CS 536 : Sample Midterm

CS 536 Fall 2008 **Name:** ______________________
by writing my name I swear by the honor code

Problem 1	25	
Problem 2	25	
Problem 3	15	
Problem 4	15	
Problem 5	20	
Total	100	

Read all of the following information before starting the exam:

- Please write your name in this page. If I do not see a name, I will assume you do not want the paper to be graded.
- This booklet should have 6 printed pages. Please verify you have all the pages before you begin.
- Please keep your written answers brief; be clear and to the point. There is absolutely no point rambling or writing true but irrelevant statements.
- On the other hand, all questions will have partial credit. So, if you think you have an intuition for the answer, please write it down. For numerical questions, appropriate justification is required for any credit. Just writing down a number is not sufficient.
- Please make appropriate assumptions as required. It is important to learn what are good assumptions and bad assumptions. If unclear, please ask the instructor before making a bad assumption. Bad assumptions usually will result in not obtaining full credit.
- One scrap page is included at the end of the question paper. If you want, you could use either the back of each page or that extra scrap page for writing your answer as well.
- The exam is closed book. The problems in the exam are usually thinking-oriented; so having the book is not going to help anyways.
- Good luck!

Problem 1 (25pts). Assorted Questions

- Consider a circuit-switched network with 10 links each with a bandwidth of 10 Mbps. How many users can the network support simultaneously if half of the users want 50 Kbps and half of them want 450 Kbps ? [5pts]

  Answer: 10 Mbps is the aggregate bandwidth that needs to be shared. Let x be number of users that want 10 Kbps. Then $x \times 50Kbps + x \times 450Kbps = 10Mbps$. Solving for x, we get $x = 10,000Kbps/500Kbps = 20$. So, total number of users is 40.

- Assume that there is only one router/switch between a source and destination. The bandwidth of the first link, i.e., from source to the router is R_1. The second link, i.e., from router to destination is R_2. What is the total end-to-end delay for a packet of length L. Ignore queuing, propagation and processing delays. [5pts]

  Answer: The end-to-end delay for a packet of length L will be equal to the transmission delay L/R_1 between the source and the router plus the transmission delay L/R_2 from the router to the destination.

- Let us suppose there is a new application in the Internet which requires reliability but does not care about out-of-order delivery. What is the appropriate transport layer protocol for the application ? Justify! [5pts]

  Answer: There are two options: First, we can use TCP. Since the application does not care, it will certainly not be a problem to use TCP. An alternative is to use UDP, but ensure that reliability be provided at the application layer. **Answering one or the other will get you half the points.**

- Suppose two services, say Web and Email are running on a given host. How does the operating system know which application to forward a given packet to. [5pts]

  Answer: This is a direct question. The operating system uses the port number as a means to distinguish between packets that belong to two different applications.

- Which of the two types of architectures—client-server or peer-to-peer—that is more scalable in terms of number of nodes that can be supported in the architecture ? Justify?[5pts]

  Answer: This is also a direct question. The p2p model is more scalable as peers can utilize their upload bandwidth to transmit bits of information directly to the peers. In the client-server model, the server needs to transmit all the bits which places a restriction on the number of simultaneous uploads the server can perform, thus hurting its scalability.

Problem 2 (25pts). **Web caching.** Web caching traditionally has been designed to overcome the inherent propagation delays in the Internet by storing frequently retrieved content at the local cache. To help Web caching, HTTP protocol uses the conditional GET which allows HTTP request messages to be tagged with "If-modified-since: " header line to fetch objects. Assume the institution's access link has a capacity of 2.5 Gbps. Also, assume that GET requests, conditional GET requests, conditional GET responses (for the case when there is no modification of the object) are of size 100 Bytes.

- Suppose there are two object sizes that are most popular for the users in an institution. First is a small object of size 1 KByte and the second is a 50 MByte object. Assume that normal HTTP GET requests are less than 100 Bytes and both these types of objects are requested with equal probability (0.5). If the average request rate is 10 per second, what fraction of the access link's bandwidth is utilized ? [10pts]

  Answer: Average request rate is 10/sec. So, total download rate $= 5 \times 1KByte + 5 \times 50MByte \approx 250MByte/second = 2Gbps$. Thus, utilization is $2Gbps/2.5Gbps = 80\%$. Here, the large downloads are overwhelming the download.

- Suppose we employ Web caching for the above scenario which has a cache hit ratio of 0.5 for objects. How much would the access link utilization be ? [10pts]

  Answer: Average request rate is 10/sec out of which 5 requests per second are satisfied by the local web cache. So, total download rate $= 2.5 \times 1KByte + 2.5 \times 50MByte \approx 125MByte/second = 1Gbps$. In addition, there is an extra 100 Byte download whenever there is a cache hit for the conditional GET message that would add an additional $10 \times 0.1KBytes \times 0.5 = 0.5KBytes/second$, which can be ignored as it is really small compared to the main downloads. Thus, utilization is $1Gbps/2.5Gbps = 40\%$. Here still, large downloads occupy the bulk of the download.

- What if the cache hit rate for the small objects is 0.7 and that of the large objects is 0.1. Is it still beneficial to perform Web caching ? [5pts]

  Answer: Average request rate is 10/sec out of which 5 requests per second are for small objects and 5 requests for the large objects. Thus, download rate for small objects is $5 \times 1KByte \times (1 - 0.7) + 5 \times 100Byte \times 0.7 \approx 0.6KBytes$. For the large objects, it is $5 \times 50MByte \times (1 - 0.1) + 5 \times 100Byte \times 0.1 \approx 225MByte/second = 1.8Gbps$. It does not make sense to perform Web caching any more. Note that the 100 Byte extra download for the case when there is a cache miss is due to the fact that the Web cache makes a conditional GET request which is of size 100 bytes.

Problem 3 (15pts). Peer-to-peer network. Consider a p2p network which is used for file distribution and it has a hierarchy of peers, super peers and ultimate peers with each super peer responsible for 100 peers and each ultimate peer responsible for 100 super peers. The query propagation follows strict hierarchy, going all the way from the node to the super and then to the ultimate peer the super node is connected to. After that, the ultimate peer will contact other ultimate peers sequentially, until an answer is found. All ultimate peers are connected to each other.

- How many total nodes exist in a network with 5 Million peers ? [5pts]

  Answer: The total number of super peers is 5 Million/100 = 50,000. The total number of ultimate peers is 50,000/100 = 500. Total number of nodes = $5,000,000 + 50,000 + 500 = 5,050,500$.

- Suppose that a query is answered by peers with a probability of 0.1, super peer with a probability of 0.2 and a ultimate peer with a probability of 0.5. What is the average number of messages required to answer a query ? [10pts]

  Answer: The expected number of messages required can be obtained by first estimating the probability that x query messages are required.

$$
\begin{aligned}
P(x = 1) &= 0.2 \\
P(x = 2) &= (1 - 0.2) \times 0.5 = 0.4 \\
P(x = 3) &= (1 - 0.2) \times (1 - 0.5) \times 0.5 = 0.2 \\
P(x = 4) &= (1 - 0.2) \times (1 - 0.5)^2 \times 0.5 = 0.1 \\
P(x = 5) &= (1 - 0.2) \times (1 - 0.5)^3 \times 0.5 = 0.05 \\
P(x = 6) &= (1 - 0.2) \times (1 - 0.5)^4 \times 0.5 = 0.025
\end{aligned}
$$

  Average number of messages = $\sum_{k=1}^{\infty} k \times P(x = k)$. Since, the later terms will become very small. We can ignore after say $P(x = 6)$. Thus, the number of messages is $0.2 + 0.8 + 0.6 + 0.4 + 0.25 + 0.125 \approx 2.375$.

Problem 4 (15pts) Reliable Data Transfer Protocols.

- Remember a go-back-N (GBN) receiver. All it does is, on receipt of a packet that is in-order it will deliver the data. In all cases, it will acknowledge the packet that is the last in-order packet that has arrived. Suppose there is a host that runs a go-back-N receiver wants to communicate with a TCP sender. Would that work ? Justify [5pts]

  Answer: The TCP receiver buffers out-of-order segments, in contrast to a GBN receiver. Although buffering is usually beneficial, it is not absolutely essential, as the TCP sender will re-send any packets that have not been acknowledged.

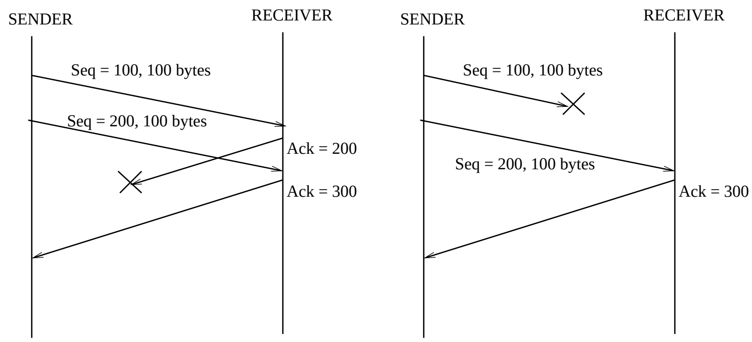
- Remeber a selective-repeat receiver. It selectively acknowledges the packets that it receives. Suppose now, that there is a host that runs a go-back-N receiver and a selective-repeat sender. Would that work ? Justify [5pts]

  Answer: No this will not work, since selective repeat uses selective acknowledgements while go-back-N uses cumulative acknowledgements.

- Consider a network which has a lot of bursty losses (several packets lost in a burst) on the wire. Which of the two protocols functions better in such scenarios—go-back-N or TCP or both ? Justify [5pts].

  Answer: Go-back-N responds quickly since after one packet timeout, the sender pretty much transmits all the segments which have been transmitted before. Whereas, TCP will retransmit only one segment after timeout. It will wait for the individual timeouts for all packets before it transmits the entire window.

Problem 5 (20pts) Different protocols, different scenarios. Assume sliding window parameter N=10 in all cases.



(a) Lost ack (b) Lost packet

- Consider scenario in Figure 1(a). Suppose this scenario where an ack from the receiver is lost, happens 10% of the time. Assume that each ack is 40 Bytes and each packet is 100 Bytes. Compare the two protocols, GBN and SR, in terms of the average number of packet transmissions in both directions when transferring a file of size 1 MByte. [10pts]

  Answer: The total number of packets is 1 MByte/100 Bytes = 10,000. For each of the 10,000 packets, the receiver will generate exactly one ACK. Go-back-N uses cumulative acks, so it will incur no more than 10,000 bytes since the ACK losses are recovered by the next cumulative-ACK.

  On the other hand, for SR, since each ACK is not a cumulative ACK, on an average one extra transmission will happen for each lost ACK. Thus, number of extra packets that need to be sent will be $0.1 \times 10,000 = 1,000$, which will generate an extra 1,000 acknowledgements in the reverse direction. However, again out of these 1,000 acks, 10% will be lost. So, another extra 100 packets need to be sent, and another 10 to counter the lost acks within these 100 acks. So, the total number of extra packets that need to be sent is $1,000 + 100 + 10 + 1 = 1111$ packets.

- Consider the scenario in Figure 1(b). In this scenario, a packet is lost in the forward direction with a probability of 0.01. Compare the two protocols in terms of number of pkt transmissions for transferring a 1 MByte file. Assume that there is no loss in the acknowledgements. [10pts]

  Answer: In the forward direction, things are bit more complicated. For selective repeat, exactly 1% of lost packets need to be resent. That means an extra 100 packets. But again these packets may be lost, so that would require another 10 packet transmissions and so on. Thus, the total number of extra transmissions required for the selective repeat is 111.

  For GBN, this is a problematic scenario. For every lost packet, it needs to transmit upto N packets. Given a loss rate of 1%, the total number of extra packets is $N \times 100 = 1,000$. However, every time the entire window is transmitted, 1% of the packets might get lost within the window. Thus, again another $N \times 1\% \times 1000 = 100$ packets might be lost. And so on. The total number of transmitted packets would be 1110.

Scrap Page
(please do not remove this page from the test packet)