Computer Networks - Midterm

November 1, 2019

Duration: 2h15m

- This is a closed-book exam.
- Please write your answers on these sheets in a readable way, in English or in French.
- You can use extra sheets if necessary (don't forget to put your name on them).
- The total number of points is 600.
- This document contains 20 pages.
- Good luck!

Last Name:
First Name:

SCIPER No:

(answers to the questions are shown in italic and blue) *(grades in red)*

1 Short questions (50 points)

For each question, please circle a single best answer.

1. What is an Internet eXchange Point (IXP)?
   (a) An end-system that is connected to the Internet through multiple ISPs.
   (b) A special packet switch that provides performance guarantees.
   (c) A large ISP that peers with many other ISPs.
   (d) An entity that provides physical connections between ISPs. *(Correct)*

2. Why does the Internet architecture use layers?
   (a) Layers reduce complexity and increase flexibility. *(Correct)*
   (b) Layers reduce packet delay and increase average throughput.
   (c) Both of the above.
   (d) No particular reason.

3. End-systems A and B are connected through one physical link. Adding a second link, of the same properties, in parallel to the first one, may significantly improve:
   (a) The propagation delay from A to B.
   (b) The average throughput from A to B. *(Correct)*
   (c) Both of the above.
   (d) None of the above.

4. End-system A is continuously sending packets back-to-back to end-system B. If A doubles the size of the packets, the following will happen:
   (a) The transmission delay of each packet will double. *(Correct)*
   (b) The average throughput from A to B will double.
   (c) The average throughput from A to B will be reduced by half.
   (d) None of the above.

5. Which of the following is a valid packet that you may observe leaving your computer (e.g., using Wireshark)?
   (a) An HTTP GET request, encapsulated in a DNS message.
   (b) An HTTP GET request, encapsulated in a TCP segment. *(Correct)*
   (c) An HTTP GET request, encapsulated in a UDP segment.
   (d) A TCP segment, encapsulated in an HTTP GET request.

6. All the mappings from `epfl.ch` DNS names to IP addresses expire at the same time. Which of the following may happen over the next few hours as a result?
   (a) Authoritative DNS servers for `epfl.ch` may receive a higher rate of DNS requests.
   (b) TLD servers for `.ch` may receive a higher rate of DNS requests.
   (c) Root DNS servers may receive a higher rate of DNS requests.
   (d) All of the above. *(Correct)*

7. What does it mean that peer-to-peer (P2P) file distribution "scales better" than client-server?
   (a) P2P file distribution always takes less time.
   (b) P2P file distribution always takes the same or less time.
   (c) As the number of downloaders increases, P2P file distribution time stays the same.
   (d) As the number of downloaders increases, P2P file distribution time increases at a lower rate. *(Correct)*

8. What is the minimum number of sockets that a DNS server process must open?
   (a) 1. *(Correct)*
   (b) 1 listening socket and 1 connection socket.
   (c) 1 listening socket and 1 connection socket per DNS client it is communicating with.
   (d) 1 listening socket and 1 connection socket per DNS client and DNS server it is communicating with.

9. End-systems A and B need reliable data delivery (RDD) over a network that may corrupt but never drops packets. Which of the following RDD elements is not needed in this scenario?
   (a) Checksums.
   (b) Timeouts. *(Correct)*
   (c) Retransmissions.
   (d) Acknowledgments.

10. End-systems A and B communicate over one of the two pipelined transport-layer protocols that we saw in class. Is it possible that several segments get lost without resulting in any timeout?
    (a) No.
    (b) Yes, if A and B communicate over Go-back-N. *(Correct)*
    (c) Yes, if A and B communicate over Selective Repeat.
    (d) Yes, if the network between A and B never reorders packets.

2 Web browsing (150 points)

Alice's computer is somewhere outside EPFL or ETHZ. It accesses the web through proxy web server `proxy.net` (whose IP address it knows), which is in the same network as Alice. Alice's web browser does NOT cache web pages. All end-systems in Alice's network use `1.1.1.1` as their local DNS server.

Alice uses her web browser to access web page `www.epfl.ch`, which contains 1 embedded image with URL `www.epfl.ch/1.jpg`. Right after, she uses her web browser to access web page `www.ethz.ch`, which contains 1 embedded image with URL `www.ethz.ch/1.jpg`. The two images happen to be identical (i.e., identical copies of the same image).

Question 1 (100 **points):**

What is the maximum and what is the minimum number of application-layer messages (sent by any end-system, including any DNS server, and any proxy or origin web server) that may result from Alice's actions? Do not count TCP connection setup requests and responses. Justify your answer. State the role of each message, which end-system sends it, and which one receives it.

Maximum 32 packets:

- 1 HTTP GET request from Alice to `proxy.net` for the EPFL base file + 1 response.
- 1 DNS query from `proxy.net` to `1.1.1.1` to resolve the IP address of `www.epfl.ch` + 1 DNS response.
- 3 DNS queries from 1.1.1.1 to a root DNS server, a `.ch` TLD server, and an `epfl.ch` authoritative server + 3 DNS responses.
- 1 HTTP GET request from `proxy.net` to `www.epfl.ch` for the base file + 1 response.
- 1 HTTP GET requests from Alice to `proxy.net` for the EPFL image + 1 response.
- 1 HTTP GET request from `proxy.net` to `www.epfl.ch` for the image + 1 responses.
- Repeat all of the above, but for obtaining the ETHZ base file and images.

Minimum 16 packets: As above minus the DNS queries, because Alice's DNS client may have previously cached the IP addresses of `www.epfl.ch` and `www.ethz.ch`, in which case there are no DNS queries and responses. Whether `proxy.net` has previously cached the base files and/or images or not does not change the number of exchanged messages; it only makes the HTTP GET requests for the base diles and images conditional.

Question 2 (25 **points**):

How many sockets are created and/or used on the EPFL web server as a result of Alice's actions? What is the type and purpose of each socket? What IP address(es) and port number(s) is each socket associated with?

- 1 TCP listening socket for incoming connection requests.
  - Local IP address: the EPFL web server's IP address.
  - Local port number: 80.
- 1 TCP connection socket to communicate with the proxy web server.
  - Local IP address: the EPFL web server's IP address.
  - Local port number: 80.
  - Remote IP address:the proxy's IP address.
  - Remote port number: whatever port number the proxy web server used to initiate the connection.

Question 3 (25 **points):**

Bob opens a terminal on his computer and types “dig epfl.ch MX”. In return, he gets the following answer:

```
my-pc:Desktop me$ dig epfl.ch MX

; <<>> DiG 9.10.6 <<>> epfl.ch MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3105
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 11

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;epfl.ch.                        IN      MX

;; ANSWER SECTION:
epfl.ch.                86256    IN      MX      50 mx1.epfl.ch.
epfl.ch.                86256    IN      MX      50 mx2.epfl.ch.
epfl.ch.                86256    IN      MX      50 mx3.epfl.ch.

;; AUTHORITY SECTION:
epfl.ch.                78946    IN      NS      stisun1.epfl.ch.
epfl.ch.                78946    IN      NS      stisun2.epfl.ch.

;; ADDITIONAL SECTION:
mx1.epfl.ch.            86256    IN      A       128.178.166.20
mx2.epfl.ch.            86256    IN      A       128.178.166.21
mx3.epfl.ch.            86256    IN      A       128.178.166.22
stisun1.epfl.ch.        78946    IN      A       128.178.15.8
stisun2.epfl.ch.        78946    IN      A       128.178.15.7

;; Query time: 13 msec
;; SERVER: 1.1.1.1#53(1.1.1.1)
;; WHEN: Sun Oct 27 20:27:07 CET 2019
;; MSG SIZE  rcvd: 360
```

- What is `mx1.epfl.ch`? Think of one socket that you are certain is active (has been created and not deleted) on this end-system. What is its type and purpose? What IP address(es) and port number(s) is it associated with?
- What is `stisun2.epfl.ch`? Think of one socket that you are certain is active on this end-system. What is its type and purpose? What IP address(es) and port number(s) is it associated with?

- `mx1.epfl.ch` is one of the EPFL SMTP (mail) servers. There should be at least one active TCP socket on this end-system listening for incoming connections. It is associated with local IP address `128.178.166.20`, and port number 25 (or 587).
- `stisun2.epfl.ch` is one of the EPFL authoritative DNS servers. There should be at least one active UDP socket on this end-system to receive DNS requests. It is associated with local IP address `128.178.15.7`, and port number 53.

Note: It was OK if a student did not remember the exact port numbers associated with SMTP or DNS. It was enough to say that each of the two end-systems should be listening at a given port number that corresponds to the service they offer (SMTP or DNS).

3 Transport layer (200 **points)**

Consider two end-systems A and B. A process running on A has an infinite amount of data to send to a process running on B. The two processes communicate over a pipelined transport-layer protocol with the following properties:

- Window size N.
- Retransmission timeout $1.01RTT$, where RTT is the round-trip time between A and B.
- Sequence numbers $0 - 1023$.

A's transport layer generates data segments of size L. Header size is insignificant relative to L. The transmission delay of any single segment from A to B is insignificant relative to the RTT between A and B.

The following loss pattern repeats indefinitely: The first $N/2$ original segment transmissions sent from A to B are lost, the next $N/2$ successfully delivered, the next $N/2$ lost, the next $N/2$ successfully delivered, and so on. Retransmitted segments are never lost or corrupted. ACK segments sent from B to A are never lost or corrupted. For example, if $N = 4$, the segments with the following SEQs are lost the first time they are transmitted: $0, 1, 4, 5, 8, 9, 12, 13$, and so on. No other segments are lost or corrupted.

Question 1 (100 **points):**

Suppose the window size is $N = 4$. Complete the diagrams in Figs. 1 and 2 to show what happens if A and B communicate over Go-back-N (GBN), and if they communicate over Selective Repeat (SR). Please show:

- All transmitted (data and ACK) segments with their SEQ or ACK numbers.
- All segments that are lost.
- Timeout events.
- Changes in window status.

Show only until (but not including) the moment where A sends segment with SEQ 8 for the first time.

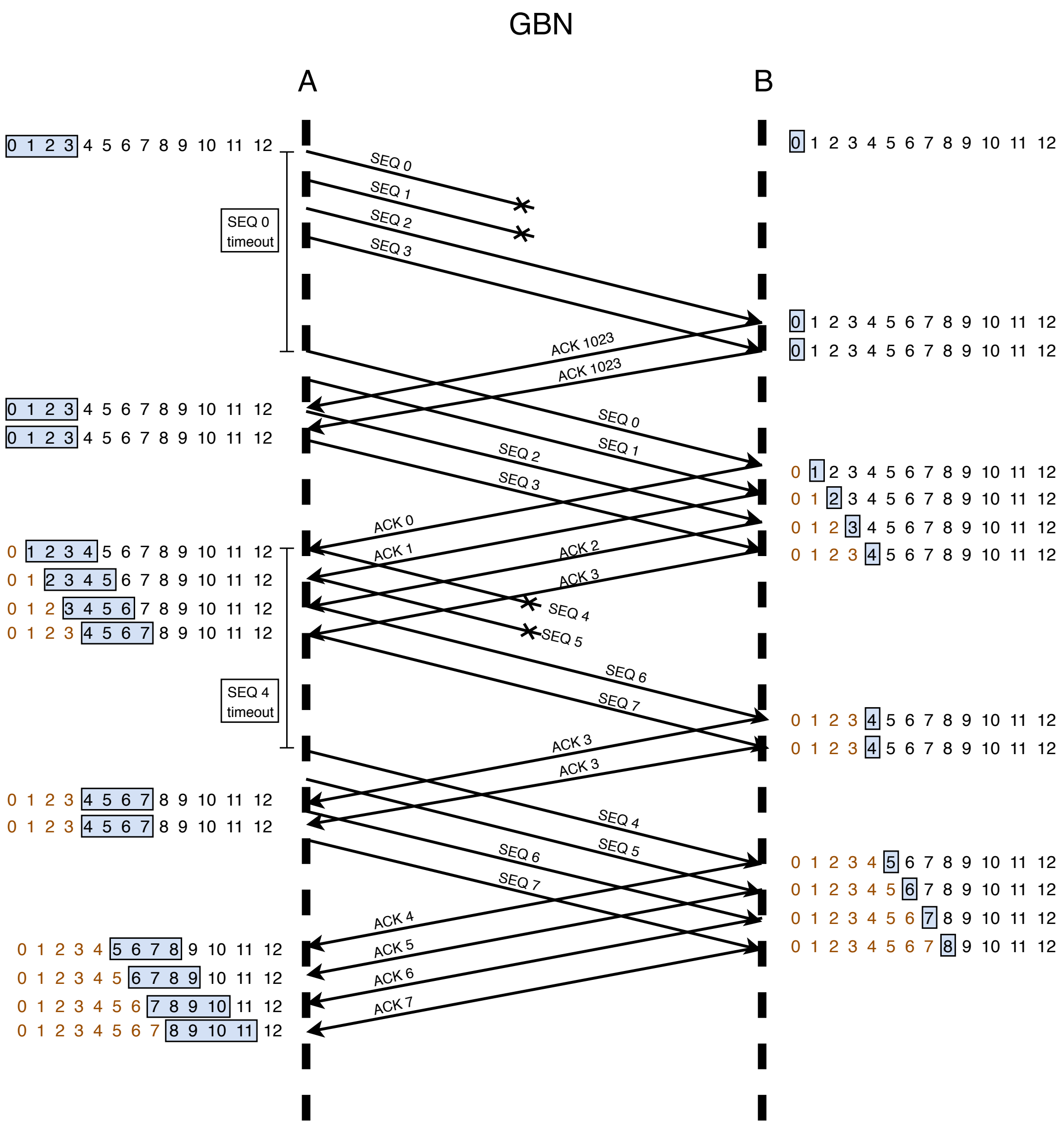

Figure 1: Sequence diagram for Question 1 for GBN.

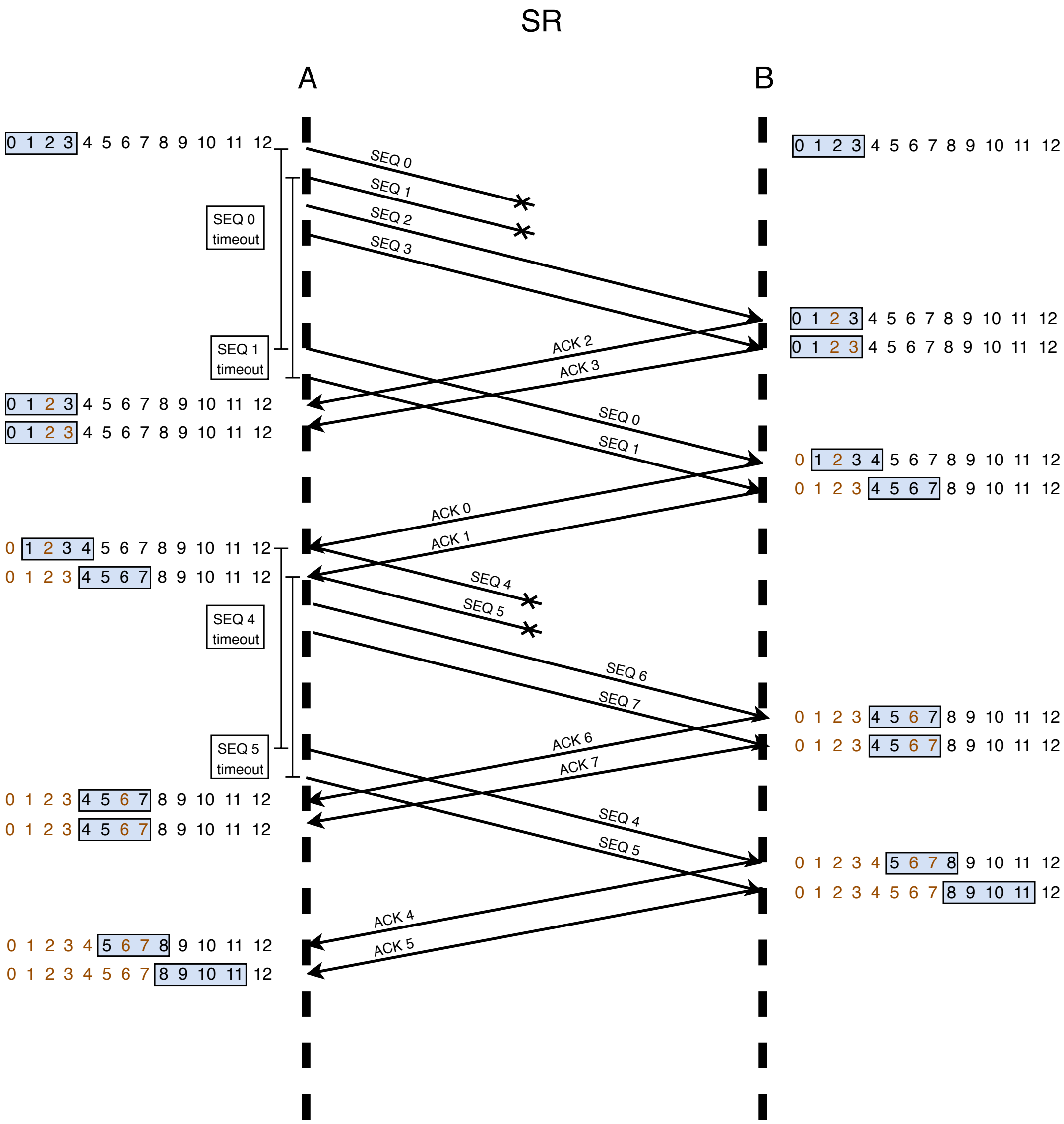

Figure 2: Sequence diagram for Question 1 for SR.

Question 2 (25 **points):**

In the scenario of the last question, what is the average throughput achieved by the communicating processes if they communicate over GBN? if they communicate over SR? Express the average throughput as a function of L and RTT. Justify your answer. Be careful: in this context, the average throughput is the average rate at which the process running on B receives data sent by the process running on A.

The average throughput is data size over transfer time. For both GBN and SR, every two rounds:

- A communicates $4L$ bytes of data to B (because A successfully transmits 4 segments, each segment is L bytes, and header size is insignificant relative to L).
- The two rounds last approximately $2RTT$ (because transmission delays are insignificant relative to RTT).

Hence, for both GBN and SR, the average throughput is approximately $\frac{4L}{2RTT} = \frac{2L}{RTT}$.

Question 3 (50 **points):**

Generalize your answer to Question 2 to any window of size N: what is the average throughput achieved by the communicating processes if they communicate over GBN? if they communicate over SR? Express the average throughput as a function of L, RTT, and N. Justify your answer.

For both GBN and SR, every two rounds:

- A communicates NL bytes of data to B (because A successfully transmits N segments, each segment is L bytes, and header size is insignificant relative to L).
- The two rounds last approximately $2RTT$ (because transmission delays are insignificant relative to RTT).

Therefore, for both GBN and SR, the average throughput is approximately $\frac{NL}{2RTT}$.

Question 4 (25 **points):**

Suppose that, at some point, one of the packet switches traversed by A-to-B traffic becomes congested (due to traffic exchanged by other end-systems), so, all the packets that arrive at the packet switch experience significant queuing delays. In this situation, do you think that it would create less network congestion if A and B communicated over GBN or over SR? Justify your answer.

It will create less network congestion if A and B communicate over SR. When there is network congestion, there is frequent packet loss, hence both GBN and SR will timeout frequently. On timeout, GBN retransmits all unacknowledged segments, whereas SR retransmits only one, hence GBN generates more network traffic and contributes more to network congestion.

4 Delay computation (200 points)

Consider the network topology in Figure 3.

- End-system A stores a file of size F bits.
- N other end-systems ("downloaders"), $D_i, i = 1..N$, want to download this file.
- "File distribution time" is the amount of time that elapses from the moment A sends its first bit until the moment the last downloader receives the last bit of the file.
- When an end-system decides to send a set of packets, it sends them back to back, i.e., it transmits the first bit of one packet as soon as it finishes transmitting the last bit of the previous one.
- Packet switch S performs store-and-forward packet switching and introduces insignificant processing delays. S has one infinite-size queue per output link. Whenever a packet arrives at S, S determines the output link and places the packet in the corresponding queue.
- There is no packet loss or corruption.
- Each link has transmission rate R, propagation speed c (both directions) and length ℓ.

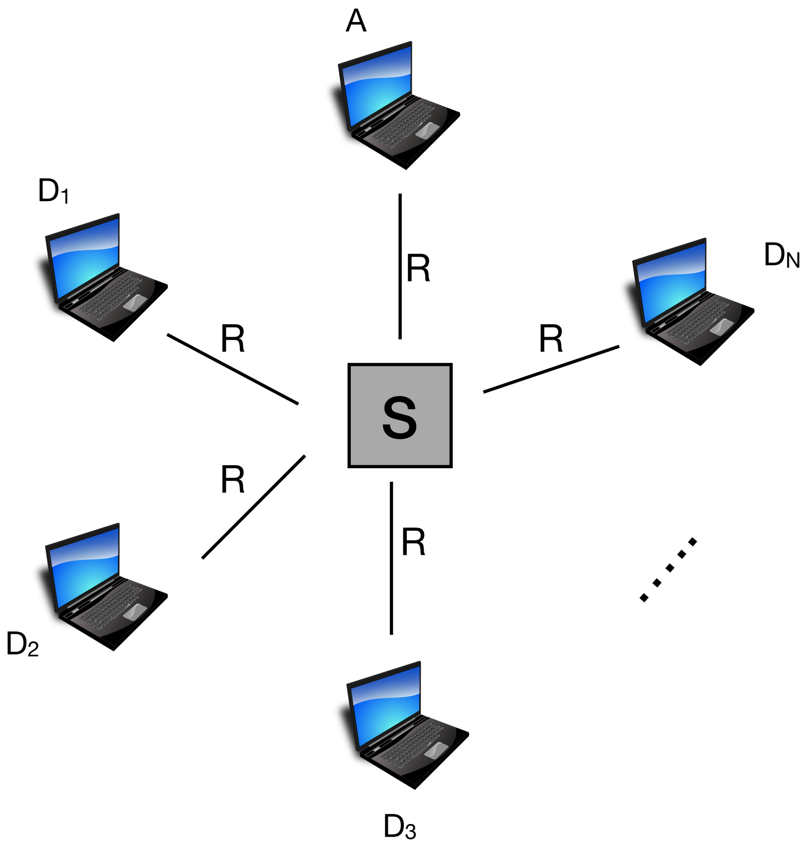

Figure 3: Network topology for Problem 4.

We will examine two different ways to distribute the file:

- In client/server mode,
  - A creates N copies of the file.
  - A sends one copy of the file to each downloader. To do so, A splits each of the N file copies into N chunks. Each chunk has size $\frac{F}{N} = L$ bits and fits exactly in a single packet. Then, A sends all the chunks back to back.
- In peer-to-peer (P2P) mode,
  - A distributes pieces of the file to the downloaders:
    * A splits the file into N chunks: $C_i, i = 1..N$.
      Each chunk has size $\frac{F}{N} = L$ bits and fits exactly in a single packet.
    * Chunk C_i is destined to downloader D_i.
    * A sends all the chunks back to back.
  - The moment the last downloader receives the last bit of its chunk,
    all downloaders start distributing their chunks to each other:
    * Downloader D_i creates $N - 1$ copies of chunk C_i: $C_{ij}, j = 1..N, j \neq i$.
    * Copy C_{ij} is destined to downloader D_j.
    * D_i sends all the copies back to back starting with $C_{i,(i+1)\%N}$, then $C_{i,(i+2)\%N}$, and so on. For example, D_3 would send $C_{34}, C_{35}, \ldots C_{3N}, C_{31}, C_{32}$.

Question 1 (50 points):

Suppose the file distribution happens in client/server mode. What is the file distribution time? Justify your answer.

$$
\begin{aligned}
D_{CS} = \;& \textit{time for } A \textit{ to transmit } N \textit{ copies of the file over the link } A \to S \\
& + \textit{propagation delay over the link } A \to S \\
& + \textit{time for } S \textit{ to transmit the last chunk to the last downloader } D_{last} \\
& + \textit{propagation delay over the link } S \to D_{last} \\
= \;& NN\frac{L}{R} + \frac{l}{c} + \frac{L}{R} + \frac{l}{c} \\
= \;& (N^2+1)\frac{L}{R} + 2\frac{l}{c}.
\end{aligned}
$$

Question 2 (50 points):

Suppose the file distribution happens in P2P mode. What is the lower bound of the file distribution time (based on the formula we computed in class)? Justify your answer.

Here is the formula we computed in class:

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

where u_s is the upload capacity of the end-system that originally has the file, d_{min} is the minimum download capacity of any downloader, and u_i is the upload capacity of downloader i.

In this scenario, all the upload and download capacities are equal to R, hence the bound becomes:

$$D_{P2P} \geq \max\left\{\frac{F}{R}, \frac{NF}{(N+1)R}\right\} = \frac{F}{R} = N\frac{L}{R}.$$

Question 3 (50 points):

Suppose the file distribution happens in P2P mode. What is the file distribution time? Justify your answer.

$$D_{P2P} = \Delta t_{phase1} + \Delta t_{phase2} = (2N+1)\frac{L}{R} + 4\frac{l}{c},$$

where:

$$\begin{aligned}
\Delta t_{phase1} = & \; \textit{time for } A \textit{ to transmit } N \textit{ chunks over the link } A \to S \\
& + \textit{propagation delay over the link } A \to S \\
& + \textit{time for } S \textit{ to transmit the } N^{th} \textit{ chunk to the last downloader } D_{last} \\
& + \textit{propagation delay over the link } S \to D_{last} \\
= & \; N\frac{L}{R} + \frac{l}{c} + \frac{L}{R} + \frac{l}{c} \\
= & \; (N+1)\frac{L}{R} + 2\frac{l}{c}.
\end{aligned}$$

$$\begin{aligned}
\Delta t_{phase2} = & \; \textit{time for } D_i, i \in [1, N] \textit{ to transmit } (N-1) \textit{ chunks over the link } D_i \to S \\
& + \textit{propagation delay over the link } D_i \to S \\
& + \textit{time for } S \textit{ to transmit the last chunk to downloader } D_j, j \in [1, N] \\
& + \textit{propagation delay over the link } S \to D_j \\
= & \; (N-1)\frac{L}{R} + \frac{l}{c} + \frac{L}{R} + \frac{l}{c} \\
= & \; N\frac{L}{R} + 2\frac{l}{c}.
\end{aligned}$$

The second formula holds only because (in phase 2) each downloader sends its chunk to the other downloaders in the very specific way described in the problem formulation (counter-clockwise). As a result, S receives a new chunk for each downloader as soon as it finishes transmitting the previous one. So, S sends all the chunks back-to-back, and none of the chunks experience queuing delay.

Question 4 (50 points):

Consider again the P2P scenario, but with the following modification: αN downloaders are "helpful," i.e., want to help A distribute the file, while $(1-\alpha)N$ downloaders are "leechers," i.e., want to download the file without helping distribute it. $\frac{1}{\alpha}$ is an integer and αN is an integer.

Can you propose a modification to the P2P mode that will enable the file to be distributed to all the downloaders—helpful and leechers—as fast as possible? What will be the file distribution time? Justify your answer.

Assume that: (1) A knows which of the downloaders are helpful and which ones are leechers, before file distribution begins. (2) All the helpful downloaders start distributing at the same time. Note that we are not looking for a scheme that punishes/disadvantages the leechers, like tit-for-tat etc.

Here is one option (though other correct ones exist):

In the first phase, S splits the file into N chunks, each of size $\frac{F}{N} = L$ bits, and sends $\frac{1}{\alpha}$ chunks to each helpful downloader. In the second phase, each helpful downloader sends each of its chunks to each of the other downloaders. This way, all downloaders get all chunks, while S leverages help from all the helpful downloaders.

Regarding the file distribution time: Δt_{phase1} remains the same (as in Question 3) since S sends out the same amount of data. In the second phase, each helpful downloader sends $\frac{1}{\alpha}$ chunks to each of the other downloaders. Hence, each helpful downloader finishes sending its last chunk after $\frac{1}{\alpha}(N-1)\frac{L}{R}$. If the helpful downloaders send out their chunks in an organized manner (e.g., all of them counter-clockwise), such that S receives one packet for each downloader during each time slot, then S sends all the chunks back to back, while the chunks experience no queuing delay. In that case, phase 2 finishes after time $\frac{1}{\alpha}(N-1)\frac{L}{R} + \frac{L}{R} + 2\frac{l}{c}$.

To sum up:

$$D_{P2P} = \Delta t_{phase1} + \Delta t_{phase2} = \left(N + 2 + \frac{N-1}{\alpha}\right)\frac{L}{R} + 4\frac{l}{c},$$

where:

$$\Delta t_{phase1} = (N+1)\frac{L}{R} + 2\frac{l}{c}.$$

$$\Delta t_{phase2} = \frac{1}{\alpha}(N-1)\frac{L}{R} + \frac{L}{R} + 2\frac{l}{c}.$$