CSEE 4119 Sample Midterm Key

Uploading course materials, including questions/answers from this midterm, to sites such as CourseHero, Chegg or Github is academic misconduct at Columbia (see pg 10).

Short Answer Questions (28 points)

Go-Back-N and Selective Repeat

1. (8 points) Consider Go-Back-N and Selective Repeat. (1-2 sentences each)
    a. What is an advantage of Go-Back-N over Selective Repeat?

Multiple correct answers (2pts)

- Only one timer
- Receiver does not need to buffer out-of-order segments
- For fixed sequence number space, GBN can transmit more efficiently with larger window size because it allows more unacked packets on the wire
- SR requires 2 * window_size <= sequence number size, GBN doesn't.

Partial credit (1pt)

- Simpler (without describing one of the ways above in which it is simpler)
- Simpler to implement (without more details including one of the correct answers)
- Less space in header of packet without explaining why.
- Cumulative ACK so fewer ACKs needed to re-ack successfully received packets.
- In GBN, the receiver doesn't need a sliding window (or only need a sliding window of size 1)

Wrong (0pt)

- In GBN, packets are guaranteed to receive in-order. (both GBN and SR deliver segments in order to the application)

    b. What is an advantage of Selective Repeat over Go-Back-N?

    Correct answers (2pts)

    - Buffers out of order segments, so fewer retransmissions required
    - (ok to call this more efficient or more resource efficient, as long as it says it is because it doesn’t have to discard out of order segments

Partial answers (1pt)

- More efficient/less transmission without mentioning how SR deals with out of order packets using a receiver-side buffer.
- Only explains SR retransmits when necessary but doesn't mention this is done by having the receiver-side buffer or by not discarding the out-of order packets.

c. Are there ways in which TCP is more like Go-Back-N than like Selective Repeat? Answer Yes or No. If Yes, include a brief description of one way.

Correct answer (2pts)
Yes, mention at least one of the following key ideas:

- Smallest transmitted but unacked seq #: TCP sender need only maintain the smallest seq # of a transmitted but unacknowledged byte (and the seq # of the next byte to be sent)
- Receiver resends the smallest missing sequence number (duplicative ack for the oldest missing sequence).
- Both uses cumulative ACKs.

Partial answer (1pt)
Yes. Without explaining why.
Yes, but reason with in-order delivery: Selective repeat also has in order delivery.
Yes, TCP only maintains 1 timer.

.

Incorrect answer (0pt)
Answering no, regardless of reasoning.

d. Are there ways in which TCP is less like Go-Back-N than like Selective Repeat? Answer Yes or No. If Yes, include a brief description of one way.

Correct answer (2pts)
Yes, mention at least one of the following key ideas:

- Selective retransmission (skipping retransmitting segments that have already been acknowledged by receiver, and only transmit one sequence)
- Selective acknowledgement (allowing TCP to ack out-of-order segments selectively with a buffer rather than cumulatively acknowledging the last received. Notice that TCP RFCs do not impose any rules to enforce how out-of-order packets are handled on the receiver side, but it's indeed the

approach taken in practice for higher efficiency. Please refer to the textbook (ed7. Page 239) for a detailed explanation.)

- Receiver buffers out of order packets.

Partial answer (1pt)
Yes, without explaining or fail to explain why.

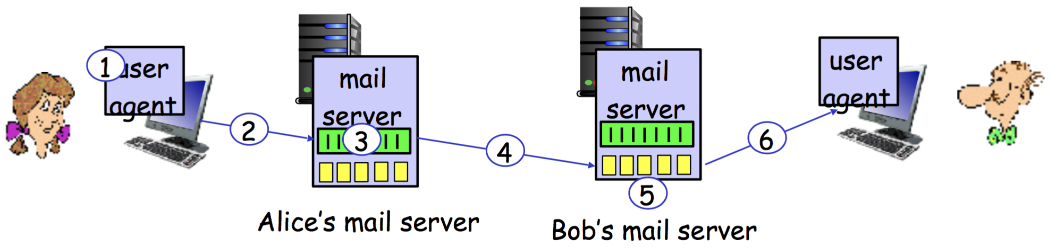
Incorrect answer (0pt)
Answering no, regardless of reasoning.

Conditional Get

2. (6 points) Answer the following questions in short sentences:
   a. What protocol includes a **Conditional Get**?
      HTTP
      +1 for HTTP
   b. What is the purpose of a **Conditional Get**?
      The Conditional Get is a mechanism that allows a cache to verify its objects are up to date. +2
      +2 for mentioning verifying object up to date, or if object has been modified.
   c. How does **Conditional Get** work? Describe the pattern of communication between the entities involved.
      After the user proxy sends a GET request with an IF-Modified-Since date to server. If the object on the server has not been modified since the specified time, the server sends Not-Modified message with empty body; otherwise sends the updated object to the proxy.
      +1 for mentioning user proxy send GET with IF-Modified-Since;
      +1 for mentioning when modified server sends the object
      +1 for mentioning when not modified, server sends Not-modified.

Email Protocols

3. (6 points) Suppose Alice is sending an **email** to Bob. Answer the following questions:



   a. What is a protocol commonly used in step 2?
      SMTP
      +2 for SMTP, +1 for TCP or DNS. If multiple protocols are written, both protocols must be correct to get credit.

   b. What is a protocol commonly used in step 4?
      SMTP
      +2 for SMTP, +1 for TCP. If multiple protocols are written, both protocols must be correct to get credit.

   c. What is a protocol commonly used in step 6?
      POP3, IMAP, or HTTP
      +2 for any of the above, +1 for TCP. If multiple protocols are written, both protocols must be correct to get credit.

Transport Protocols

4. (8 points) Consider clients A and B, communicating with server S using either UDP or TCP (as specified below). Answer True or False for the following statements. If it's false, please provide a brief explanation with 1 or 2 sentences. Missing or wrong explanation won't receive any credit.
   a. If using UDP, A and B must explicitly (i.e. call the bind function to) bind to a port. But if using TCP, they don't have to do that.
      Answer: False. In both protocols, clients don't need to bind a port explicitly to communicate with the server.
      +2; wrong explanation for one protocol -1;
   b. S must explicitly bind to a port regardless of whether it is using TCP or UDP
      Answer: True.
      right answer +2 or right explanation for one protocol +1
   c. If using UDP, S can use the same port to communicate with A and B. But if using TCP, S can't use the same port to communicate to both clients.
      Answer: False. In TCP, server can also use the same port like 80 or 8080 to serve multiple clients.
      +2; wrong explanation for one protocol -1;

   d. S can use the same socket to communicate with A and B, whether it is using UDP or TCP.
      Answer: False. In TCP, server would open a new socket for each client.
      +2; wrong explanation for one protocol -1;

DNS (12 points)

Host A, B, and C connect to the Internet for the first time and must perform DNS lookups to find IP addresses for websites they want to access. Make the following assumptions in this problem:

- All hosts know the IP address of their local DNS server, and all DNS servers know the hostname and IP address of the next servers in the hierarchy.
- All three hosts are connected to the same ISP and use the same local DNS server.
- None of the hosts or servers have any additional information cached at the beginning. However, as queries are made, all results are cached and remain indefinitely.
- Hosts make recursive queries to their local DNS server, which makes iterative queries to other DNS servers in the network.
- DNS resolvers will respond to request with both a hostname and its corresponding IP address (both NS-type and A-type). The emphasis in this question is on the hierarchy of the DNS system rather than the exact structure of the messages sent.

(a) [4 points] Host A, connects to the Internet for the first time and tries to connect to www.columbia.edu and must perform a DNS lookup to find Columbia's IP address. List the different queries made, in order, between different hosts and DNS servers until Host A gets the IP address for www.columbia.edu.

1) Host A queries local DNS server
2) Local DNS server queries root DNS server; root responds with name/IP for .edu TLD server
3) Local DNS server queries .edu TLD server; gets response with name/IP for Columbia's authoritative server
4) Local DNS server queries Columbia's server; gets response with name/IP for www.columbia.edu
5) Local DNS returns name/IP to Host A

For all parts, full credit given if student considers each query and response as two separate steps, as long as the order is still correct.

2 points: Correctly understood hierarchy and process flow
-1 point for confusing recursive and iterative process or not specifying sender
, -2 points for no indication of hierarchy

2 points: Correct identification of specific servers: root, TLD (.edu), authoritative (Columbia)
-1 point per missing step (up to maximum of -2)
-0.5 if TLD/.edu server described as something else (e.g. "regional server")
Full credit if other servers misnamed as long as function is clear (e.g. "global" instead of "root")

(b) [4 points] After, Host B then tries to access www.nyu.edu and performs a DNS lookup in the same fashion. List the different queries made for this process.

(i) Host B queries local DNS server
(ii) Local DNS server queries .edu TLD server (which was saved in its cache); gets response with IP for NYU's authoritative server
(iii) Local DNS server queries NYU's server; gets response with IP for www.nyu.edu
(iv) Local DNS returns IP to Host B

2 points: Identify saved step because of caching

+2 points if correctly skips root server

+1 point if some other step was skipped instead. Partial credit only given if a step was removed compared to student's answer in part (a).

2 points: Correct process flow chain.

-1 point for each missing step.

-1 for recursive instead of iterative flow

(c) [4 points] Then, Host C tries to access www.columbia.edu for the first time. List the queries made in this DNS lookup process.

(1) Host C queries local DNS server
(2) Local DNS server responds immediately with IP address from its cache

4 points total:

-2 for skipping local DNS server (i.e. student assumes Host C will have address already)

-1 for each unnecessary step

Utilization (8 points)

Suppose two hosts (source and destination) are directly connected using the **stop-and-wait** protocol. The transmission rate is R. The propagation speed is V. The distance between the two hosts is D. The size of one data packet is F. The size of one ACK is F_{ack}. Assume that the packet processing time is negligible.

a. (4 points) Assume the sender starts a timer as soon as it finishes transmitting a packet. What is the minimum safe timeout interval t_{out} that avoids spurious timeout?

Propagation time:

$$t_{prop} = D/V$$

Time to transfer one ACK packet:

$$t_{I(ACK)} = F_{ack}/R$$

Ideal timeout interval counts from the time when last bit of packet is sent to the time when ACK of this packet is received at sender:

$$t_{out} = 2t_{prop} + t_{I(ACK)} = 2D/V + F_{ack}/R$$

If write $2D/V + F_{ack}/R$ for the final answer, +4;

If write $2D/V + F_{ack}/R + F/R$ for the final answer, +3;

If final answer is incorrect, but write propagation time $=D/V$ and time to transfer one ACK$=F_{ack}/R$, +1 for each;

Otherwize +0.

(4 points) Derive the sender's utilization of its connection. Assume that packets (both data and ACK) never fail (no packet loss or corruption).

Time for one packet transmission $t_I = F/R$

Utilization:

$$utilization = t_I/(t_I + t_{out}) = F/R/(F/R + 2D/V + F_{ack}/R)$$

If write $F/R/(F/R + 2D/V + F_{ack}/R)$ for final answer, +4;

If final answer is incorrect but write time to transmit one packet is F/R, and $utilization = t_I/(t_I + t_{out}) = t_{activelyTrans}/(t_{activelyTrans} + t_{idlelyWait}))$, +1 for each;

Otherwise +0.

File Segmentation (14 points, parts a-h)

Considering transferring a file of length L=1 MB from Host A to Host B. The path from A to B has two links, of the same transmission rate R=2 Mbps, and packet-switching is used.
(Ignore propagation and processing delays in this problem.)

a. [2 points] How long will it take to transfer this file from Host A to Host B, if sent as a single large packet?

b. [2 points] If we divide the file into 1000 packets, how long does it take to move the first packet from Host A to Host B?

c. [2 points] Following b), the file is divided into 1000 packets, and suppose we generate these 1000 packets in one time. What's the average queuing delay experienced by the 1000 packets, considering all queueing they experience along the path? (Hint: Since 1000 packets are generated in one time, you can imagine 1000 packets arrive simultaneously at the first link.)

d. [2 points] Following c), the file is divided into 1000 packets and generated in one time, what's the transmission delay in total for all the 1000 packets?

e. [2 points] Following b), the file is divided into 1000 packets, but for this time, the next packet won't be transmitted until the previous packet completely arrives at the switch between two hosts. How long will it take to move all the packets from Host A to Host B?

f. [2 points] Compare the result obtained in a) and e), please explain why the time cost differs.

g. [1 points] Name one advantages of file segmentation in the transmission. ("By segmentation," we mean the process of breaking up into packets, like in e).)

h. [1 points] Please name another advantage of file segmentation.

Solution:

a. Only one file, so there is no queuing delays.
   10^6 * 8 bit / 2*10^6 bps * 2 (links)= 8 s

b. The queuing delay is 0 for the first transmitted packet.
   10^6 * 8 / 1000 = 8000 bit/each packet
   8000 / 2Mbps * 2(links)= 8ms

c. The queuing delay for the first packet is 0, 4ms for the second packet, and generally, (n-1)*4 ms for the nth transmitted packet. Thus the average delay for 1000 packets is (4 + 8 + … + 999*4) / 1000 ms = 999*500*4 ms = 1998 s
(there is no queuing delay in the second link)

d. It's the same as a), 8s

e. No queuing delay under this case.
The first packet will use 4ms*2 = 8ms to reach Host B. Then every 4ms a new packet will be received. 8ms + 4ms*999 = 4004 ms = 4.004 s

f. The queuing delay in both cases is 0, and we only need to consider the transmission delay.
Because the switch uses store-and-forward packet switching method, for a), the switch cannot begin to transmit the file until it receives it fully over the 1st link, resulting in a long delay before the switch can begin transmitting even the beginning of the file.
But for e), after the first packet arrives at the switch, both of the links could be busy transmitting segments of the files at the same time, thus decreasing the delay.

g. Reduce delay

h. Loss/bit tolerant: If one packet is lost or some bits are not correctly transmitted during the transmission, we needn't transfer the whole file.
If other answers make sense, it's all right.

TCP: On Elephants and Mice [11 points]

An elephant flow is an extremely large (several MBytes) TCP flow between a sender and a receiver. In contrast, mice flows are small (tens of KB, say) TCP connections. Even though most flows are short mice, the small number of elephant flows often occupy a disproportionate share of the total bandwidth on a link.

a. [1 point] Give one example of an application/application level protocol/Internet use case that generates an elephant flow.

Video OR large file ftp OR other large transfers
+1 for mentioning a large transfer
No credit for simply saying 'something large'

b. [1 point] Give one example of an application/application level protocol/Internet use case that generates a mouse flow.

Loading HTML OR email OR other small transfers
+1 for mentioning a small transfer

Partial credit for things that might be small transfers, but with no clear justification (video games)
No credit for simply saying ‘web browsing’ since Youtube is web browsing & tons of pages have large video ads. Anything suggesting the website or content in it was small, however, was given (at least some) credit.
No credit for saying HTTP, since any file can be fetched via HTTP

We will now consider how TCP treats mice relative to elephants. When we discussed fairness in class, we assumed that both flows were large (elephants). Now we will consider whether TCP is fair when elephants and mice compete.

c. [3 points] Give one way in which TCP (inadvertently?) favours elephants over mice. Explain in terms of specific aspects of how TCP works. [Hint: you might consider one sender sending an elephant flow and another sender sending a series of mice flows (in different connections) that combine to the same size as the elephant flow.]

d. [3 points] give a second way in which TCP favours elephants over mice. Explain in terms of specific aspects of how TCP works.

e. [3 points] give a third way in which TCP favours elephants over mice. Explain in terms of specific aspects of how TCP works.

1. Initial handshake contributes larger fraction to overall time cost for short flows, so they pay a larger startup overhead
2. Loss recovery can be more costly for short flows because of lack of established ACK clocking
3. Loss recovery can be more costly for short flows because of limited retransmission triggers, e.g. possibly not enough data for fast retransmits or forward retransmits
4. Loss recovery can be more costly for short flows because no RTT sampling and therefore starting with large initial RTO value
5. Initial slow start phase contributes larger fraction to overall time cost (need to grow congestion window)
6. parallel connections deal better with larger files.
7. TCP flows with large window sizes fill up router buffers, causing mice flows to have (on average) larger queueing delays than elephants. Related idea is that TCP is only fair in steady state, whereas mice flows only exist in the transient (unfair) state.

(c~e): providing a correct answer with vague explanation +1.5, if with correct explanation in terms of how TCP works +3
Mentioning header overhead is not a good answer, consider that a single flow with 1.4k-byte data is also a mouse. Only +1

LAST PAGE