44100113: COMPUTER NETWORKS
HOMEWORK 2: CHAPTER 3 Transport Layer
SOLUTIONS

Notes: All exercises are in accordance with the 6th edition (International Edition). We change data values in some problems, which are highlighted.

Exercise 1 (P5)

No, the receiver cannot be absolutely certain that no bit errors have occurred. This is because of the manner in which the checksum for the packet is calculated. If the corresponding bits (that would be added together) of two 16-bit words in the packet were 0 and 1 then even if these get flipped to 1 and 0 respectively, the sum still remains the same. Hence, the 1s complement the receiver calculates will also be the same. This means the checksum will verify even if there was transmission error.

Exercise 2 (P12)

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 n is sent will increase without bound as n approaches infinity.

Exercise 3 (P22)

(a). Here we have a window size of N=4. Suppose the receiver has received packet k-1, and has ACKed that and all other preceding packets. If all of these ACK's have been received by sender, then sender's window is [k, k+N-1]. Suppose next that none of the ACKs have been received at the sender. In this second case, the sender's window contains k-1 and the N packets up to and including k-1. The sender's window is thus [k-N,k-1]. By these arguments, the senders window is of size 4 and begins somewhere in the range [k-N,k].

(b). If the receiver is waiting for packet k, then it has received (and ACKed) packet k-1 and the N-1 packets before that. If none of those N ACKs have been yet received by the sender, then ACK messages with values of [k-N,k-1] may still be propagating back. Because the sender has sent packets [k-N, k-1], it must be the case that the sender has already received an ACK for k-N-1. Once the receiver has sent an ACK for k-N-1 it will never send an ACK that is less that k-N-1. Thus the range of inflight ACK values can range from k-N-1 to k-1.

Exercise 4 (P24)

(a). True. Suppose the sender has a window size of 3 and sends packets 1, 2, 3 at t0. At t1 (t1 > t0) the receiver ACKS 1, 2, 3. At t2 (t2 > t1) the sender times out and resends 1, 2, 3. At t3 the receiver receives the duplicates and re-acknowledges 1, 2, 3. At t4 the sender receives the ACKs that the receiver sent at t1 and advances its window to 4, 5, 6. At t5 the sender receives the ACKs 1, 2, 3 the receiver sent at t2. These ACKs are outside its window.

(b). True. By essentially the same scenario as in (a).

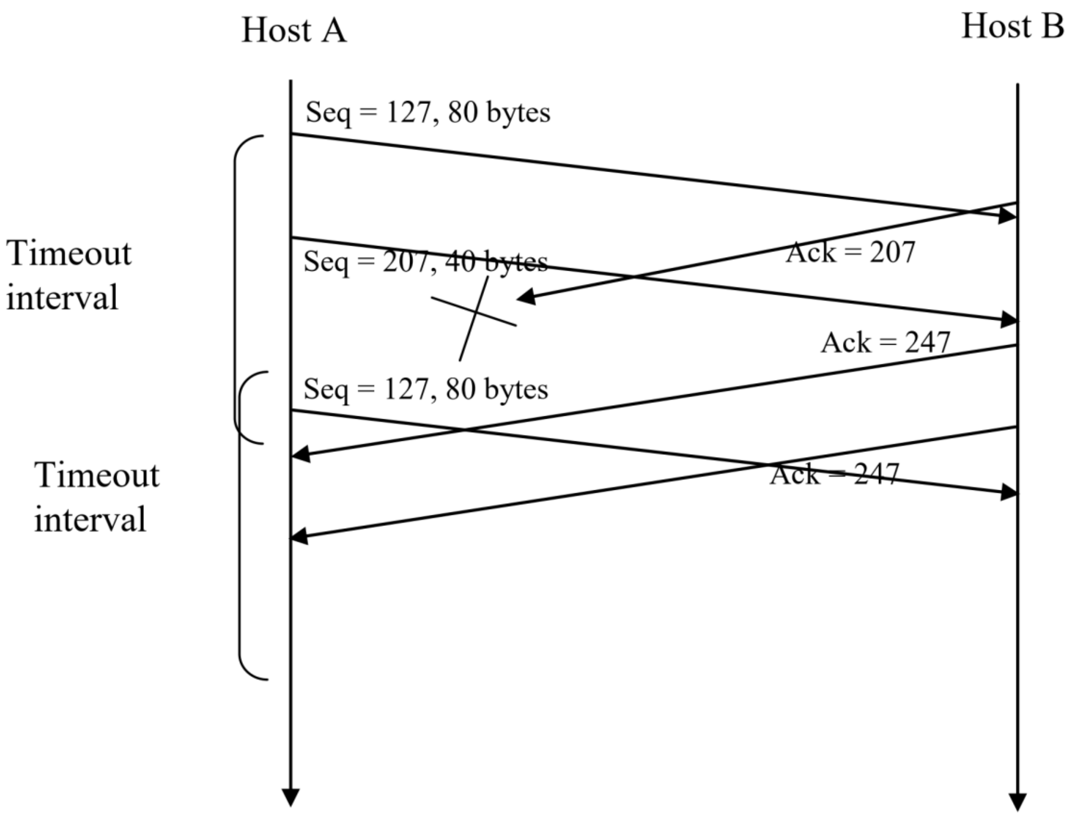
(c). True

(d). True. Note that with a window size of 1, SR, GBN, and the alternating bit protocol are functionally equivalent. The window size of 1 precludes the possibility of out-of-order packets (within the window). A cumulative ACK is just an ordinary ACK in this situation, since it can only refer to the single packet within the window.

Exercise 5 (P26)
There are $2^{32} = 4294967296$ possible sequence numbers.
(a). The sequence number does not increment by one with each segment. Rather, it increments by the number of bytes of data sent. So the size of the MSS is irrelevant -- the maximum size file that can be sent from A to B is simply the number of bytes representable by $2^{32} \approx 4.19\ Gbytes.$

(b). The number of segments is $\left\lceil \frac{2^{32}}{536} \right\rceil = 8{,}012{,}999$. 66 bytes of header get added to each segment giving a total of $528{,}857{,}934$ bytes of header. The total number of bytes transmitted is $2^{32} + 528{,}857{,}934 = 4.824 \times 10^9$ bytes.
Thus it would take 249 seconds to transmit the file over a 155 Mbps link.

Exercise 6 (P27)
(a). In the second segment from Host A to B, the sequence number is 207, source port number is 302 and destination port number is 80.
(b). If the first segment arrives before the second, in the acknowledgement of the first arriving segment, the acknowledgement number is 207, the source port number is 80 and the destination port number is 302.
(c). If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, the acknowledgement number is 127, indicating that it is still waiting for bytes 127 and onwards.
(d).



Exercise 7 (P44)

(a). It takes 1 RTT to increase CongWin to 7 MSS; 2 RTTs to increase to 8 MSS; 3 RTTs to increase to 9 MSS; 4 RTTs to increase to 10 MSS; 5 RTTs to increase to 11 MSS; and 6 RTTs to increase to 12 MSS.

(b). In the first RTT 6 MSS was sent; in the second RTT 7 MSS was sent; in the third RTT 8 MSS was sent; in the fourth RTT 9 MSS was sent; and in the fifth RTT, 10 MSS was sent. Thus, up to time 5 RTT, 6+7+8+9+10 = 40 MSS were sent (and acknowledged). Thus, we can say that the average throughput up to time 5 RTT was (40 MSS)/(5 RTT) = 8 MSS/RTT.