CS 5480/6480: Computer Networks – Spring 2012
Homework 2
Due by 9 AM MT on February 22nd 2012

Important:
- **No cheating will be tolerated.**
- **No extensions will be granted**

Total points for cs5480: 31
Total points for cs6480: 41

Question 1 (Reliable Data Transfer Protocol Design) *6 points***:** In the generic SR protocol, the sender transmits a message as soon as it is available (if it is in the window) without waiting for an acknowledgment. Suppose now that we want an SR protocol that sends messages two at a time. That is, the sender will send a pair of messages and will send the next pair of messages only when it knows that both messages in the first pair have been received correctly. Suppose that the channel may lose messages but will not corrupt or reorder messages. Design an error-control protocol for the unidirectional reliable transfer of messages. Give an FSM description of the sender and receiver. Describe the format of the packets sent between sender and receiver, and vice versa. If you use any procedure calls other than the ones used in class (for example, udt_send(), start_timer(), rdt_rcv(), and so on), clearly state their actions. Give an example (a timeline trace of sender and receiver) showing how your protocol recovers from a lost packet.

Question 2 (GBN & SR Protocols) *6 points*:

(a) *4 points***:** Consider the GBN and SR protocols. Suppose the sequence number space is of size *k*. What is the largest allowable sender window that will avoid the occurrence of problems such as that shown on slide 3.54 for each of these protocols?

(b) *2 points:* Answer true or false to the following questions and briefly justify your answer: (i) With the SR protocol, it is possible for the sender to receive an ACK for a packet that falls outside of its current window. (ii) With GBN, it is possible for the sender to receive an ACK for a packet that alls outside of its current window.

Question 3 (RTT Estimate/Timeout Computation) *5 points***:** Run the *ping* command to a destination at least 10 hops away. Collect the delay for 101 ping packets. Setting *(i)* the delay of the first ping packet (call it packet #0) to be the initial estimated RTT, *(ii)* the delay of the second ping packet (call it packet number #1) to be the first sample RTT, and *(iii)* the initial devRTT to 0, use the TCP formulae to find the estimated RTT, estimated RTT variation, and the timeout interval from the ping delay data for packet #1 to packet #100. Plot the sample RTT, the estimated RTT, and the timeout interval on the y-axis of a graph with the packet sequence numbers (1-100) on the x-axis. Please also print the destination host name and IP address to which the ping packets were sent, and the time of the day you collected the data.

Question 4 (TCP Congestion Control) *6 points***:** Assuming TCP Reno is the protocol experiencing the behavior shown in figure below, answer the following questions. In all cases, you should provide a short discussion justifying your answer. (*9/6 points* for each part)

a. Identify the intervals of time when TCP slow start is operating.

b. Identify the intervals of time when TCP congestion avoidance is operating.

c. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

d. After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?

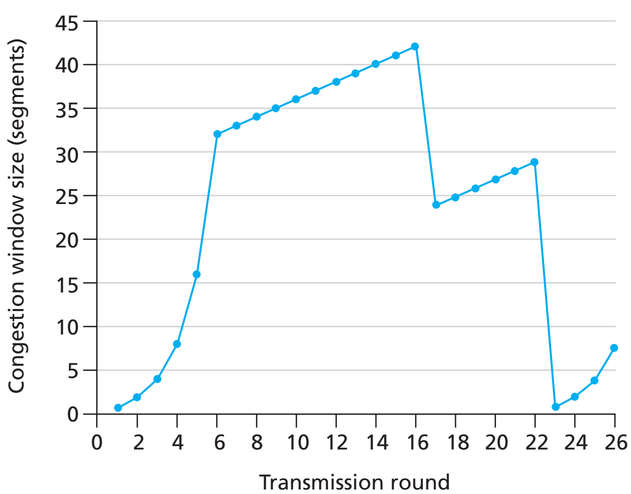
e. What is the value of ssthresh at the 18th transmission round?

f. During what transmission round is the 70th segment sent?

g. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of ssthresh?

h. Suppose TCP Tahoe is used (instead of TCP Reno), and assume that triple duplicate ACKs are received at the 16th round. What are the ssthresh and the congestion window size at the 19th round?

i. Again suppose TCP Tahoe is used, and there is a timeout event at 22^{nd} round. How many packets have been sent out from 17th round till 22^{nd} round, inclusive?



Question 5 (TCP Fairness) *3 points***:** Consider the plot used in the class to explain TCP fairness. Using this plot, explain why fairness cannot be achieved if MIMD or MIAD is used instead of AIMD.

Question 6 (Number of Transmissions) *5 points*:

(a) *1 point*: What is the average number of retransmissions necessary to reliably transmit a packet from a source to a destination when the network can drop packets independently with probability 0.2?

(b) *2 points*: What is the approximate number of transmissions necessary to reliably transmit a packet from a source to 10000 receivers when the network can drop packets independently for each receiver with a probability 0.2?
(c) *2 points*: What is the probability of at least one receiver not receiving a packet when it is multicast from a source to 10000 receivers and the when the network can drop packets independently for each receiver with a probability 0.02?

Question 7 (required for cs6480, extra credit for cs5480) *10 points***:**
Read the following paper: "*Fast and Robust Signaling Overload Control,*" Sneha K. Kasera et al, in the IEEE International Conference on Network Protocols, November 2001. Extended version is available from
http://www.cs.utah.edu/~kasera/myPapers/icnp2Ton.pdf.
Answer the following questions that are based on this paper.
(a) *4 points*: The overload control occupancy algorithm described in Section 4.1 of the paper uses a multiplicative increase and multiplicative decrease strategy. Does this strategy suffer from "fairness" issues discussed in the context of TCP congestion control? Rewrite the f_{n+1} equation so that an additive increase and multiplicative decrease (AIMD) strategy is followed with α being the maximum additive increment and β being the minimum multiplicative decrement.
(b) *2 points*: Is queue length a robust measure for system load? Explain.
(c) *4 points*: What is two-layer throttling? Justify the choice of r and d in Section 6. When the throttling fraction, $f = 0.5$, the cost of releasing a call, c_r, is 0.5, and the cost of dropping a call, c_d, is 0.1, what will be the reduction in cost of throttling a call using two-layer throttling. Suggest a different choice of r and d such that all the properties of a good solution (as stated in Section 6) are satisfied?