CS 118 Homework 3

Wenxuan Liu 805152602

Problem 1

The protocol will still work. Because If there were only bit errors, the receiver will not send acknowledgement to the sender. Then, the sender will retransmit the packet after time out, so packet would not be lost or incorrect in the end.
When a packet is sent several times and the repetition is approaching infinity, however, the sender will have great overhead, because it keeps sending a same packet to the receiver and the remaining packets have to wait for this packet to finish. Then this method is very inefficient.

Problem 2

(a) No, a NAK-only protocol would not be preferable when the sender sends data only infrequently. Because if there is a packet loss when the sender sends the packet to the receiver, the receiver would only be notified of such loss after the lost packet's next packet has arrived with unordered seq number, and then sends back a NAK to the sender. When the time gap between the two packets is long given that the data is sent infrequently, it would be a long time to be notified and resend the lost packet and recover the problem. Comparatively, if a packet is lost under an ACK protocol, the sender would not receive an ACK before time out, so it would immediately resend the lost packet after time out, instead of waiting for the NAK that the receiver sent when it received the second packet. Thus, Using NAK would be inefficient.

(b) Yes, a NAK-only protocol would be preferable when the sender has a lot of data and there are few losses. Because when the sender send data frequently, the sender would know the situation of packet loss or errors when it receives a NAK right away given that the loss packet's next packet arrives at the receiver very fast and can check the sequence number in short period of time. Besides, if there are few losses, then few NAK would be sent to the sender, and the sender would seldomly resend the lost packets. More importantly, the sender doesn't have to receive a lot of feedback from receiver or wait for an ACK to move forward the window, but can consistently sending packet to the receiver, which can greatly increase the efficiency.

Problem 3

(a) When the receiver is expecting a packet with sequence number of k, the last six ACKs have sequence number k-1, k-2, k-3, k-4, k-5, k-6. If all of the packets have been properly received with ACKs sending back to sender, the senders window would be moved forward to [k, k+1, k+2, k+3, k+4, k+5] at time t. If none of the packets have been received, the senders window would still be [k-6, k-5, k-4, k-3, k-2, k-1]. Besides, all the other possible sender's windows are in between the two edge cases such that some of the packets before k has not received the ACK and some packets after k-6 have received ACK. Then, all the possible sender's window are:
[k-6, k-5, k-4, k-3, k-2, k-1],
[k-5, k-4, k-3, k-2, k-1, k],
[k-4, k-3, k-2, k-1, k, k+1],
[k-3, k-2, k-1, k, k+1, k+2],
[k-2, k-1, k, k+1, k+2, k+3],
[k-1, k, k+1, k+2, k+3, k+4],

[k, k+1, k+2, k+3, k+4, k+5].

(b) Since the receiver is expecting a packet with sequence number of k, the packet with sequence number k-7 must have ACK sent to sender so that the sender's window has been at least moved forward to [k-6, k-5, k-4, k-3, k-2, k-1] and receiver could receive all of them and possibly waiting for k. Also, k has not been received by the receiver, so k and its later packets cannot have AKCs in the sender. Thus, all the packets between the edge cases, which are k-6, k-5, k-4, k-3, k-2, k-1, are the possible values of ACK field currently propagating back to the sender.

Problem 4

(a) The sender will not be kept busy. First, the transmission delay is $\frac{4000*8bits}{8*10^6 bits/sec} = 4ms$. Since the sender window size is 5, there could be 5 packets sent the receiver. The RTT is two times of the one-way propagation delay, which is 2*20msec = 40 msec. So the efficiency is $\frac{(5*L/R)}{\boldsymbol{RTT+L/R}} = \frac{5*4msec}{40msec+4msec} = \frac{20msec}{44msec} = 0.45$. So the sender will not be kept busy.

(b) To achieve the full utilization, the efficiency would be equal or greater than 1. Let the sender window size be n. Then, the efficiency is $\frac{(n*L/R)}{\boldsymbol{RTT+L/R}} = \frac{n*4msec}{40msec+4msec} = \frac{n*4msec}{44msec}$≥1, then n≥11. Thus, the minimum sender window size is 11. Then, since 2^4 = 12 > 11, and 2^5=32>2*11, we need at least 4 bits for the sequence number field if we use GBN, and we need at least 5 bits if we use SE.

Problem 5

(a) True. Because if a packet A has been sent to the receiver and hasn't get corresponding ACK for a long time, it will time out and A will be sent again. Then, at this time, the previous sent packet's ACK could successfully arrive at the sender, so the sender's window would move forward with this packet A out of the window. Then, when the second sent packet's ACK arrives later, it will find that the ACK for A falls outside of sender's current window. Thus, premature timeout will lead to situation like this.

(b) True. It is similar to SR, such that when a packet is sent to receiver and timed out, the sender will resend the packet. However, if the previous sent packet wasn't lost or broken, the ACK would arrives at the sender now and move forward the window with this packet be excluded. Then, when the second sent packet's ACK arrives at the sender, it will falls outside of the current window. Such situation is also caused by the premature timeout.

(c) True. Because the SR with window size of 1 for sender and receiver would ack the same as Stop-and-Wait protocol by sending a packet from sender to receiver one at a time, and only send the next one after receiving the ACK at the sender. Also, if the sender doesn't receive the ACK before timeout, it would resend the packet to the receiver, so the functionalities for them are the same.

(d) True. Because GDN was designed at the time that memory was very expensive, so GDN would need the sender to resend all of the packets after the one that didn't get ACK accumulatively. Comparatively, SR would only need the sender to resend the packets that didn't get ACKs, so much less packets would be resent, and network communication cost is reduced for SR. Also, SR would store the packets after the packet that didn’t come to the receiver with the proper sequence number, while GBN doesn’t store them in the buffer. So additional memory would be used for SR.