ECE/CS 438: Homework 2 Solutions: Transport Layer, TCP, UDP

Fall 2018, UIUC

(I) Write True/False with a brief justification (around 1 sentence):

(a) The TCP ACKs consume bandwidth. It would be better if the TCP receiver only sends NACKs upon receiving out of order (or corrupted) packets.

Answer: False. If NACKs fail, reliability is violated.

(b) Assume TCP is in the Slow Start phase, with SSThreshold as 32. At some time instant, the congestion window is 16. When the congestion window increases the next time, it will become 32.

Answer: False. When the CW increases it will be 17 (16+1=17). If there is a timeout then the CW will be 2 (1+1=2) upon the next increment. If 3 DupACKs are received, CW will be 12 (16/2+3=11, 11+1=12) at the next increment.

(c) A TCP socket creates an end to end connection between two devices.

Answer: False. It creates a end to end connection between two processes.

(d) The TCP sender has packets 20 to 30 in its congestion window (CW), all waiting for ACKs, when a timeout occurs. The sender will cut down CW to 1 and will have to gradually retransmit each of these packets.

Answer: False. It may not retransmit all the packets since a cumulative ACK might arrive, making the TCP receiver skip one or more retransmissions.

(e) Consider the case where a TCP sender transmits a file to a TCP receiver. A system administrator tells you that during this transfer, no timeout or dupACKs were recorded. In that case, the congestion window during the transfer should have never decreased (i.e., it should either remain same or increase).

Answer: True. This means the network is not getting more congested, hence no reason to reduce congestion window.
Note: flow control does not change congestion window directly. The sender's rate is actually limited by the minimum of the flow control and congestion window.

(f) The lower end of the TCP transmitter's window is never greater than the lower end of the TCP receiver's window.

Answer: True. If transmitter base was greater than the receiver base, it would mean that the transmitter received an ACK for a packet that the receiver has not yet received. That's not possible.

(g) TCP can cope with any amount of losses, and hence, TCP should work without modifications on lossy wireless networks.

Answer: False. TCP is designed under the assumption that losses are due to congestion. If losses happen due to bit errors in the network, TCP will unnecessarily cut back on CW. You should get points if you mention something like: this statement is true because TCP will still work without modifications but the performance will be poor.

(h) The selective repeat receiver need not send an ACK if the received packet is less than its lower end of the (current) receive window.

Answer: False. The receiver needs to send an ACK since prior ACKs may not have been received by the transmitter.

(II) Answer the following questions on TCP – refer to Figure 1. Assume slow start, unless otherwise mentioned. Copy the timelines in your answer-sheet and work on them:

(a) In Figure 1(a), what are the values of CW at times t1, t2, t3? How should the TCP transmitter react after receiving A3 and A2.

(b) In Figure 1(b), assume TCP starts from CW=1. What should CW be at times t1 and t2? How should the TCP transmitter react after the timeout? How should the TCP transmitter react when each of the the last two ACKs (shown in the figure) arrive?

(c) In Figure 1(c), what should CW be at time t1? How should the TCP transmitter react upon receiving A13?

(d) In Figure 1(d), what should the CW be at times t1, t2, t3, and t4? How should the TCP transmitter react upon receiving each of the four ACKs?

(e) In Figure 1(e), say that the first ACK that is shown to arrive at the TCP transmitter is A5. What should the CW be at times t1 and t2? How should the TCP transmitter react upon receiving A5, when the timeout occurs, and upon receiving the last shown ACK (just before t2)?

(f) In Figure 1(f), LIT is the acronym for *linear increase threshold*. What should the values of CW be at times t1, t2, t3, and t4? How should the TCP transmitter react upon receiving each of the ACKs?

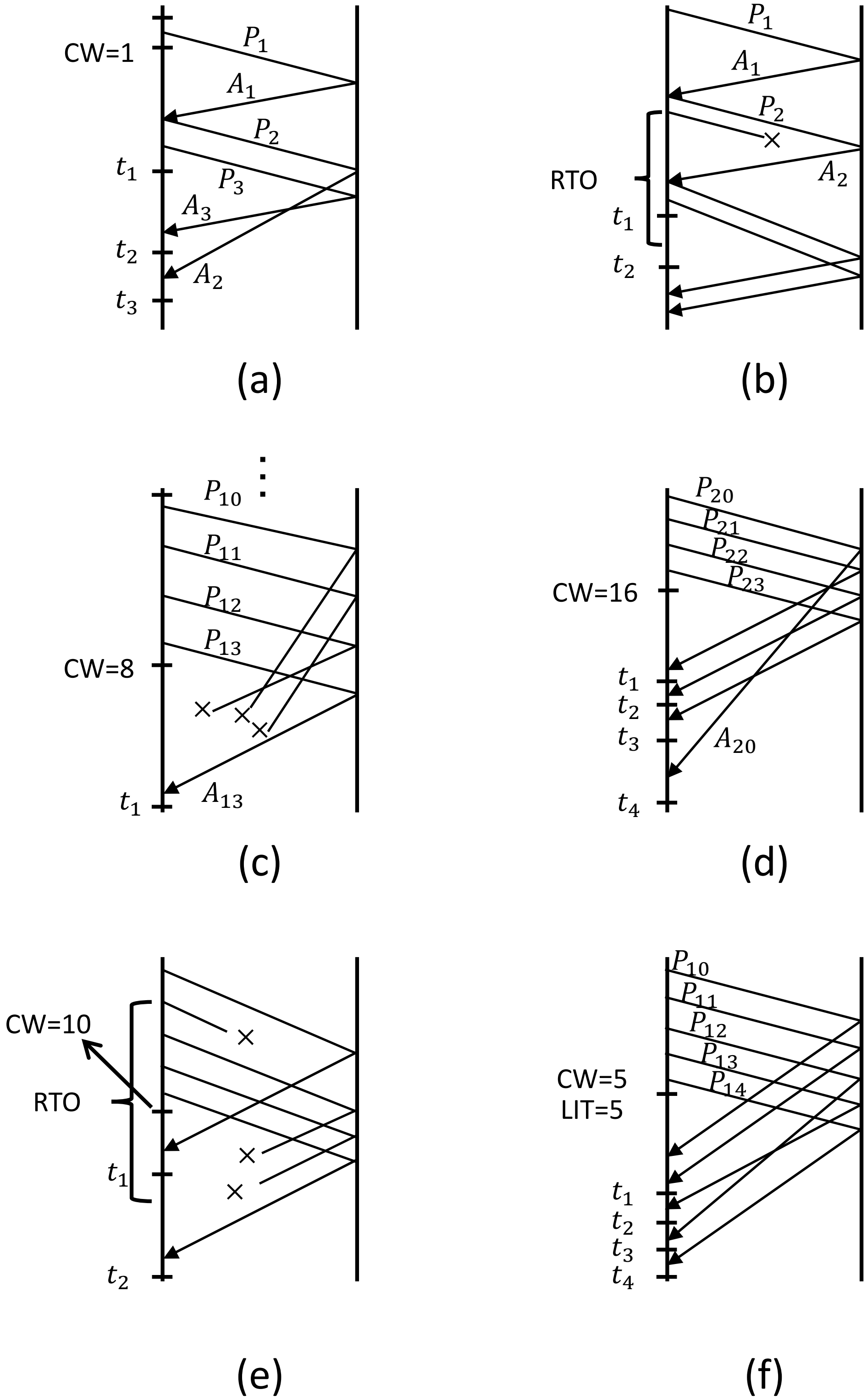
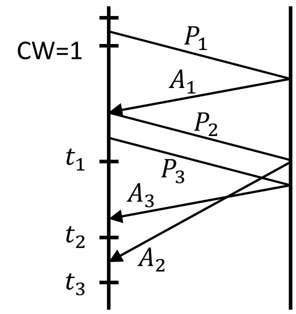


Figure 1: See question II

Answer: See Figure 2 and 3. Please directly draw on the figure.
Note: Please refer to Figure 4: TCP state transition diagram (copied from midterm review slides) for more details.

t_1: CW=2
t_2: CW=4 (because to the transmitter, A3 is a cumulative ACK which acknowledges two packets, P2 and P3)
t_3: CW=4 (because to the transmitter, P2 has already been acknowledged by A3)

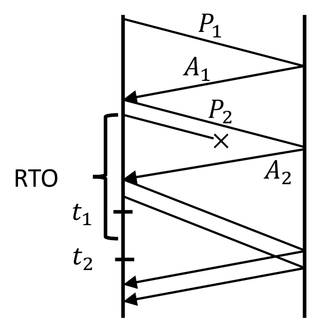
Receiving A3: Change CW to [4 5 6 7], send P4 - P7
Receiving A2: Don't do anything



(a)

t_1: CW=3 (CW adds 1 after receiving A2)
t_2: CW=1 (CW becomes 1 after timeout)

After timeout: Set threshold to 3/2=1.5, set CW to 1, retransmit P3
Receiving last two ACK (both A2): Don't do anything (wait for the 3rd DupACK)

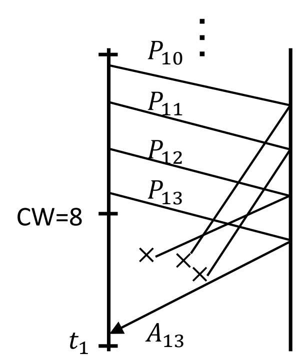


(b)

(Assume packets before P10 have already been acknowledged in the past)

t_1: CW=12 (because A13 acknowledges P10 - P13)

Receiving A13: Change CW to [14, 15 ... 25], send P14 - P25



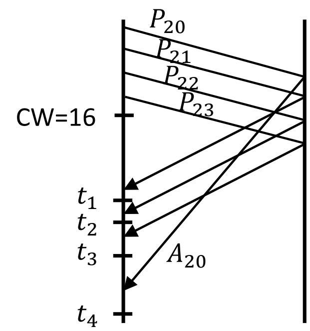
(c)

Figure 2: Solutions to question II (a-c)

(Assume packets before P20 have already been acknowledged in the past)

t_1: CW=18, t_2: CW=19, t_3: CW=20, t_4: CW=20

Receiving A21: Change CW to [22, 23 … 39]. Send P24 - P39
Receiving A22: Change CW to [23, 24 … 41]. Send P40 - P41
Receiving A23: Change CW to [24, 25 … 43]. Send P42 - P43
Receiving A20: Don't do anything

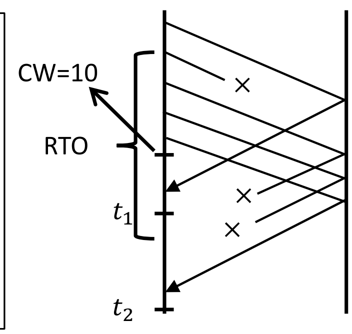


(d)

(Assume packets before P5 have already been acknowledged in the past)

t_1: CW=11, t_2: CW=1

Receiving A5: Change CW to [6, 7 … 16]. Send P10 - P16
After timeout: Set threshold to 11/2=5.5, set CW to 1, retransmit P6
Receiving last shown ACK: Don't do anything

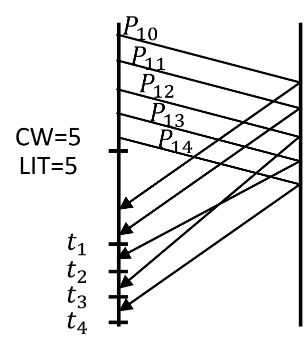


(e)

(Assume packets before P10 have already been acknowledged in the past)

t_1: CW=5.4, t_2: CW=5.8, t_3: CW=5.8, t_3: CW=6.0
(Because of linear increase)

Receiving A10: Send P15
Receiving A11: Send P16
Receiving A13: Send P17, P18
Receiving A12: Don't do anything
Receiving A14: Send P19, P20



(f)

Figure 3: Solutions to question II (d-f)

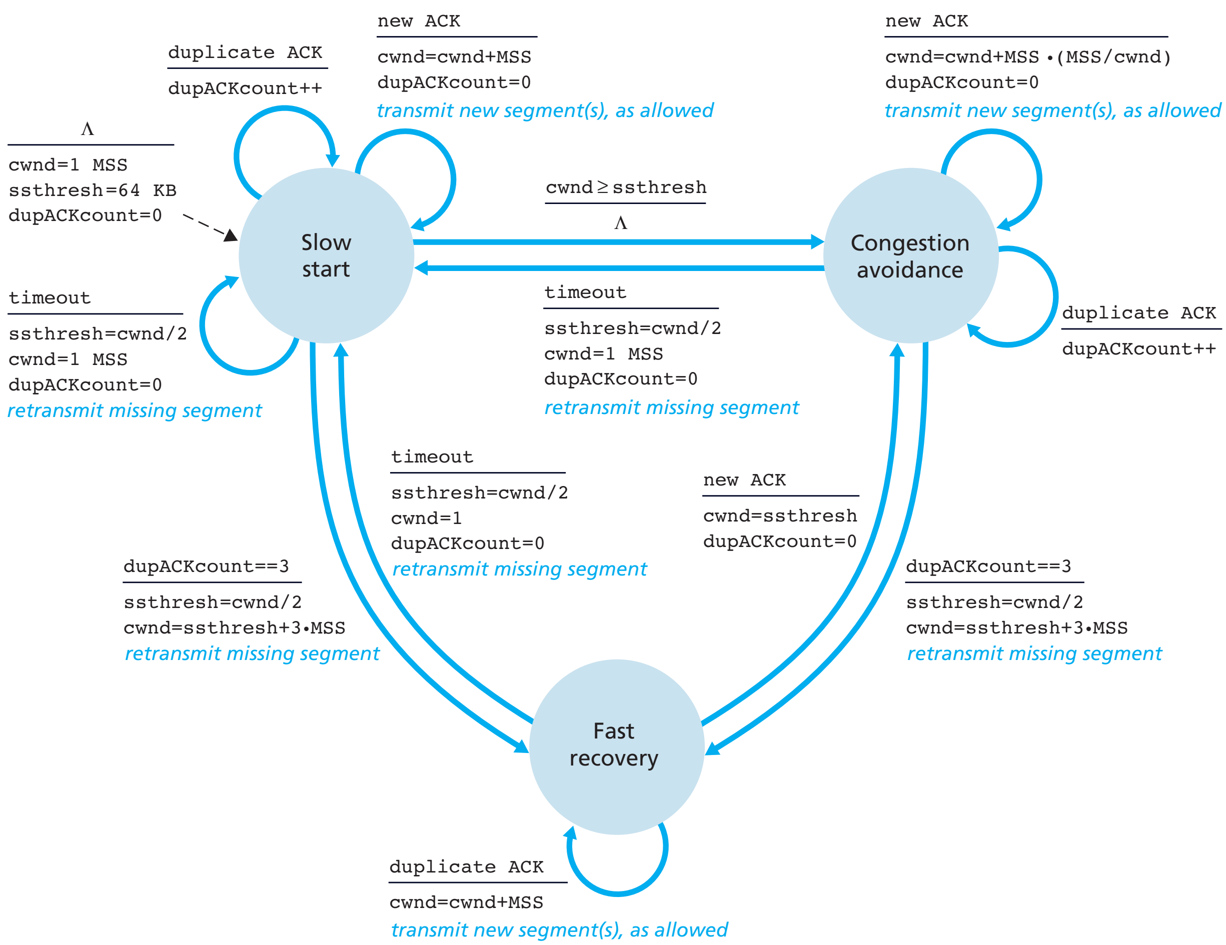


Figure 4: TCP state transition diagram.