COMP3331/93331: Solutions to Mid-session Exam

Question 1 Hodge-Podge (6 marks: 1 mark for each question)

Write down your answer and a short explanation (where relevant) in the space provided in the exam paper below.

(a) Assume that 10 clients (each running on a separate machine) are simultaneously communicating with a web server using HTTP1.1 and requesting 5 objects each (do not consider a separate index page). Assume that the web server has sufficient resources to service all received requests simultaneously. How many sockets are simultaneously open on each client machine and on the web server?

Answer and short explanation:

Since the clients are using HTTP1.1 (persistence with pipelining), all objects will be requested and transmitted over a single TCP connection between each client and the server.

Thus, each client will have one TCP socket open.

The server will service all 10 clients simultaneously. Thus the server will have 10 sockets open, 1 socket to communicate with each client. In addition, the server will also have a welcome socket open. Thus in total, there will be 11 sockets open at the server.

(b) For the question above, now assume that the 10 clients are using HTTP1.0 with parallel connections and are requesting 5 objects each (do not consider a separate index page). Assume that each client can open as many parallel connections as required and that the web server has sufficient resources to service all received requests simultaneously. How many sockets are simultaneously open on each client machine and on the web server?

Answer and short explanation:

Since the clients are using HTTP1.0 (non-persistent HTTP), each object will be requested and transmitted over a distinct TCP connection between each client and the server.

Thus, each client will have five TCP sockets open.

The server will service all 10 clients simultaneously. Thus, the server will now have 50 sockets open (10 clients x 5 sockets each). In addition, the server will also have a welcome socket open. Thus in total, there will be 51 sockets open at the server.

(c) The author of BitTorrent claims "incentives build robustness in BitTorrent." In other words, the tit-for-tat strategy ensures that all peers cooperate in the file distribution process. However, a peer can still cheat in BitTorrent, i.e., download the entire file without uploading any chunks to any other peers. How is this possible?

Answer and short explanation:

A selfish peer could download only from the seed (which has the entire file and thus does not need any chunks) or through the "optimistic unchoking" process when other peers give this peer a chance to download some chunks from them.

(d) Which of the following applications typically use UDP? (Circle ALL that are correct):

1. **DNS**
2. Bittorrent (for file transfer)
3. E-mail
4. **First person shooter games**
5. HTTP
6. FTP

(e) Which of the following is/are true about web caches? (circle ALL that are correct)

1. **A web cache is a network entity that satisfies HTTP requests from clients.**
2. **A web cache is both a client and a server at the same time.**
3. HTTP does not explicitly support caching or cache consistency.
4. All HTTP objects are cacheable.

(f) Using TCP, a sender has sent out a total of ten data segments, each containing 1000 bytes of payload. Assume that the sender's initial sequence number (ISN) is 5000. After the sender receives a packet from the receiver with ACK number 8000, how many TCP data segments (denote their sequence numbers), if any does the sender know that the receiver definitely received?

Answer and short explanation:

Since TCP uses cumulative acks, the receipt of ACK number 8000 guarantees that everything up to byte # 7999 has been correctly received by the receiver. Thus, the sender can be sure that the first 3 packets (#5000, 6000 and 7000) have been correctly received.

NOTE: In practice, the sequence #s would be 5001, 6001 and 7001 since the SYN packet uses up 1 sequence number. However, we will ignore this here.

Question 2 – The Poisoning of Google (3 marks)

Alice works at a search engine startup called Searchzilla (www.searchzilla.com) whose main competitor is Google (www.google.com). She would like to crush her competitor in the "non-traditional" way by messing up with DNS servers. Recalling from her COMP3331/9331 class that DNS servers cache A and NS records from DNS replies, Alice realises she can configure the authoritative DNS server of Searchzilla (nspowned.searchzilla.com) to return incorrect results for arbitrary domains. If other DNS servers caches Alice's malicious results, they will return bad results. Help Alice complete her master plan to hijack Google's domain name by writing down exactly what Alice's name server returns upon receiving a DNS query. To be precise, what Alice wants to achieve is that when anyone on the Internet types www.google.com in their browser, they should be presented with the Searchzilla webpage rather than Google's. Assume that the Searchzilla web server (www.searchzilla.com) is running on 9.9.9.9 and the authoritative name server for Searchzilla, nspowned.searchzilla.com is running on 9.9.9.10. Recall that DNS records are of the format <name, value, type, ttl>.

(a) Can you precisely outline the steps that Alice would have to undertake to launch the aforementioned attack? You must explicitly provide the DNS records that she will have to configure and explicitly state which servers these records must be stored/updated in.

Answer:

Alice would configure the following entries in her authoritative name server (i.e. in 9.9.9.10)

www.searchzilla.com 9.9.9.9 A long-TTL
www.searchzilla.com ns1.google.com NS long-TTL
ns1.google.com 9.9.9.10 A long-TTL
www.google.com 9.9.9.9 A long-TTL

The first record is the standard A record for searchzilla. This is not specific to the attack.
The second record is a malicious entry, which suggests that the name server for searchzilla is ns1.google.com, which happens to be name server for google.com.
The third record is the hostname to IP address mapping for the google name server. However, note that this is a malicious entry as it incorrectly maps the google name server to the authoritative nameserver for searchzilla (9.9.9.10).
The last record maliciously maps the google.com hostname to the searchzilla web server.

Alice would encourage users on the Internet to visit her website, www.searchzilla.com

When Alice's name server receives a type A DNS query for www.searchzilla.com, it will return the 2nd and 3rd record (from above) in the reply. The 2nd record would be included in the authority section of the reply and the 3rd record will be included in the additional section.

If DNS servers in the Internet blindly cache these records, then future queries for www.google.com will be directed to Alice's authoritative name server, 9.9.9.9 (as a result of the 3rd record). Alice's name server will reply back with the 4th record to such queries. As a result, the user querying for www.google.com will instead be presented with the searchzilla home page. To be precise, the GET request for the google home page will be sent to the searchzilla web server, which will reply back with the searchzilla home page.

(b) What must a robust DNS server implementation do to counter this attack?

Answer:

A robust DNS server implementation should be less trustful of results returned by other DNS servers and only cache information that's directly relevant to the queried domain and which is coming from the authoritative name servers. In the above example, since google.com is not a subdomain of searchzilla.com, a correct DNS server implementation should ignore all information related to google.com in the results, i.e. the 2nd and 3rd records would no be cached.

Another option is to use something like DNSSec, which allows the query response to be authenticated.

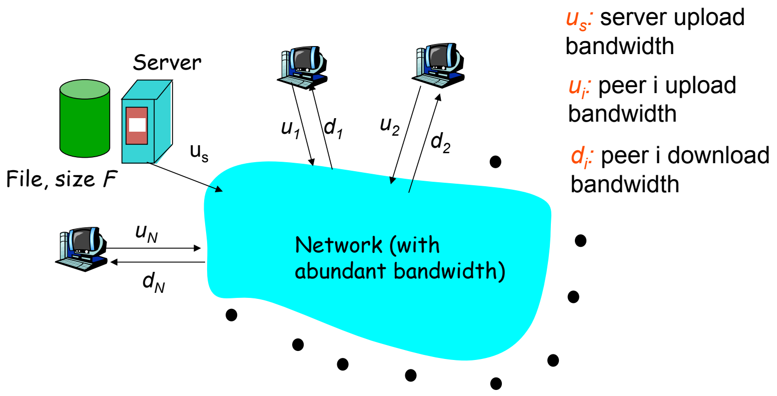
Question 3 - Oh why so selfish? (2 marks)

Consider the peer-to-peer file distribution problem illustrated in the Figure below. The same problem has been discussed in the lecture slides as well as in the text. In this problem, there are N peers, and peer i has an upload transmission rate of u_i bps and a download transmission rate of d_i bps. The upload rate for the server is u_s bps. The text (and lecture slides) shows that a lower bound for the minimum time for all the N peers to download a specific file of size F bits, using peer-to-peer distribution, is:

$$D_{P2P} \;\geq\; \max\left\{\frac{F}{u_s}, \frac{F}{d_{\min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\}$$

where $d_{\min} = \min_{i=1,2,\ldots,N} d_i$.

Note that the above expression has been derived based on the assumption that every peer is willing to upload the file (or portions of the file) after it has received it.


u_s: server upload bandwidth
u_i: peer i upload bandwidth
d_i: peer i download bandwidth
Server
File, size F
u_s
u_1
d_1
u_2
d_2
u_N
d_N
Network (with abundant bandwidth)

Now, let us assume that peers *1,2,…., k* (where *k* is a positive integer strictly less than *N*) are selfish peers which will only download files from others but do not upload any portion of the file that they have received. Derive an expression for the lower bound of the minimum time for all *N* peers to download the file under this revised assumption. Using the expression that you have derived, explain whether the delay when there are selfish users will be bigger or smaller than the case when all users participate in uploading.

Hint: The lower bound for the case with selfish peers is of the form

$$D_{P2P}^{\text{selfish}} \;\geq\; \max\left\{\frac{F}{u_s}, \frac{F}{d_{\min}}, x\right\}$$

and you will need to find what *x* is.

Answer:

In order for all the peers to receive the file, the file must have been uploaded N times by the server and the non-selfish peers. The total number of bits that are uploaded is NF. The total bandwidth for uploading is $u_s + \sum_{i=k+1}^{N} u_i$ because only the server and the non-selfish peer perform upload. A lower bound for the download time is: $\frac{NF}{u_s+\sum_{i=k+1}^{N} u_i}$.

Therefore, we have

$$D_{\text{P2P}}^{\text{selfish users}} \geq \max\left\{\frac{F}{u_s}, \frac{F}{d_{\min}}, \frac{NF}{u_s + \sum_{i=k+1}^{N} u_i}\right\} \tag{2}$$

Given that $u_s + \sum_{i=1}^{N} u_i > u_s + \sum_{i=k+1}^{N} u_i$, we have $\frac{NF}{u_s+\sum_{i=k+1}^{N} u_i} > \frac{NF}{u_s+\sum_{i=1}^{N} u_i}$, the delay when there are selfish users will be no less than when all users upload.

Remark: The aim of this question is to test whether you understand how the expression $\frac{NF}{u_s+\sum_{i=1}^{N} u_i}$ is derived. If you do, you will be able to adapt the original solution to the new situation suggested in the question.

Question 4 – Sliding Windows (5 marks)

Suppose that a sender and receiver are connected via a point-to-point link that has 1 Mbps bandwidth and a one-way propagation delay of 4.5 ms. Assume that the sender always has data for transmission and that the size of each data packet is 125 Bytes. Neglect any headers. Also assume that the size of Ack packets is negligible. Answer each of the following for both go-back-n and selective-repeat sliding window schemes. You must explain your reasoning in the space provided.

(a) Assuming that the link is error-free, what should be the size of the window (in terms of number of packets) to achieve a throughput of 0.8 Mbps.

Go-back-n:

The time to transmit one data packet, T = 125 x 8 bits /1Mbps = 1ms

RTT = 2 x one-way propagation delay = 9ms

As discussed in the Week 5 lecture slide # 6, the utilization of the link (channel) when a sliding window protocol with window size N, is given by,

U = NT/(T + RTT)

Achieving a throughput of 0.8Mbps on a 1Mbps link implies that the utilisation should 0.8.

Thus, 0.8 = N x 1ms /1ms + 9ms. This results in N = 8 packets.

Selective repeat:

Over an error-free link, GBN and SR achieve exactly the same performance, so the above answer also applies to SR.

(b) What is the minimum number of bits needed to represent the sequence numbers corresponding to the above window size? Recall that an *n* bit sequence number results in a range of sequence numbers from *0* to 2^n-1.

Go-back-n:

To achieve a window size of 8 with GBN, we will need 9 unique sequence numbers (1 more than the window size). Thus, it is necessary to have a 4-bit sequence number space in GBN.

Selective repeat:

As was discussed on slide 16, Week 5, with SR, the window size must be less than or equal to half the size of the sequence number space. Thus, in order to accommodate a window size of 8, we need at least 16 unique numbers, which implies that SR would require a 4-bit sequence number space.

(c) Suppose that the sender is transmitting a large number of packets to the receiver and there are always packets waiting to be transmitted. Suppose the 4th and 12th packet are lost during transmission. No other packets or acks are lost. Assume that sequence numbers start at 0. Assuming the window size and the minimum sequence number space that you have computed in the previous two parts, how many packets get retransmitted and what are their sequence numbers? Explain. (**NOTE:** This question is a bit involved. We suggest that you only attempt this once you have finished all other questions)

Go-back-n:

In GBN, we need 9 sequence numbers (0-8), so the sequence numbers will wrap around back to 0 after 8. So, the 9th packet will have sequence number 0, 10th packet will be sequence number 1 and so on.

The first 3 packets (packet #s 0, 1 and 2) will be correctly received at the receiver. Packet # 3 is lost. From that point on, the receiver will drop all subsequent packets and will keep sending a cumulative Ack (for packet 2) for each subsequent packet that it receives.

The sender window will slide forward for each ACK received (from Ack 0 to Ack 2) and eventually when the base of the window is 3, it will stall as the ACK for 3 will not be forthcoming.

Eventually, the timer for packet 3 will expire and the entire window of packets (3, 4, 5, 6, 7, 8, 0, 1) will be re-transmitted. Note here that the sequence numbers have wrapped around back to 0.

This entire window of packets will be received correctly at the receiver and the corresponding ACKs will make their way back to the sender. The sender will slide the window forward by 1 as it receives each of these ACKs.

Packet 12, which has sequence number 2, is lost. As before, all subsequent packets will be dropped by the receiver and will generate cumulative ACKs for packet #1.

Eventually, the timer for packet # 2 will expire and the entire window of packets (3, 4, 5, 6, 7, 8, 0, 1) will be re-transmitted.

Thus in total, 16 packets with sequence #s, 3, 4, 5, 6, 7, 8, 0 and 1 will be re-transmitted.

Selective repeat:

In selective repeat, each packet is individually acknowledged. A timer is maintained for each packet and only one packet is re-transmitted if the timer expires.

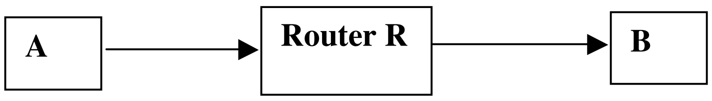
Thus, with SR, only the 2 packets that get lost will be retransmitted, i.e. packets with sequence number 4 and 12.

Question 5 – Switchapalooza (4 marks)

SOLVE ONE OF THE FOLLOWING DEPENDING ON YOUR ENROLLMENT. YOU MAY RECEIVE ZERO MARKS IF YOU ATTEMPT THE INCORRECT VERSION.

COMP3331 (UNDERGRADUATE VERSION)

Consider the network in the figure below. Node A and B are connected to each other through Router R. The link between node A and router R has bandwidth T and propagation delay L. The link between the router R and node B has bandwidth 2T and propagation delay L/2.



Consider two cases:

Case 1) The network is circuit-switched: Assume that the circuit setup between A and B has already occurred and that at time t=0, node A begins transmitting a 1 MByte file to node B. As soon as the last bit of the file has been transmitted, node A immediately transmits a 2 MByte file to B. The last bit of the 1MByte file arrives at node B at time t=0.8 seconds. The last bit of the 2MByte arrives at node B at time t=1.8 seconds. Disregard circuit teardown. Hint: A timing diagram may be useful.

What is L (in msec)? **200**

What is T (in Mbps)? **16**

SHOW YOUR WORK BELOW

The circuit between A and B as a whole can offer service with end-to-end bandwidth T, and end-to-end delay of L+L/2. Recall that in circuit switching, there is no store-and-forward delay at the router.

Time to transmit the 1MB file is 1 MB/T.

Thus, 1 MB/T + L + L/2 = 0.8 sec - (1)

The 2nd file is sent back-to-back, thus,

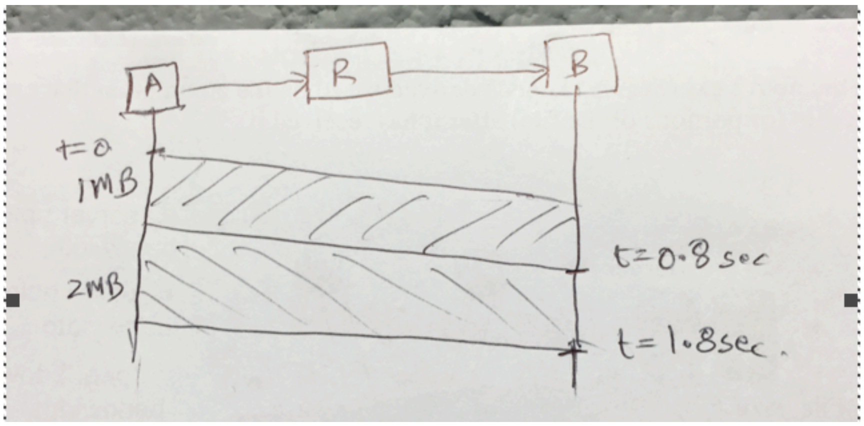
(1MB + 2 MB) /T + L + L/2 = 1.8 sec – (2)

Subtracting (1) from (2), 2MB/T = 1 sec. Thus, T = 2 MB/s (= 18 Mbps.)

Substituting T = 2 MB/s in (1), 0.5 sec + 1.5 L = 0.8 sec

Thus, L = 0.2 sec = 200 msec.

A timing diagram is included below to illustrate the delay



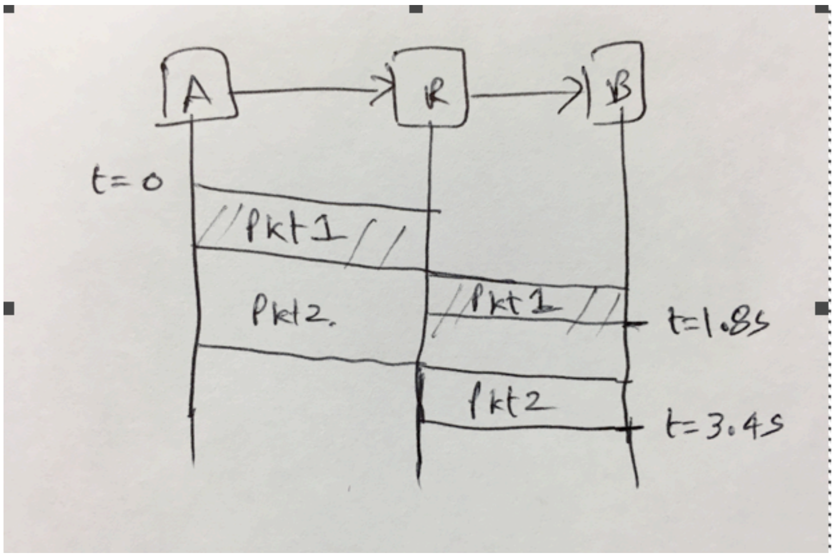
Case 2) The network is packet-switched: Assume that at time t=0, node A begins transmitting a 1 KByte packet to node B. As soon as the last bit of the packet has been transmitted, node A immediately transmits a 2 KByte packet to B. Assume there is no processing delay at the router. The first packet arrives at node B at time t=1.8 milliseconds. The second packet arrives at node B at time t=3.4 milliseconds. Hint: A timing diagram may be useful.

What is L (in msec)? **0.56**

What is T (in Mbps)? **12.5**

SHOW YOUR WORK BELOW

Since, the second link is faster than the first link, there is no queuing delay. See timing diagram below.



The first packet arrives at node B at 1.8ms.

Thus, (1KB/T) + L + (1KB/2T) + L/2 = 1.8 ms – (1)

The second packet will arrive at node B at 3.4 seconds.

Thus, (1KB/T) + (2KB/T) + L + (2KB/2T) + L/2 = 3.4 – (2)

Subtracting (1) from (2),

2KB/T + 1KB/2T = 1.6

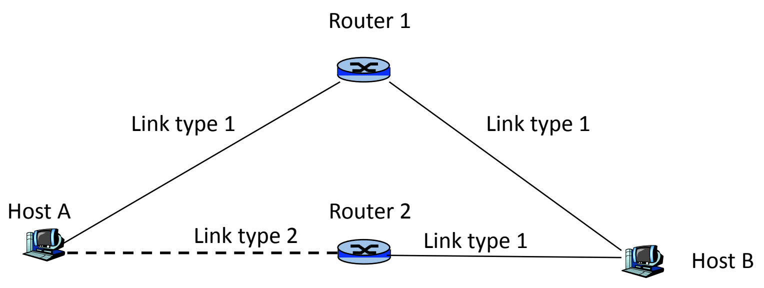
Thus, T = 2.5 KB/1.6 = 12.5 Mbps

Substituting for T in (1),
1.5KB/12.5Mbps + 1.5L = 1.8ms

Thus, L = 0.56ms

COMP9331 (POSTGRADUATE VERSION)

Consider the network in the figure below. Host A can choose between two different paths to communicate with host B. The situation is depicted in Figure 1. Host can choose to send packets via either Router 1 or Router 2 to host B. The communication links are of two different types, as indicated in the figure. The characteristics of these two types of links are:



Link type 1: Each link is of length 2000km, propagation speed is 2 x 10^8 m/s and bandwidth is 100kbps.

Link type 2: Each link is of length 4000km, propagation speed is 2 x 10^8 m/s and bandwidth is 50kbps.

Host A wishes to transmit a message of size 4Kbytes to host B. It breaks this message into 4 packets of equal size. Neglect any packet headers. Remember that routers work on the store-and-forward principle.

Assume that the processing delay and queuing delay in the routers are negligible.

(a) If host A chooses to send the packets via Router 1, draw a timing diagram to show the delay experienced by the packets. By using this timing diagram, determine the time it takes to move the packets from host A to host B, i.e., beginning from the time that host A starts to send the first bit of the first packet till the time that host B receives the last bit of the last packet.

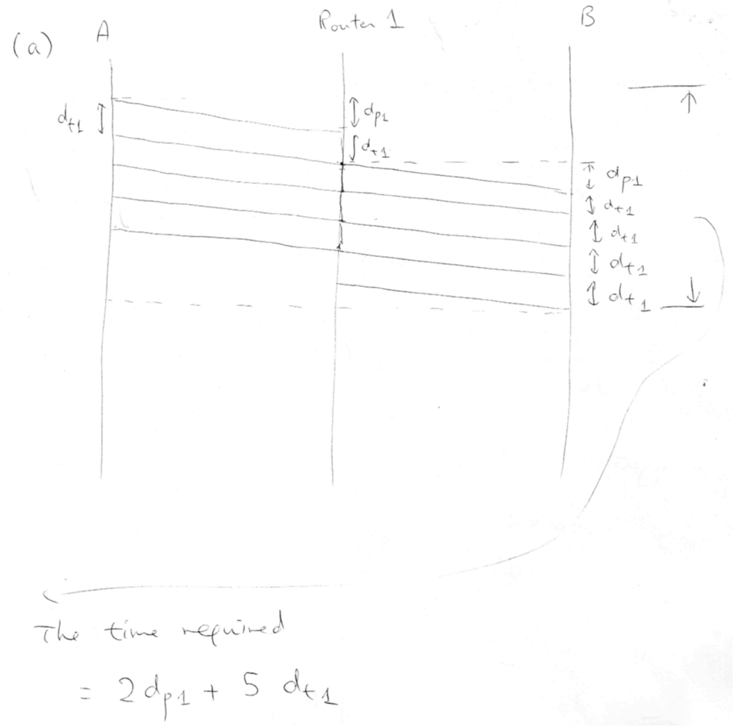
Note: You are required to draw your timing diagram roughly to scale. This will also help you to determine the delay.

Let us begin with a number of basic calculations:

- Propagation delay in link type 1 (d_{p1}) = 2000 x 10^3/2 x 10^8 = 0.01 s
- Transmission delay in link type 1 (d_{t1}) = 8 x 1024/100 x 10^3 = 0.08 s (approx. 1024 by 1000)
- Propagation delay in link type 2 (d_{p2}) = 4000 x 10^3/2 x 10^8 = 0.02 s

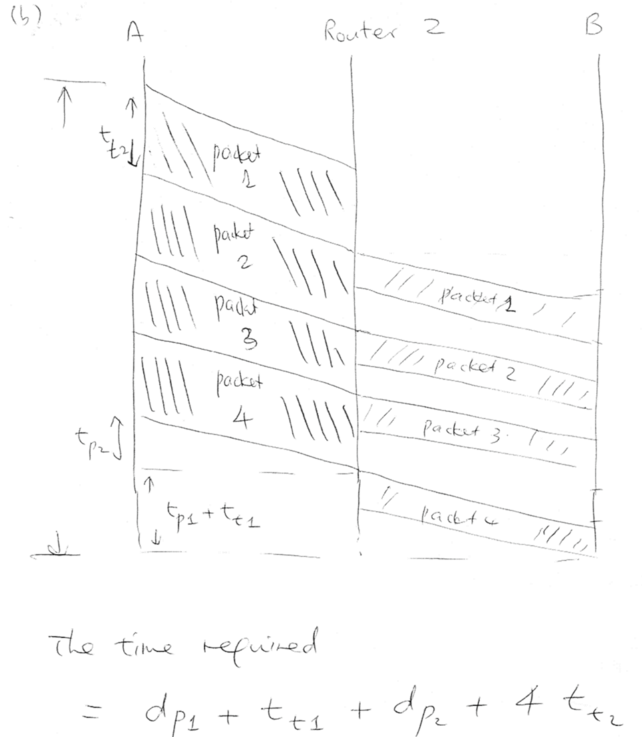
- Transmission delay in link type 1 (d_{t2}) = 8 x 1024/50 x 10^3 = 0.16 s (approx. 1024 by 1000)

From the timing diagram below, the time required = $2d_{p1} + 5d_{t1} = 2(0.01) + 5(0.08) = 0.42$ s.



(b) Repeat part (a), if host A chooses to send the packets via Router 2

From the timing diagram below, the time required = $d_{p1} + d_{t1} + d_{p2} + 4d_{t2} = 0.01 + 0.08 + 0.02 + 4(0.16) = 0.75$ s.



(b)
A
Router 2
B
t_{t2}
packet 1
packet 2
packet 3
packet 4
t_{p2}
$t_{p1} + t_{t1}$
packet 1
packet 2
packet 3
packet 4
The time required
$= d_{p1} + t_{t1} + d_{p2} + 4t_{t2}$