# HW3 Solutions

### Problem 8

The sender side of protocol rdt3.0 differs from the sender side of protocol 2.2 in that timeouts have been added. We have seen that the introduction of timeouts adds the possibility of duplicate packets into the sender-to-receiver data stream. However, the receiver in protocol rdt.2.2 can already handle duplicate packets. (Receiver-side duplicates in rdt 2.2 would arise if the receiver sent an ACK that was lost, and the sender then retransmitted the old data). Hence the receiver in protocol rdt2.2 will also work as the receiver in protocol rdt 3.0.

### Problem 19

This problem is a variation on the simple stop and wait protocol (rdt3.0). Because the channel may lose messages and because the sender may resend a message that one of the receivers has already received (either because of a premature timeout or because the other

receiver has yet to receive the data correctly), sequence numbers are needed. As in rdt3.0, a 1-bit sequence number will suffice here.

The sender and receiver FSM are shown in Figure 3. In this problem, the sender state indicates whether the sender has received an ACK from B (only), from C (only) or from neither C nor B. The receiver state indicates which sequence number the receiver is waiting for.

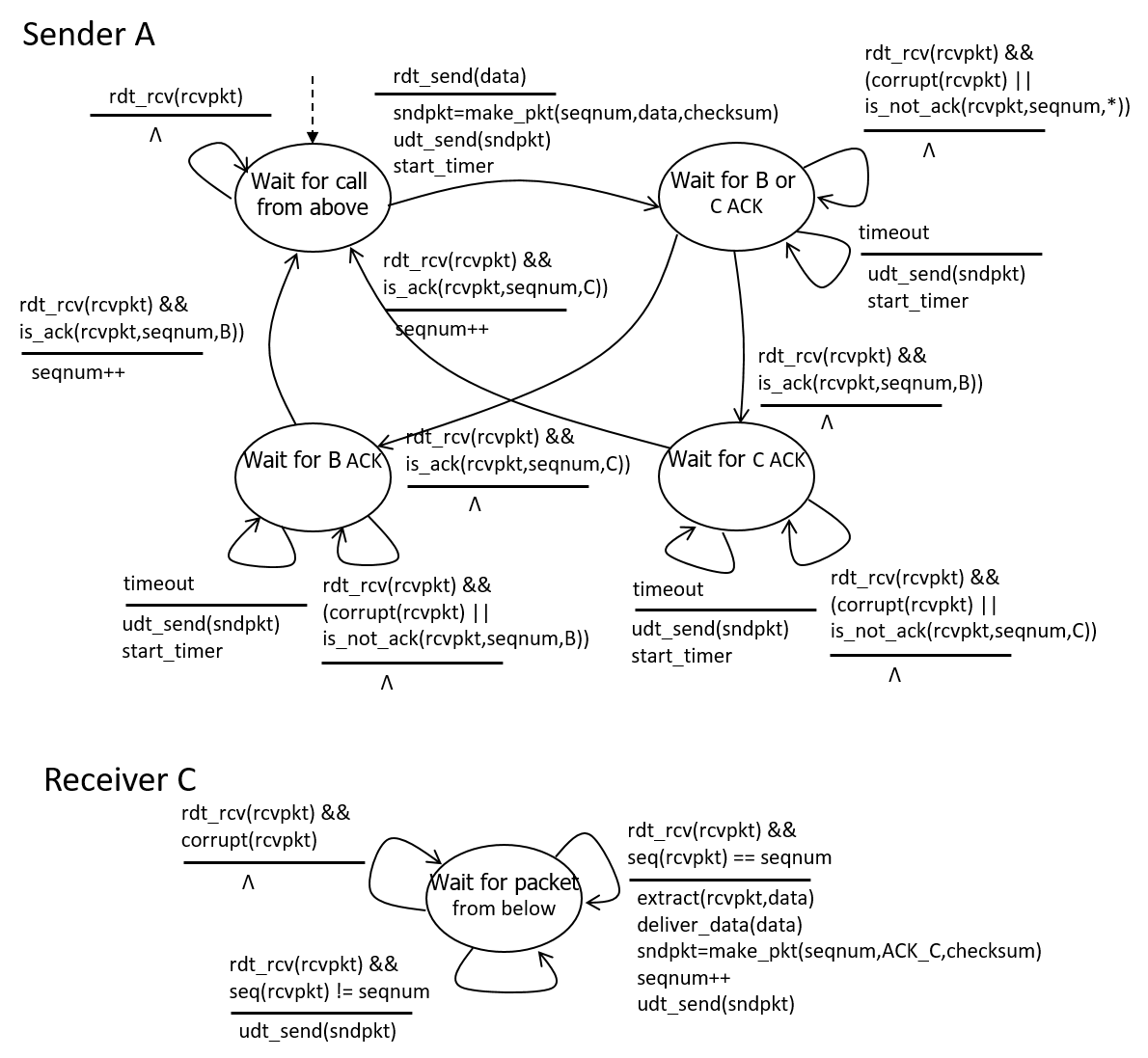
****

Figure 3. Sender and receiver for Problem 3.19 (Problem 19)

Note: Original solution that comes with the textbook has two critical errors: the sender does not send any new data from the application layer, and the receiver does not deliver any received in-order data to the application layer.

### Problem 20

The sender side FSM is exactly same as given in Figure 3.15 in text. The receiver side FSM is given below.



### Problem 27

1. 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.
2. 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.
3. 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.

Timeout interval

Ack = 247

Ack = 247

Seq = 127, 80 bytes

Seq = 127, 80 bytes

Seq = 207, 40 bytes

Ack = 207

Host A

Host B

Timeout interval

### Problem 32

a)

Denote  for the estimate after the *n*th sample.











b)





c)





The weight given to past samples decays exponentially.

Note: The above assumes that the first EstimatedRTT equals the first SampleRTT. If assuming an initial value of EstimatedRTT0, the formula needs to be modified as:  
.

### Problem 37

1. GoBackN:

A sends 9 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2, 3, 4, and 5.

B sends 8 ACKs. They are 4 ACKS with sequence number 1, and 4 ACKS with sequence numbers 2, 3, 4, and 5.

Selective Repeat:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2.

B sends 5 ACKs. They are 4 ACKS with sequence number 1, 3, 4, 5. And there is one ACK with sequence number 2.

TCP:

A sends 6 segments in total. They are initially sent segments 1, 2, 3, 4, 5 and later re-sent segments 2.

B sends 5 ACKs. They are 4 ACKS with sequence number 2. There is one ACK with sequence numbers 6. Note that TCP always send an ACK with expected sequence number.

1. TCP. This is because TCP uses fast retransmit without waiting until time out.

### Problem 40

1. TCP slowstart is operating in the intervals [1,6] and [23,26]
2. TCP congestion avoidance is operating in the intervals [6,16] and [17,22]
3. After the 16th transmission round, packet loss is recognized by a triple duplicate ACK. If there was a timeout, the congestion window size would have dropped to 1.
4. After the 22nd transmission round, segment loss is detected due to timeout, and hence the congestion window size is set to 1.
5. The threshold is initially 32, since it is at this window size that slow start stops and congestion avoidance begins.
6. The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 16, the congestion windows size is 42. Hence the threshold is 21 during the 18th transmission round.
7. The threshold is set to half the value of the congestion window when packet loss is detected. When loss is detected during transmission round 22, the congestion windows size is 29. Hence the threshold is 14 (taking lower floor of 14.5) during the 24th transmission round.
8. During the 1st transmission round, packet 1 is sent; packet 2-3 are sent in the 2nd transmission round; packets 4-7 are sent in the 3rd transmission round; packets 8-15 are sent in the 4th transmission round; packets 16-31 are sent in the 5th transmission round; packets 32-63 are sent in the 6th transmission round; packets 64 – 96 are sent in the 7th transmission round. Thus packet 70 is sent in the 7th transmission round.
9. The threshold will be set to half the current value of the congestion window (8) when the loss occurred and congestion window will be set to the new threshold value + 3 MSS . Thus the new values of the threshold and window will be 4 and 7 respectively.
10. threshold is 21, and congestion window size is 4.
11. round 17, 1 packet; round 18, 2 packets; round 19, 4 packets; round 20, 8 packets; round 21, 16 packets; round 22, 21 packets. So, the total number is 52.

Note: Additional explanation for (k): In round 21, after receiving first 5 ACKs, cwnd = 16+5=21, and TCP enters congestion avoidance state. For each of the remaining 11 ACKs, cwnd = cwnd+1/cwnd (note: the unit here is “segment”). Hence, at the end of round 21, cwnd = 21.518, which only allows sending 21 segments in round 22.

### Problem 45

1. The loss rate,, is the ratio of the number of packets lost over the number of packets sent. In a cycle, 1 packet is lost. The number of packets sent in a cycle is











Thus the loss rate is



Note: The problem does not specify the unit of W. The above implicitly assumes that W is in “packets” or multiples of MSS. If W is measured in bytes, then you need to replace “W” in the above derivation by “W/MSS”, i.e., converting it to the number of packets.

b) For  large, . Thus  or . From the text, we therefore have

average throughput 

