Problem 1

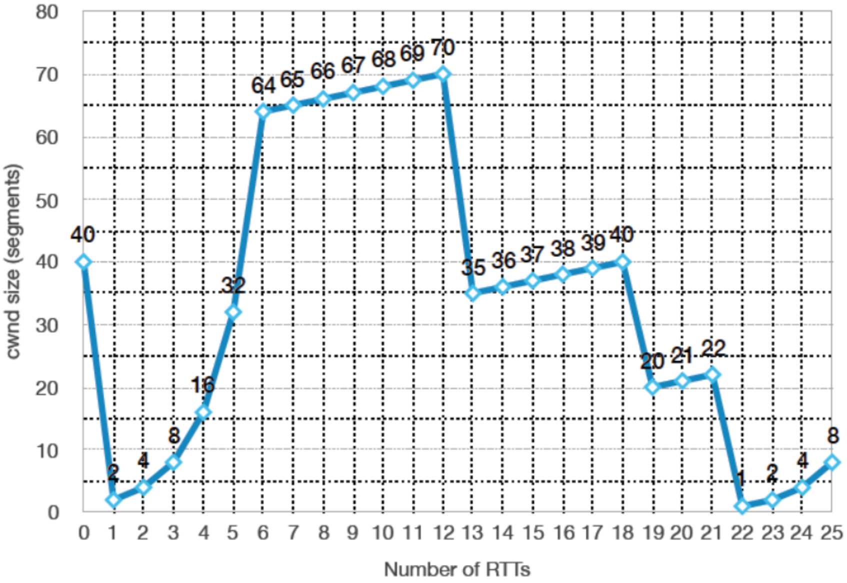

Figure 1: TCP congestion control.

Assume that a TCP connection has been running between hosts A and B for sometime, so that the number of RTTs shown in the above graph are with respect to the time when you started observing the the cwnd value of this connection. Hosts A and B use TCP Reno (with Fast Retransmit and Fast Recovery).

1. **On the graph above**, identify the time periods when TCP slow start is operating.
2. **On the graph above**, identify the time periods when TCP congestion avoidance is operating (AIMD).
3. For each loss event, specify whether it was detected by a triple duplicate ACK or by a timeout
4. For each loss event, indicate the value of the slow start threshold (ssthresh)

1. RTT1 - RTT6, RTT22 - RTT25

2. RTT6 - RTT12, RTT13 - RTT18, RTT19 - RTT21

3. Triple ACK: RTT12, 18
Timeout: RTT0, 21

4.RTT0: $40/2 = 20$
RTT12, 18: 20
RTT21: $22/2 = 11$

Problem 2

Assume that host A sets up a TCP connection with host B to send data. Assume that A's `ssthresh` value is 64 KB and TCP segment size is 3 KB.

1. How many bytes have been transmitted after 3 RTTs assuming no losses? Show your work.
2. Now, suppose that after the third RTT, a loss occurs which results in the TCP sender's retransmission timer to expire. What actions will TCP congestion control take in this case?
3. Assuming no further packet loss occurs from then on, how many RTTs does it take to transmit an additional 22 KB of data?

1. 3+6+12 = 21 KB

2. It will make the ssthresh to half the previous cwnd value, 32KB, and take the cwnd walut down to 1 MSS (3KB.) And it would continue slow start.

3.3+6+12+24 is greater than 22
Additional 4 RTT needed.

Problem 3

True or False? Briefly explain your answer in a single sentence.

1. 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.
2. The size of the TCP *rwnd* never changes throughout the duration of the connection.
3. Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.
4. 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.
5. The TCP segment has a field in its header for *rwnd*.
6. Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. The current value of TimeoutInterval for the connection will necessarily be $\geq$ 1 sec.
7. 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.

1. False. A is sending B data, then B needs to acknowledge.

2. FALSE. RcvWindow is used to give the sender an idea of how much free buffer space is available at the receiver. Thus as the amount of unacknowledged TCP data varies the RcvWindow also changes.

3. TRUE. TCP is not permitted to overflow the allocated receiver buffer. Hence when the sender can not send any more data RcvWindow would be 0 and hence all the buffer would have unacknowledged data.

4. False. If data was sent, the next sequence number would not be m+1.

5. True. From the header table it is true.

6. False. Depending on the value of or EstimatedRTT it may or may not be greater than 1.

7. False. The acknowledgement number has nothing to do with the sequence number. The ack. number indicates the next sequence number A is expecting from B.

Problem 4

As we have discussed in the class, a timer is a useful component in various protocol designs: because a communicating end cannot see what is going on either inside the network or at the other end, when needed it sets up an ”alarm”, and takes some action when the alarm goes off.

1. Does HTTP use any timers? If so, please briefly describe how each is used. If not, please explain why it does not need one.
2. Does DNS use any timers? If so, please briefly describe how each is used. If not, please explain why it does not need one.
3. Does TCP use any timer? If so, please briefly describe how each is used. If not, please explain why it does not need one.
4. Does UDP use any timer? If so, please briefly describe how each is used. If not, please explain why it does not need one.

1. HTTP does not use timer, because it is a pull. So the clients send requests to the server, if the clients don't get the reply they want, they can just request again.

2. Yes. DNS uses timer in the TTL field. The caching resolver can just access the previous result if time to live is still non-zero.

3. Yes. Timer is used for reliable transmission. When A sends data to B, A wants to make sure the data is delivered by getting an ACK from B. If ACK not received in a certain amount of time, A would send it again to make sure the data gets delivered. It is because TCP ensured in-order reliable transmission, so everything needs to be delivered.

4. UDP does not use timer. UDP is connection-less and out of order, so one piece of data missing won't affact the other part of the transmission.