CSCE 416 Introduction to Computer Networks

Homework #3 solutions (10 points)

[1]

(a*) (2 points)* Suppose you have the following 2 bytes: 01011100 and 01010110. What is the 1s complement of the sum of these 2 bytes?

Answer: Adding the two bytes gives 10110010. Taking the one’s complement gives 01001101.

(b) Suppose you have the following 2 bytes: 11011101 and 00110110. What is the 1s complement of the sum of these 2 bytes?

Answer: Adding the two bytes

11011101

00110110

00010100

The one’s complement gives 11101011

(c) For the byte in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1a complement doesn’t change.

Answer: First byte = 01011110 ; second byte = 01010100.

[2] Consider the Stop-and-Wait protocol rdt3.0 and the sender side simply ignores (that is, takes no action on) all received packets that are either in error or have the wrong value in the ACK seq field of an acknowledgement packet. Suppose that the protocol is modified so that each time a frame is found in error, at either the sender or the receiver, the last transmitted frame is immediately resent.

Does the protocol still operate correctly? Explain why or why not. (Hint: consider what would happen if there were only bit errors; there are no packet losses but premature timeouts can occur. Consider how many times the nth packet is sent, in the limite as n approaches infinity.)

**Answer:**

The protocol would still work, since a retransmission would be what would happen if the packet received with errors has actually been lost (and from the receiver standpoint, it never knows which of these events, if either, will occur).

To get at the more subtle issue behind this question, one has to allow for premature timeouts to occur. In this case, if each extra copy of the packet is ACKed and each received extra ACK causes another extra copy of the current packet to be sent, the number of times packet is sent will increase without bound as approaches infinity.

[3*] (2 points)* A channel has a data rate of 8 Kbps (bandwidth) and a propagation delay of 20ms. For what frame size does stop-and-wait given an efficiency of at least 75%? Ignore processing time at the receiver.

**Answer**

**Applying the utilization equation, we get L = 960 bits.**



*[4].* Consider the GBN protocol with a sender window size of 3. Suppose that at time *t*, the next in-order packet that the receiver is expecting has a sequence number of *k*. Assume that the medium does not reorder messages. Answer the following questions:

(a). *(2 points)* What are the possible sets of sequence number inside the sender’s window at time t? Justify your answer

Answer: The senders window is of size 3 and begins somewhere in the range [k-3, k].

We have a window size of 3. The receiver has received packet k-1, and has ACKed that and all other proceeding packets.

--If all of these ACK's have been received by sender, then sender's window is [k, k+2].

--If none of the ACKs have been received at the sender, the sender's window contains k-1 and the N packets up to and including k-1. The sender's window is thus [k-3,k-1].

Thus, the sender’s window is of size 3 and begins somewhere in the range [k-3, k].

(b) *(2 points)* What are all possible values of the ACK field in all possible messages currently propagating back to the sender at time *t*? Justify your answer.

Answer: [k-4, k-1].

--If the receiver is waiting for packet k, then it has received (and ACKed) packet k-1 and the 2 packets before that.

--If none of those N ACKs have been yet received by the sender, then ACK messages with values of [k-3,k-1] may still be propagating back.

--Because the sender has sent packets [k-3, k-1], it must be the case that the sender has already received an ACK for k-4. Once the receiver has sent an ACK for k-4 it will never send an ACK that is less that k-4. Thus the range of in-flight ACK values can range from k-4 to k-1.

*[5]. (2 points*) Referring to the TCP’s algorithm of linear increase and multiple decrease, suppose that instead of a multiplicative decrease, TCP decreased the window size by a constant amount. Would the resulting algorithm converge to an equal share algorithm? Support your answer using diagrams similar to the one show below. (Hint, consider two cases: (1) the constant amount for both connections is the same, (2) the ratio of the linear decrease on loss between connection 1 and connection 2 is 2:1)

**Solution:**

No, it will not always converge to the equal share line. Refer to the figure below.

Case 1: In Figure (a), the ratio of the linear decrease on loss between connection 1 and connection 2 is the same - as ratio of the linear increases: unity. In this case, the throughputs never move off of the AB line segment.

Case 2: In Figure (b), the ratio of the linear decrease on loss between connection 1 and connection 2 is 2:1. That is, whenever there is a loss, connection 1 decreases its window by twice the amount of connection 2. We see that eventually, after enough losses, and subsequent increases, that connection 1's throughput will go to 0, and the full link bandwidth will be allocated to connection 2.

tcpFair

*[6]* Host A and Host B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 248. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 40 and 60 bytes of data, respectively. In the first segment, the sequence number is 249, the source port number 503, and the destination port number is 80. Host B sends an ACK whenever it receives a segment from Host A.

1. In the second segment sent from A to B, what are the sequence number, source port number, and destination port number?
2. If the first segment arrives before the second segment, in the ACK of the first arriving segment, what is the ACK number, the source port number, and the destination port number?
3. If the second segment arrives before the first segment, in the ACK of the first arriving segment, what is the ACK number?
4. Suppose the two segments sent by A arrive in order at B. The first ACK is lost and the second ACK Arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and ACKs sent. Assume there is no additional packet loss. For each segment in your diagram, show the sequence number and the number of bytes of data. For each ACK that you add, provide the ACK number.

**Answer:**

* 1. The sequence number is 289, source port number is 503 and destination port number is 80.
  2. The ack. number is 289, the source port number is 80 and the destination port number is 503.
  3. The ack. number is 249, indicating that it is still waiting for bytes 249 and onwards.

Host B

Host A

Seq = 249, 40 bytes

Seq = 289, 60 bytes

Ack = 289

Timeout interval

Ack = 349

Seq = 249, 40 bytes

Ack = 349

Timeout interval