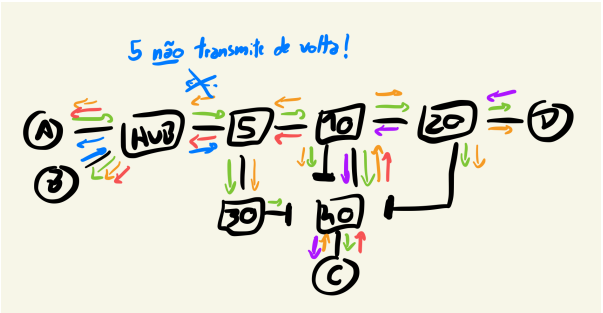


Todos os restantes portos ficam bloqueados, pelo que ficamos com a rede completamente classificada:



(b) Para a sequência de envio de tramas: ($A \rightarrow B$, $C \rightarrow D$, $C \rightarrow A$, $B \rightarrow A$, $D \rightarrow C$), indique as interfaces sobre as quais são transmitidas cópias das tramas respectivas e qual o estado das tabelas de expedição de cada comutador no final das várias transmissões.

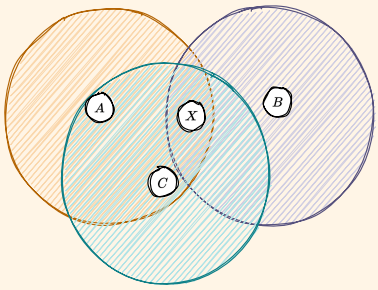
Inicialmente, as tabelas em todos os comutadores estão vazias. Vamos procurar preenchê-las, com o enviar de cada trama, com pares (`host de origem`, `interface pela qual recebeu`). A tabela abaixo, a par do desenho, mostra precisamente o *flooding* da rede aquando do envio de cada trama - note-se a particularidade do hub, que ao contrário dos outros comutadores, faz *broadcast* para todos os vizinhos, inclusive quem lhe enviou a trama.



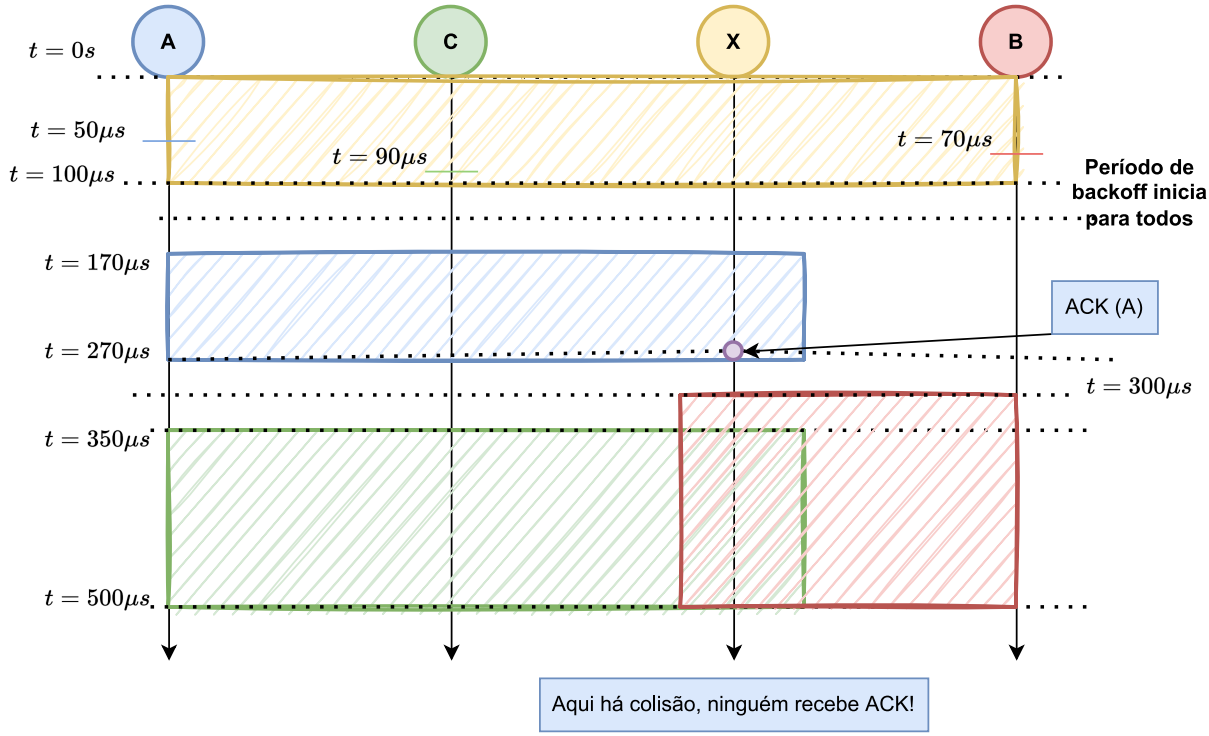
	V(5)	W(10)	X(20)	Y(30)	Z(40)
A → B	A, 5	A, 10	A, 20	A, 30	A, 42
C → D	C, 6	C, 12	C, 20	C, 30	C, 44
C → A	C, 6	C, 12	-	-	C, 44
B → A	B, 5	-	-	-	-
D → C	-	D, 11	D, 21	-	D, 42

17. O diagrama da figura mostra uma rede Wi-Fi em que X é o ponto de acesso e A, B e C são estações a ele associadas. Os círculos grandes centrados em cada uma das estações representam a sua área de cobertura, isto é, o alcance do seu sinal eletromagnético. A área de cobertura do ponto de acesso X não está representada, mas subentende-se que cobre as três estações. O protocolo de acesso ao meio é CSMA/CA. Suponha que X está a transmitir uma trama no instante 0 μs que acabará de ser transmitida no instante $100\mu s$. Suponha ainda o seguinte:

Estação	Instante em que tem trama p/ transmitir (μs)	Duração de transmissão da trama (μs)	Tempo de recuo (backoff) (μs)
A	50	100	70
B	70	200	200
C	90	150	150



O seguinte diagrama espaço-tempo será útil para a resolução das alíneas que não contemplam CSMA/CA com RTS-CTS:



Tal como se pode visualizar no diagrama acima, enquanto X tenta transmitir, todas as estações tentam transmitir (mas não conseguem, porque todas sabem que X está também a transmitir). Neste sentido, após essa transmissão, todas as estações entram num período de *back-off*, sendo que a primeira a sair dele é a estação A, que começa a transmitir a sua trama no instante $170\mu s$. Ora, B não consegue "ouvir" A, pelo que não interrompe o seu período de back-off (ao contrário de C, que interrompe passados $70\mu s$, restando $80\mu s$ para o seu período terminar). Assim, quando A termina de transmitir, faltam $30\mu s$ para B terminar o seu período de back-off, e $80\mu s$ para C terminar o seu. B começa a transmitir a sua trama aos $300\mu s$, e C aos $350\mu s$, entrando em colisão (já que nenhum dos dois ouve o outro). No fim, X não faz *acknowledgement* de nenhuma das tramas, pelo que só agora é que as estações se apercebem da colisão.

(a) Para cada uma das três estações, em que instante de tempo é que ela começa a transmitir a sua trama pela primeira vez? Despreze atrasos de propagação, intervalos-entre-tramas (inter-frame spacings) e tempos de transmissão dos ACK.

Tal como referido na explicação acima:

- A: $170\mu s$
- B: $300\mu s$
- C: $350\mu s$

(b) Das três tramas transmitidas pelas estações quais é que são bem recebidas na primeira tentativa de transmissão?

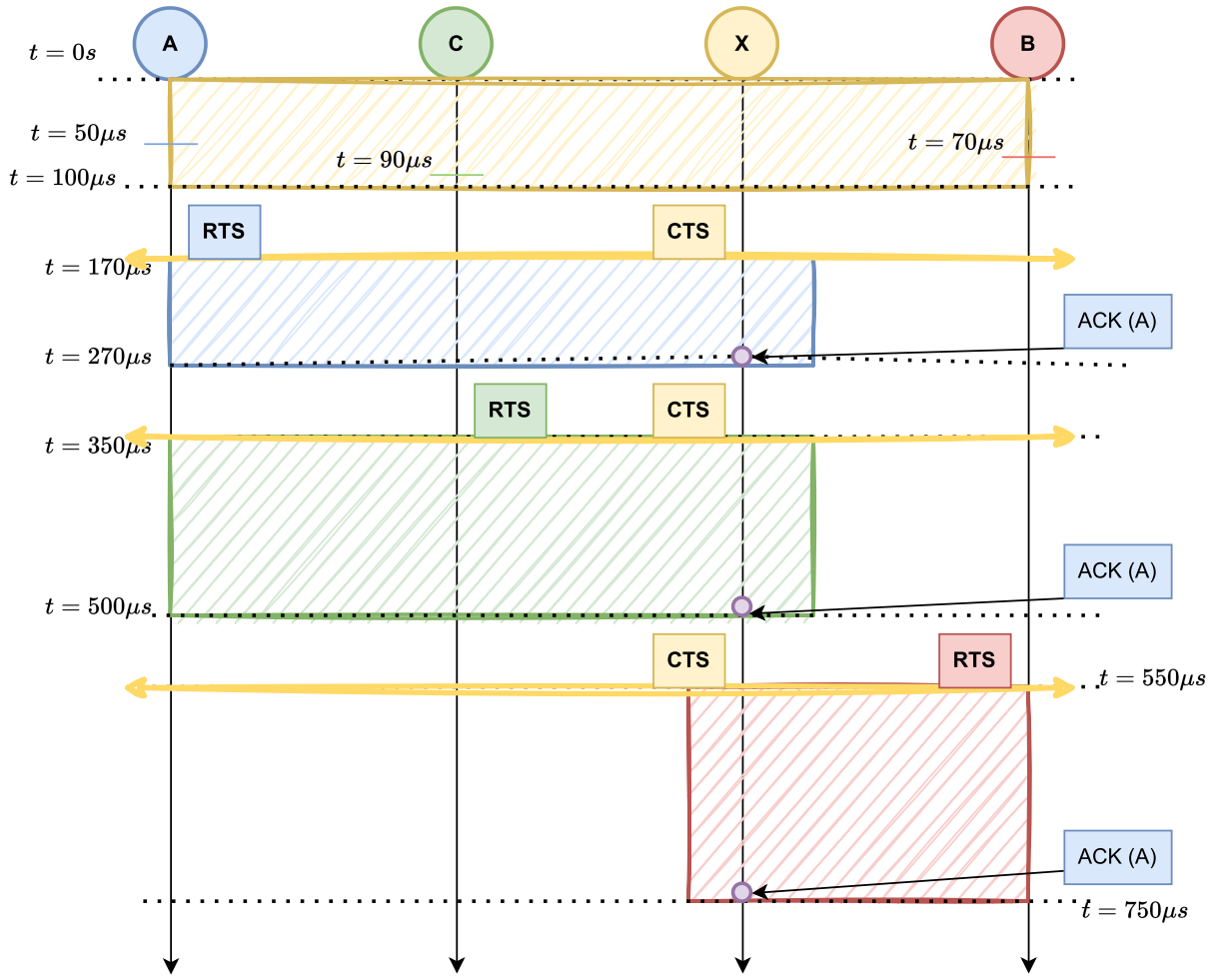
Apenas a trama de A é bem recebida na primeira tentativa de transmissão, tendo o respetivo ACK sido enviado por X no instante $270\mu s$.

(c) Para cada uma das tramas que não é bem recebida na primeira tentativa de transmissão, indique qual o instante de tempo em que a estação correspondente se apercebe de que a trama foi perdida. As tramas serão bem recebidas na segunda tentativa de transmissão?

Ambas as estações apercebem-se no mesmo instante, $500\mu s$. Nenhuma delas conseguirá transmitir sem colisão na segunda tentativa, já que B voltará a transmitir enquanto C transmite.

(d) Suponha agora que se ativa o protocolo de acesso ao meio RTS-CTS. Neste caso, em que instante de tempo é que cada estação começa a transmitir a sua trama pela primeira vez? As tramas são todas recebidas com sucesso?

O diagrama espaço-tempo para esta interação é o seguinte:



Note-se que ao utilizar este protocolo de acesso ao meio, não há obrigatoriedade das várias estações se ouvirem, já que todas esperam pela concessão de acesso por parte de X. Temos, claro, que agora todas tramas são recebidas com sucesso, nos instantes indicados na figura.