COMPUTER NETWORKS

Question Bank

Unit – 3
Transport Layer

1. Consider the scenario below:
   Suppose a process in Host C has a UDP socket with port number 6789. Suppose both Host A and Host B each send a UDP segment to Host C with destination port number 6789.

   Questions:
   a. Will both of these segments be directed to the same socket at Host C?
   b. If so, how will the process a Host C know that these two segments originated from two different hosts?

---

2. Consider the scenario below:
   Suppose that a Web server runs in Host C on port 80. Suppose this Web server uses persistent connections, and is currently receiving requests from two different Hosts, A and B.

   Questions:
   a. Are all of the requests being sent through the same socket at Host C?
   b. If they are being passed through different sockets, do both sockets have port 80? Discuss and explain.

---

3. Consider the scenario below:

   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.

   Questions:
   a. How much data is in the first segment?
   b. 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?

4. Consider the Telnet example , A few seconds after the user types the letter ‘C,’ the user types the letter ‘R.’ After typing the letter ‘R,’

   Questions:

   a. how many segments are sent,
   b. what is put in the sequence number and acknowledgment fields of the segments?

5. Consider the scenario below:

   Suppose two TCP connections are present over some bottleneck link of rate R bps. Both connections have a huge file to send (in the same direction over the bottleneck link). The transmissions of the files start at the same time.

   Question:

   a. What transmission rate would TCP like to give to each of the connections?

6. Consider the scenario below

   Suppose Client A initiates a Telnet session with Server S. At about the same time, Client B also initiates a Telnet session with Server S.

   Questions:

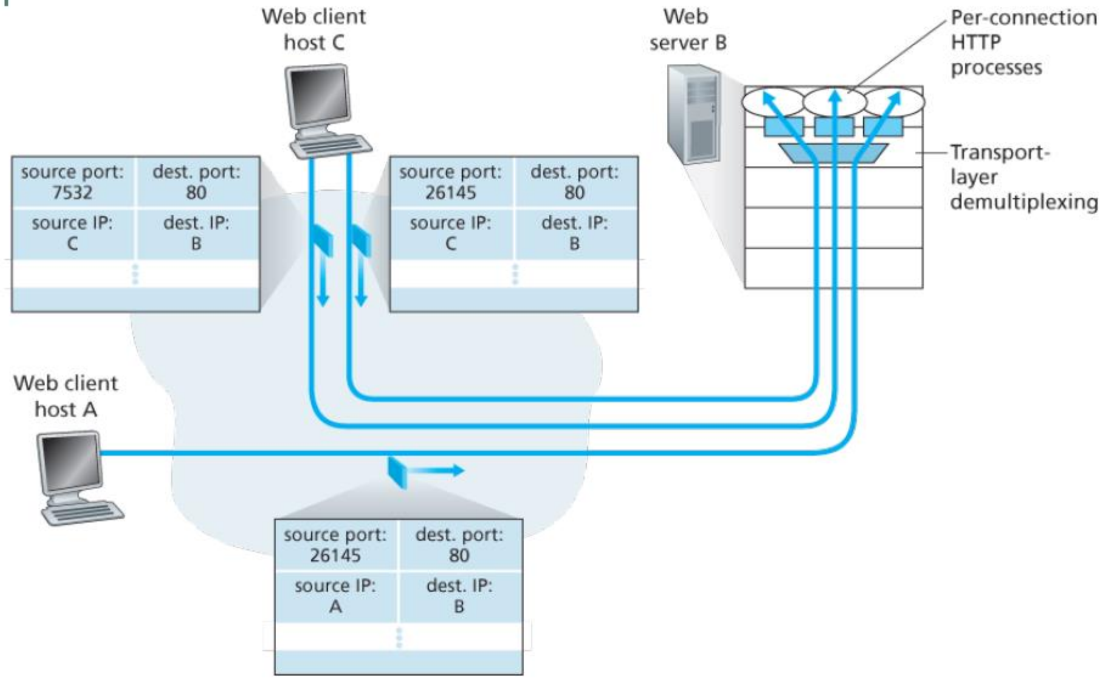
   Provide possible source and destination port numbers for

   a. The segments sent from A to S.
   b. The segments sent from B to S.
   c. The segments sent from S to A.
   d. The segments sent from S to B.
   e. If A and B are different hosts, is it possible that the source port number in the segments from A to S is the same as that from B to S?
   f. How about if they are the same host?

7. Consider the figure below,

   Question:

   a. what are the source and destination port values in the segments flowing from the server back to the clients' processes?
   b. What are the IP addresses in the network-layer datagrams carrying the transport-layer segments?



8. UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100.

   Questions:

   a. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work.
   b. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum?
   c. With the 1s complement scheme, how does the receiver detect errors?
   d. Is it possible that a 1-bit error will go undetected?
   e. How about a 2-bit error?

9. Consider UDP and TCP Checksum, Solve the following

   Questions:

   a. Suppose you have the following 2 bytes: 01011100 and 01100101. What is the 1s complement of the sum of these 2 bytes?
   b. Suppose you have the following 2 bytes: 11011010 and 01100101. What is the 1s complement of the sum of these 2 bytes?
   c. For the bytes in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1s complement doesn't change.

10. Consider the scenario below

    Suppose that the UDP receiver computes the Internet checksum for the received UDP segment and finds that it matches the value carried in the checksum field.

    Question:

    Can the receiver be absolutely certain that no bit errors have occurred? Explain.

11. Consider the scenario below

    In protocol rdt3.0, the ACK packets flowing from the receiver to the sender do not have sequence numbers (although they do have an ACK field that contains the sequence number of the packet they are acknowledging).

    Question:

    Why is it that our ACK packets do not require sequence numbers?

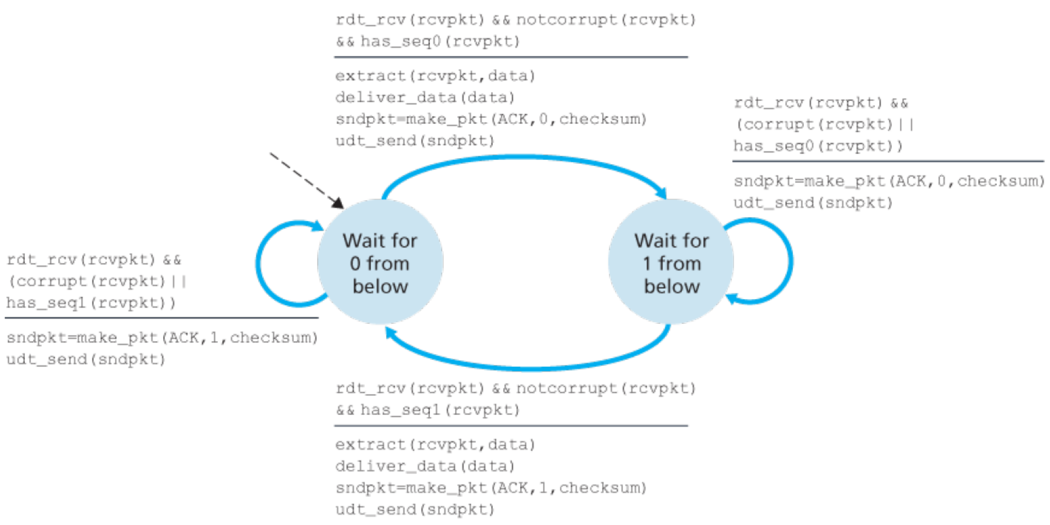
12. Give a trace of the operation of protocol rdt3.0 when data packets and acknowledgment packets are garbled. Your trace should be similar to that used in figure 3.16

13. Consider a channel that can lose packets but has a maximum delay that is known.

Questions:

a. Modify protocol rdt2.1 to include sender timeout and retransmit.
b. Informally argue why your protocol can communicate correctly over this channel.

14. Consider the rdt2.2 receiver as shown in below figure, and the creation of a new packet in the self-transition (i.e., the transition from the state back to itself) in the Wait-for-0-from-below and the Wait-for-1-from-below states: sndpkt=make_pkt(ACK, 1, checksum) and sndpkt=make_pkt(ACK, 0, checksum.


rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq0(rcvpkt)
extract(rcvpkt,data) deliver_data(data) sndpkt=make_pkt(ACK,0,checksum) udt_send(sndpkt)
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)|| has_seq0(rcvpkt))
sndpkt=make_pkt(ACK,0,checksum) udt_send(sndpkt)
Wait for 0 from below
Wait for 1 from below
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)|| has_seq1(rcvpkt))
sndpkt=make_pkt(ACK,1,checksum) udt_send(sndpkt)
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq1(rcvpkt)
extract(rcvpkt,data) deliver_data(data) sndpkt=make_pkt(ACK,1,checksum) udt_send(sndpkt)

Questions:

a. Would the protocol work correctly if this action were removed from the self-transition in the Wait-for-1-from-below state? Justify your answer.
b. What if this event were removed from the self-transition in the Wait-for-0-from-below state?

15. Consider the scenario below

The sender side of rdt3.0 simply ignores (that is, takes no action on) all received packets that are either in error or have the wrong value in the acknum field of an acknowledgment packet.

Questions:

Suppose that in such circumstances, rdt3.0 were simply to retransmit the current data packet.

a. Would the protocol still work?
b. Consider how many times the nth packet is sent, in the limit as n approaches infinity.

---

16. Consider the rdt 3.0 protocol.

Questions:

a. Draw a diagram showing that if the network connection between the sender and receiver can reorder messages (that is, that two messages propagating in the medium between the sender and receiver can be reordered), then the alternating-bit protocol will not work correctly (make sure you clearly identify the sense in which it will not work correctly).
b. Your diagram should have the sender on the left and the receiver on the right, with the time axis running down the page, showing data (D) and acknowledgment (A) message exchange. Make sure you indicate the sequence number associated with any data or acknowledgment segment.

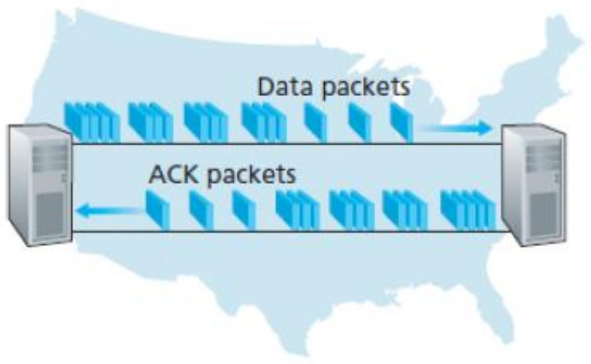
---

17. Consider a reliable data transfer protocol that uses only negative acknowledgments. Suppose the sender sends data only infrequently.

a. Would a NAK-only protocol be preferable to a protocol that uses ACKs? Why?
b. Now suppose the sender has a lot of data to send and the end-to-end connection experiences few losses. In this second case, would a NAK-only protocol be preferable to a protocol that uses ACKs? Why?

---

18. Consider the cross-country in the figure below



Questions:

a. How big would the window size have to be for the channel utilization to be greater than 98 percent?
b. Suppose that the size of a packet is 1,500 bytes, including both header fields and data.

19. Consider the scenario below

Suppose an application uses rdt 3.0 as its transport layer protocol. As the stop-and-wait protocol has very low channel utilization (shown in the cross-country example), the designers of this application let the receiver keep sending back a number (more than two) of alternating ACK 0 and ACK 1 even if the corresponding data have not arrived at the receiver.

Questions:

a. Would this application design increase the channel utilization? Why?
b. Are there any potential problems with this approach? Explain.

20. Consider two network entities, A and B, which are connected by a perfect bi-directional channel (i.e., any message sent will be received correctly; the channel will not corrupt, lose, or re-order packets).

A and B are to deliver data messages to each other in an alternating manner:

First, A must deliver a message to B, then B must deliver a message to A, then A must deliver a message to B and so on. If an entity is in a state where it should not attempt to deliver a message to the other side, and there is an event like rdt_send(data) call from above that attempts to pass data down for transmission to the other side, this call from above can simply be ignored with a call to rdt_unable_to_send(data), which informs the higher layer that it is currently not able to send data.

[Note: This simplifying assumption is made so you don't have to worry about buffering data.]

Questions:

a. Draw a FSM specification for this protocol (one FSM for A, and one FSM for B!). Note that you do not have to worry about a reliability mechanism here; the main point of this question is to create a FSM specification that reflects the synchronized behavior of the two entities. You should use the following events and actions that have the same meaning as protocol rdt1.0 in Figure 3.9 : rdt_send(data), packet = make_pkt(data), udt_send(packet), rdt_rcv(packet), extract (packet, data), deliver_data(data).

b. Make sure your protocol reflects the strict alternation of sending between A and B. Also, make sure to indicate the initial states for A and B in your FSM descriptions.

---

21. Consider the scenario below

In the generic SR protocol, the sender transmits a message as soon as it is available (if it is in the window) without waiting for an acknowledgment.

a. Suppose now that we want an SR protocol that sends messages two at a time. That is, the sender will send a pair of messages and will send the next pair of messages only when it knows that both messages in the first pair have been received correctly.

b. Suppose that the channel may lose messages but will not corrupt or reorder messages.

Question:

a. Design an error-control protocol for the unidirectional reliable transfer of messages. Give an FSM description of the sender and receiver.

b. Describe the format of the packets sent between sender and receiver, and vice versa.

c. If you use any procedure calls other than those in Section 3.4 (for example, udt_send(), start_timer(), rdt_rcv(), and so on), clearly state their actions. Give an example (a timeline trace of sender and receiver) showing how your protocol recovers from a lost packet.

22. Consider a scenario in which Host A wants to simultaneously send packets to Hosts B and C. A is connected to B and C via a broadcast channel—a packet sent by A is carried by the channel to both B and C.

Suppose that the broadcast channel connecting A, B, and C can independently lose and corrupt packets (and so, for example, a packet sent from A might be correctly received by B, but not by C).

Questions:

a. Design a stop-and-wait-like error-control protocol for reliably transferring packets from A to B and C, such that A will not get new data from the upper layer until it knows that both B and C have correctly received the current packet.

b. Give FSM descriptions of A and C. (Hint: The FSM for B should be essentially the same as for C.) Also, give a description of the packet format(s) used.

23. Consider the scenario below

Suppose we have two network entities, A and B. B has a supply of data messages that will be sent to A according to the following conventions.

When A gets a request from the layer above to get the next data (D) message from B, A must send a request (R) message to B on the A-to-B channel. Only when B receives an R message can it send a data (D) message back to A on the B-to-A channel. A should deliver exactly one copy of each D message to the layer above. R messages can be lost (but not corrupted) in the A-to-B channel; D messages, once sent, are always delivered correctly. The delay along both channels is unknown and variable.

Questions:

a. Design (give an FSM description of) a protocol that incorporates the appropriate mechanisms to compensate for the loss-prone A-to-B channel and implements message passing to the layer above at entity A, as discussed above. Use only those mechanisms that are absolutely necessary

---

24. Consider the GBN protocol with a sender window size of 4 and a sequence number range of 1,024. Suppose that at time t, the next in-order packet that the receiver is expecting has a sequence number of k.

Assume that the medium does not reorder messages.

Questions:

a. What are the possible sets of sequence numbers inside the sender's window at time t? Justify your answer.

b. What are all possible values of the ACK field in all possible messages currently propagating back to the sender at time t? Justify your answer.

---

25. Consider the scenario below

We have said that an application may choose UDP for a transport protocol because UDP offers finer application control (than TCP) of what data is sent in a segment and when.

Questions:

a. Why does an application have more control of what data is sent in a segment?
b. Why does an application have more control on when the segment is sent?

---

26. Consider transferring an enormous file of L bytes from Host A to Host B. Assume an MSS of 536 bytes.

a. What is the maximum value of L such that TCP sequence numbers are not exhausted? Recall that the TCP sequence number field has 4 bytes.

b. For the L you obtain in (a), find how long it takes to transmit the file. Assume that a total of 66 bytes of transport, network, and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow control and congestion control so A can pump out the segments back to back and continuously.

27. Consider the scenario below

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.

Questions:

a. In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?

b. 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?

c. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?

d. 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.

28. Consider the scenario below

Host A and B are directly connected with a 100 Mbps link. There is one TCP connection between the two hosts, and Host A is sending to Host B an enormous file over this connection.

Host A can send its application data into its TCP socket at a rate as high as 120 Mbps but Host B can read out of its TCP receive buffer at a maximum rate of 50 Mbps.

Question:

a. Describe the effect of TCP flow control.

29. Consider the scenario below

SYN cookies were discussed in TCP Connection Management

Questions:

a. Why is it necessary for the server to use a special initial sequence number in the SYNACK?

b. Suppose an attacker knows that a target host uses SYN cookies. Can the attacker create half-open or fully open connections by simply sending an ACK packet to the target? Why or why not?

c. Suppose an attacker collects a large amount of initial sequence numbers sent by the server. Can the attacker cause the server to create many fully open connections by sending ACKs with those initial sequence numbers? Why?

30. Consider the network shown below.

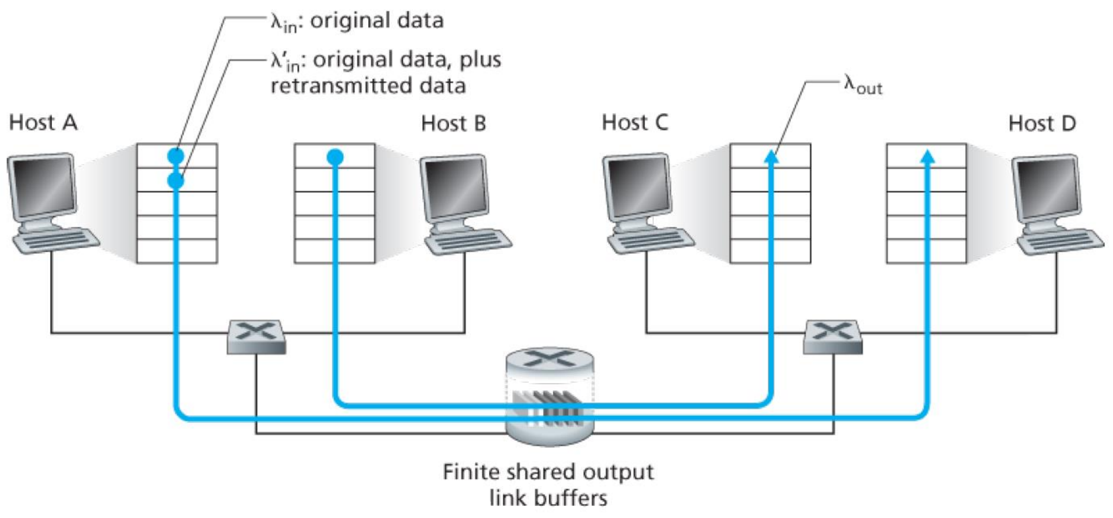

λ_{in}: original data
λ'_{in}: original data, plus retransmitted data
λ_{out}
Host A
Host B
Host C
Host D
Finite shared output link buffers

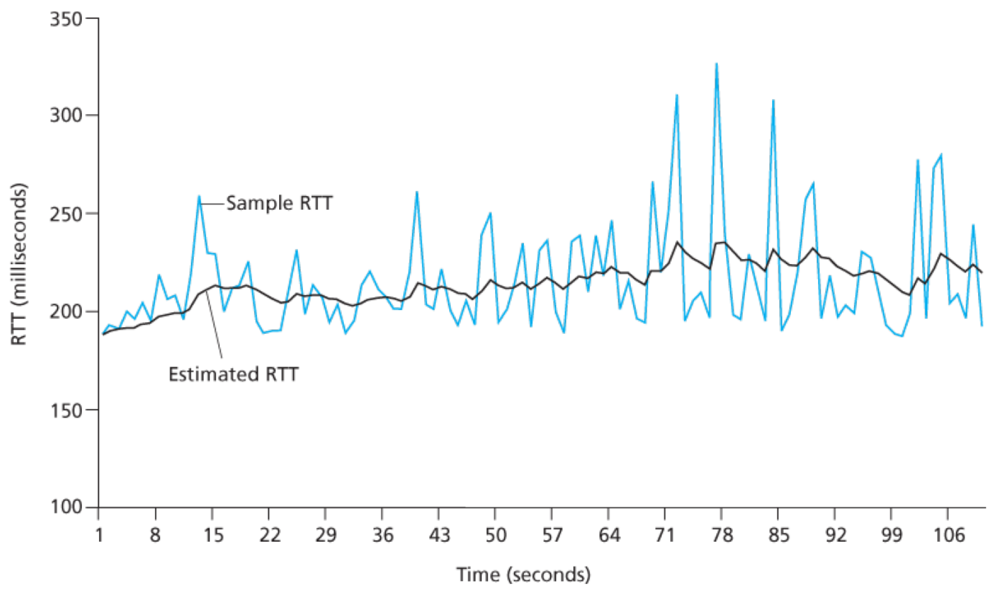
Figure 3.45 Scenario 2: Two hosts (with retransmissions) and a router with finite buffers

Suppose both sending hosts A and B have some fixed timeout values.

Questions:

a. Argue that increasing the size of the finite buffer of the router might possibly decrease the throughput (λout).
b. Now suppose both hosts dynamically adjust their timeout values (like what TCP does) based on the buffering delay at the router. Would increasing the buffer size help to increase the throughput? Why?

31. Suppose that the five measured SampleRTT values are 106 ms, 120 ms, 140 ms, 90 ms, and 115 ms.



Questions:

a. Compute the EstimatedRTT after each of these SampleRTT values is obtained, using a value of α=0.125 and assuming that the value of EstimatedRTT was 100 ms just before the first of these five samples were obtained.
b. Compute also the DevRTT after each sample is obtained, assuming a value of β=0.25 and assuming the value of DevRTT was 5 ms just before the first of these five samples was obtained. Last, compute the TCP TimeoutInterval after each of these samples is obtained.

32. Consider the TCP procedure for estimating RTT. Suppose that α=0.1.
Let SampleRTT1 be the most recent sample RTT, let SampleRTT2 be the next most recent sample RTT, and so on.

For a given TCP connection, suppose four acknowledgments have been returned with corresponding sample RTTs: SampleRTT4, SampleRTT3, SampleRTT2, and SampleRTT1. Questions:

a. Express EstimatedRTT in terms of the four sample RTTs.
b. Generalize your formula for n sample RTTs.
c. For the formula in part (b) let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.

33. Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively.

Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been correctly received by Host B.

Questions:

a. How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.

b. If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in shortest time interval?

34. In our description of TCP as shown in figure below,

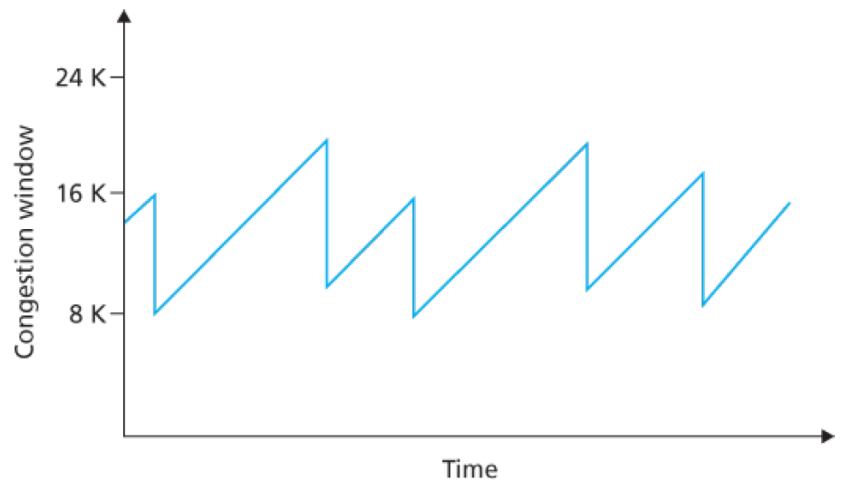


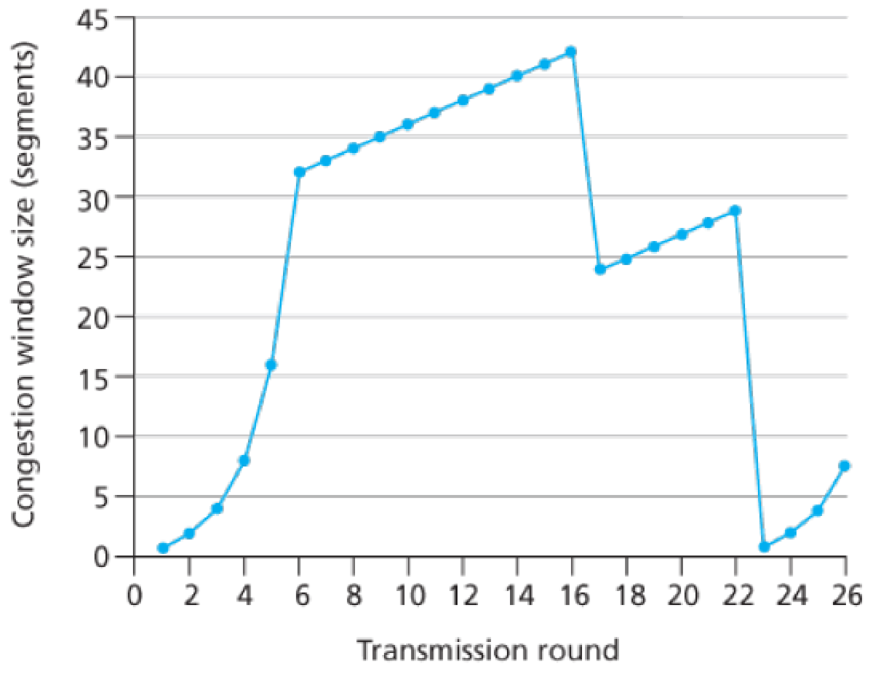
Figure 3.53 Additive-increase, multiplicative-decrease congestion control

The value of the threshold, ssthresh, is set as ssthresh=cwnd/2 in several places and ssthresh value is referred to as being set to half the window size when a loss event occurred.

Questions:

a. Must the rate at which the sender is sending when the loss event occurred be approximately equal to cwnd segments per RTT? Explain your answer.
b. If your answer is no, can you suggest a different manner in which ssthresh should be set?

35. Consider the figure shown below.



Assuming TCP Reno is the protocol experiencing the behavior shown above, answer the following questions. In all cases, you should provide a short discussion justifying your answer.

Questions:

a. Identify the intervals of time when TCP slow start is operating.
b. Identify the intervals of time when TCP congestion avoidance is operating.
c. After the 16th transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
d. After the 22nd transmission round, is segment loss detected by a triple duplicate ACK or by a timeout?
e. What is the initial value of ssthresh at the first transmission round? f. What is the value of ssthresh at the 18th transmission round?
f. What is the value of ssthresh at the 24th transmission round?
g. During what transmission round is the 70th segment sent?
h. Assuming a packet loss is detected after the 26th round by the receipt of a triple duplicate ACK, what will be the values of the congestion window size and of ssthresh?
i. Suppose TCP Tahoe is used (instead of TCP Reno), and assume that triple duplicate ACKs are received at the 16th round. What are the ssthresh and the congestion window size at the 19th round?

j. Again suppose TCP Tahoe is used, and there is a timeout event at 22nd round. How many packets have been sent out from 17th round till 22nd round, inclusive?

36. Consider the scenario below

Host A is sending an enormous file to Host B over a TCP connection. Over this connection there is never any packet loss and the timers never expire.

Questions:

a. Denote the transmission rate of the link connecting Host A to the Internet by R bps. Suppose that the process in Host A is capable of sending data into its TCP socket at a rate S bps, where S=10·R.
b. Further suppose that the TCP receive buffer is large enough to hold the entire file, and the send buffer can hold only one percent of the file. What would prevent the process in Host A from continuously passing data to its TCP socket at rate S bps? TCP flow control? TCP congestion control? Or something else? Elaborate.

37. Consider sending a large file from a host to another over a TCP connection that has no loss.

Suppose TCP uses AIMD for its congestion control without slow start. Assuming cwnd increases by 1 MSS every time a batch of ACKs is received and assuming approximately constant round-trip times,

Questions:

a. how long does it take for cwnd increase from 6 MSS to 12 MSS (assuming no loss events)?

b.What is the average throughout (in terms of MSS and RTT) for this connection up through time=6 RTT?

Recall the macroscopic description of TCP throughput. In the period of time from when the connection's rate varies from W/(2 · RTT) to W/RTT, only one packet is lost (at the very end of the period).

a. Show that the loss rate (fraction of packets lost) is equal to L=loss rate=138W2+34W

b. Use the result above to show that if a connection has loss rate L, then its average rate is approximately given by ≈1.22·MSSRTTL

38. Consider that only a single TCP (Reno) connection uses one 10Mbps link which does not buffer any data. Suppose that this link is the only congested link between the sending and receiving hosts. Assume that the TCP sender has a huge file to send to the receiver, and the receiver's receive buffer is much larger than the congestion window. We also make the following assumptions: each TCP segment size is 1,500 bytes; the two-way propagation delay of this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is, ignore slow start.

Questions:

a. What is the maximum window size (in segments) that this TCP connection can achieve? b. What is the average window size (in segments) and average throughput (in bps) of this TCP connection?
b. How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?

39. Consider the scenario described in the previous problem. Suppose that the 10Mbps link can buffer a finite number of segments. Argue that in order for the link to always be busy sending data, we would like to choose a buffer size that is at least the product of the link speed C and the two-way propagation delay between the sender and the receiver.

40. Consider the scenario:

Let T (measured by RTT) denote the time interval that a TCP connection takes to increase its congestion window size from W/2 to W, where W is the maximum congestion window size.

Question:
Argue that T is a function of TCP's average throughput.

41. Consider a simplified TCP's AIMD algorithm where the congestion window size is measured in number of segments, not in bytes.

In additive increase, the congestion window size increases by one segment in each RTT.

In multiplicative decrease, the congestion window size decreases by half (if the result is not an integer, round down to the nearest integer).

Suppose that two TCP connections, C1 and C2, share a single congested link of speed 30 segments per second. Assume that both C1 and C2 are in the congestion avoidance phase. Connection C1's RTT is 50 msec and connection C2's RTT is 100 msec. Assume that when the data rate in the link exceeds the link's speed, all TCP connections experience data segment loss.

Questions:

a. If both C1 and C2 at time t0 have a congestion window of 10 segments, what are their congestion window sizes after 1000 msec?

b. In the long run, will these two connections get the same share of the bandwidth of the congested link? Explain.

---

42. Consider the network described in the previous problem. Now suppose that the two TCP connections, C1 and C2, have the same RTT of 100 msec.

Suppose that at time t0, C1's congestion window size is 15 segments but C2's congestion window size is 10 segments.

Questions:

a. What are their congestion window sizes after 2200 msec?

b. In the long run, will these two connections get about the same share of the bandwidth of the congested link?

c. We say that two connections are synchronized, if both connections reach their maximum window sizes at the same time and reach their minimum window sizes at the same time.

In the long run, will these two connections get synchronized eventually? If so, what are their maximum window sizes?

d. Will this synchronization help to improve the utilization of the shared link? Why? Sketch some idea to break this synchronization.

---

43. Consider a modification to TCP's congestion control algorithm. Instead of additive increase, we can use multiplicative increase. A TCP sender increases its window size by a small positive constant a(0<a<1) whenever it receives a valid ACK.

Questions:

Find the functional relationship between loss rate L and maximum congestion window W.
Argue that for this modified TCP, regardless of TCP's average throughput, a TCP connection always spends the same amount of time to increase its congestion window size from W/2 to W.

44. .

Sketch some idea to break this synchronization.

45. Consider a modification to TCP's congestion control algorithm. Instead of additive increase, we can use multiplicative increase. A TCP sender increases its window size by a small positive constant a(0<a<1) whenever it receives a valid ACK. Find the functional relationship between loss rate L and maximum congestion window W. Argue that for this modified TCP, regardless of TCP's average throughput, a TCP connection always spends the same amount of time to increase its congestion window size from W/2 to W.