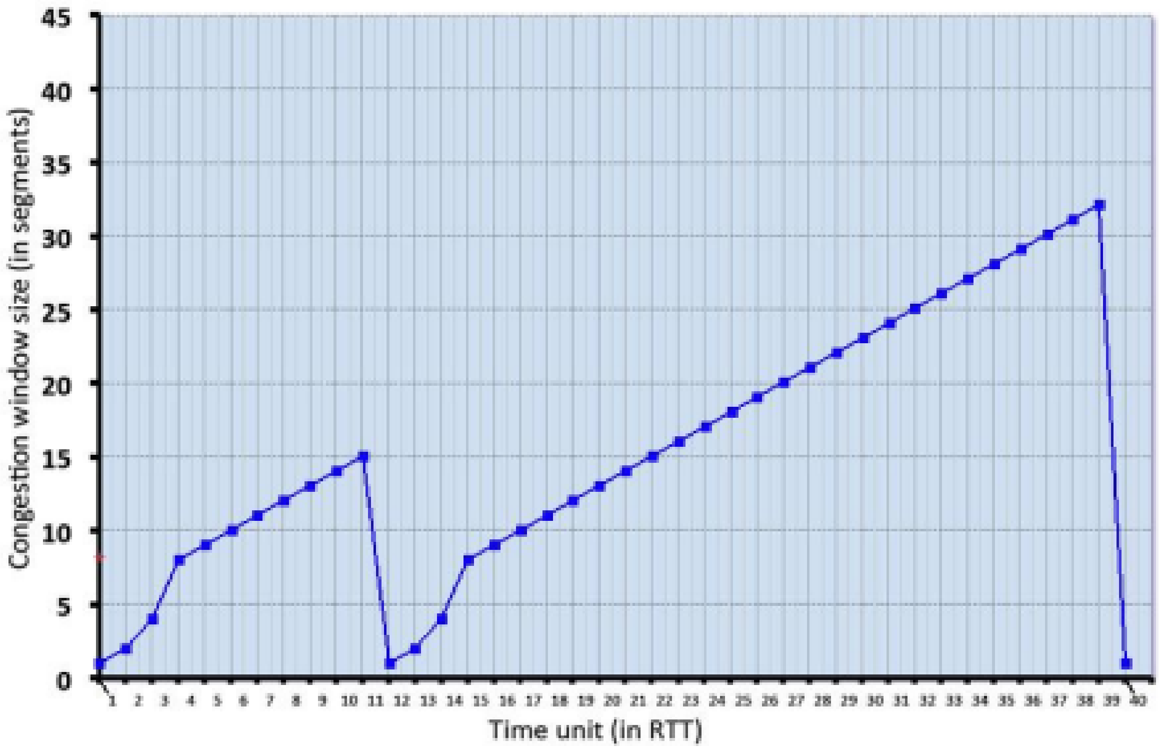
Sean Xia

Homework 2
CSE 310 Fall 2022
Due date: October 13, 2022; 11:59 PM
Submission via Blackboard.

1. **TCP in action! (20 points)**
   The figure below plots the evolution of TCP's congestion window at the beginning of each time unit (where the unit of time is equal to the RTT). Consider the evolution of TCP's congestion window in this example and answer the following questions. The initial value of cwnd is 1 and the initial value of ssthresh (shown as a red +) is 8.



   (Format your answer as: 1, 3, 5, 9. For larger ranges > 10, you can list a range like "10-20" instead of listing all numbers.)

   a. **Give the times at which TCP is in slow start.** TCP is in slow start at times t = 1, 2, 3, 12, 13, 14, 40. We can tell because it will be in slow start at the beginning and after any timeouts, shown by sudden drops to cwnd = 1, until the cwnd goes above the current ssthresh. The ssthresh = 8 in range [1, 11], ssthresh is about 8 in range [12, 39], and the ssthresh is around 16 at time t = 40 since ssthresh approximately halves for every loss.

b. **Give the times at which TCP is in congestion avoidance.** TCP is in congestion avoidance at times t = 4, 5, 6, 7, 8, 9, 10, 11, [15, 39]. When cwnd is above the current ssthresh listed above in answer 1a, it will be in congestion avoidance.

c. **Give the times at which packets are lost via timeout.** Packets are lost via timeout at times t = 11, 39. When cwnd suddenly drops so cwnd = 1, we know a timeout loss has happened.

d. **Give the times at which packets are lost via triple ACK.** Packets are lost via triple ACK at times t = {}, or empty set because no packets are lost via triple ACK. If a packet loss via triple ACK happened, we would have seen the cwnd drop to about half of its previous cwnd however all the drops leave the cwnd = 1 which indicates a timeout loss.

e. **Give the times at which the value of ssthresh changes (if it changes between t=3 and t=4, use t=4 in your answer).** The value of ssthresh changes at times t = 12, 40. The ssthresh changes whenever there is a loss in the diagram.

2. **Transport Layer protocols. Give two scenarios where UDP may be preferred over TCP. Explain why. (5 points)**
Two scenarios where UDP may be preferred over TCP are for online multiplayer video games and video streaming. The reason why UDP might be preferred over TCP is because the person choosing between UDP and TCP might be willing to trade reliability for speed. TCP, although reliable, is slower than UDP because of the need to establish a connection and perform error checks. UDP on the other hand will just send out data without caring if the receiver got the exact message or not. In online multiplayer video games such as first person shooters, or FPS games, we may prioritize speed and TCP is slower than UDP, and by the time it would take TCP to do error checking and send the information over again, the information may have become useless or redundant because the moment that the information was needed passed already. In the case of video streaming, even if a frame of information is skipped, it is likely that the person may not perceive it so it is okay to miss a little information here and there and instead prioritize the speed of sending the video to the receiver.

3. **True or false? Please provide a reasoning (1-2 sentences) for your answer. (10 points)**

   a. **The size of the TCP rwnd never changes throughout the duration of the connection.** False, the TCP rwnd or receive window, is the variable included in the TCP header that states how much data that the one who sent their TCP header can eventually receive. Another way it can be thought of is how much free space there is in the buffer. Both sides of a TCP connection will initialize their rwnd size during the 3-way handshake, but if either side is not able to handle the load that it is receiving, it can adjust the rwnd size that they advertise to each other so the data transfer rate can be adjusted to fit the new rwnd. The rwnd can even be set to 0 to stop data transfer until the buffer can be cleared up.

b. **The TCP segment has a field in its header for rwnd.** True, the TCP segment has a field in its header for rwnd. The rwnd generally is labeled “window” or “window size” and it is right before the “checksum” section in TCP header diagrams and has a reserved space of 16 bits.

c. **Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.**
False, the acknowledgement number in the same segment would not be 42. The acknowledgement number would be 42 from Host B but not in the same segment. However, in the same segment, the acknowledgement number is not related to the sequence number and and the number of bytes of the same segment.

d. **Host A is sending Host B a large file over a TCP connection. Assume Host B has no data to send Host A. Host B will not send acknowledgments to Host A because Host B cannot piggyback the acknowledgments on data.**
False, Host B will still send acknowledgements of the data to Host A. Piggybacking is an attempt to optimize sending between two sides so that if one side needs to send back an acknowledgement as well as some data, they can be sent as one packet instead. When one side such as Host B does not have any data to send to Host A, it will still have to send an acknowledgement as its own packet. Just because the acknowledgement cannot piggyback on data being sent, it does not mean that Host B does not send an acknowledgement.

e. **Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m, then the sequence number for the subsequent segment will necessarily be m+1.**
False, the sequence number for the subsequent segment doesn’t have to necessarily be m + 1. The sequence number for the subsequent segment would be m + 1 if the current segment was only sending 1 byte because host B would need the data starting at byte m + 1. However, in reality, we can send multiple bytes in the current segment, and the sequence number for the subsequent segment would be m+(the number of bytes in the current segment).

4. Sequence Numbers and ACKs (15 points)

a. **(5 points) Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number 90; the second has sequence number 110.**

   i. **How much data is in the first segment?**
   The first segment would have all the bytes starting from sequence number of the first segment, 90, to, but not including the sequence number of the subsequent or second segment, which is 110. This means the first segment has bytes [90, 110) -> [90, 109], or 110 bytes - 90 bytes = 20 bytes of data in the first segment.

ii. **Suppose that the first segment is lost but the second segment arrives at B. In the acknowledgment that Host B sends to Host A, what will be the acknowledgment number?**
If the first segment is lost but the second segment arrives at B, the acknowledgement number that Host B sends to Host A will be 90 because it still needs to receive the first segment that has sequence number 90.

b. **(10 points) Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.**

i. **In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?**
The sequence number will be 127 + 80 = 207 and the source and destination port should not have changed from the first segment sent from Host A to Host B, so the source port number should be 302 and the destination port number should be 80.

ii. **If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?**
If the first segment arrives before the second segment, in the acknowledgement of the first arriving element, the acknowledgement number would be 127 + 80 = 207, the source port number would be 80 since it is coming from Host B, and the destination port number would be 302 since it is being sent to the port that Host A used to send the first segment to port B.

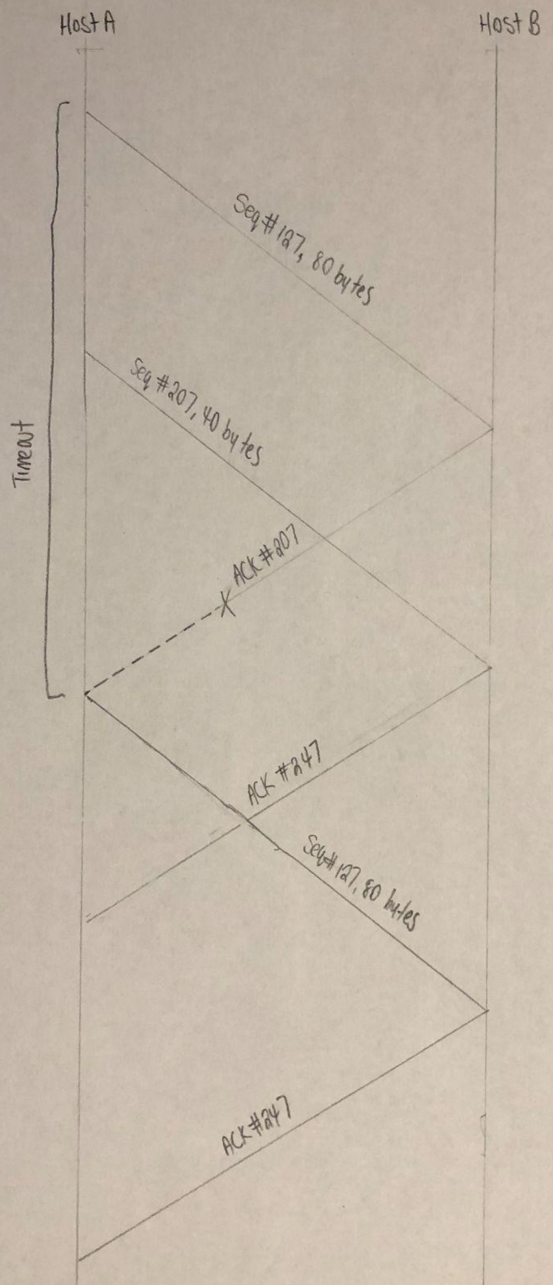
iii. **If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?**
If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, the acknowledgement number will be 127 because it is still waiting for the first segment containing bytes [127, 127 + 80) -> [127, 207) -> [127, 206].

iv. **Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost and the second acknowledgment arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgments sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes**

of data; for each acknowledgment that you add, provide the acknowledgment number.

Sean Xia 2022/10/13 CSE310

4.B.d



Host A
Host B
Timeout
Seq #127, 80 bytes
Seq #207, 40 bytes
ACK #207
ACK #247
Seq #127, 80 bytes
ACK #247