<u>Assignment 8</u>

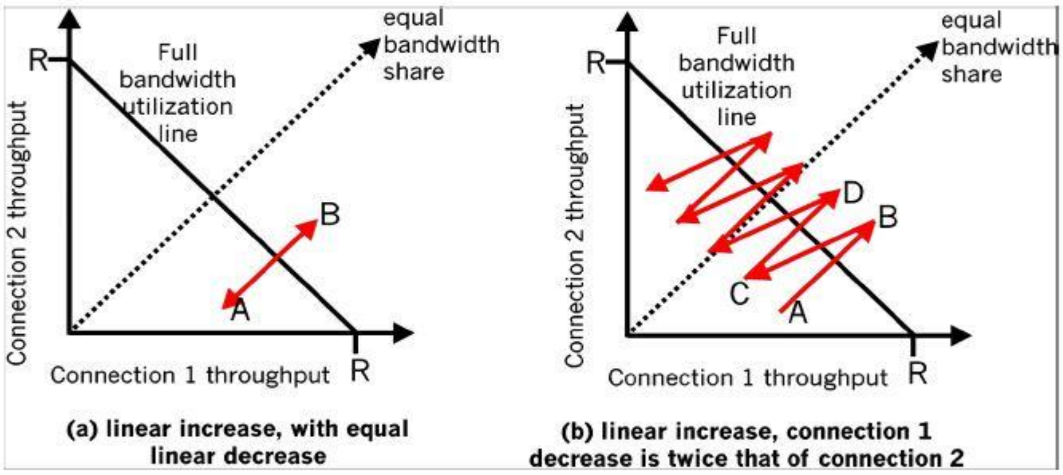
Name: Rajat Rajesh Shetty
CWID: 10477484

36. In **Section 3.5.4** , we saw that TCP waits until it has received three duplicate ACKs before performing a fast retransmit. Why do you think the TCP designers chose not to perform a fast retransmit after the first duplicate ACK for a segment is received?

<u>Answer:</u> Suppose packets n, n+1, and n+2 are sent, and that packet n is received and ACKed. If packets n+1 and n+2 are reordered along the end-to-end-path(i.e., are received in the order n+2, n+1) then the receipt of packet n+2 will generate a duplicate ack for n and would trigger are transmission under a policy of waiting only for second duplicate ACK for retransmission. By waiting for a triple duplicate ACK, it must be the case that two packets after packet are correctly received, while n+1 was not received. The designers of the triple duplicate ACK scheme probably felt that waiting for two packets (rather than 1) was the right trade off between triggering a quick retransmission when needed, but not retransmitting prematurely in the face of packet reordering.

41. Refer to Figure 3.56, which illustrates the convergence of TCP's AIMD algorithm. Suppose that instead of a multiplicative decrease, TCP decreased the window size by a constant amount. Would the resulting AIAD algorithm converge to an equal share algorithm? Justify your answer using a diagram similar to Figure 3.56.

Answer: Refer to Figure5. In Figure 5(a), the ratio of the linear decrease on loss between connection1 and connection2 is the same-as ratio of the linear increases: unity. In this case, the through puts never move off of the AB line segment. In Figure 5(b), the ratio of the linear decrease on loss between connection 1 and connection2is2:1.That is, whenever there is a loss, connection1 decreases its Window by twice the amount of connection2. We see that eventually, after enough losses, and subsequent increases, that connection1's throughput will go too, and the full link bandwidth will be allocated to connection2.



R
Full bandwidth utilization line
equal bandwidth share
B
A
Connection 2 throughput
Connection 1 throughput
R
(a) linear increase, with equal linear decrease
R
Full bandwidth utilization line
equal bandwidth share
D
B
C
A
Connection 2 throughput
Connection 1 throughput
R
(b) linear increase, connection 1 decrease is twice that of connection 2

Figure 5: Lack of TCP convergence with linear increase, linear decrease

43. 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. 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. 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.

Answer: In this problem, there is no danger in overflowing the receiver since the receiver's receive buffer can hold the entire file. Also, because there is no loss and acknowledgements are returned before timers expire, TCP congestion control does not throttle the sender. However, the process in host A will not continuously pass data to the socket because the send buffer will quickly fill up. Once the send buffer becomes full, the process will pass data at an average rate or R << S.

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

a. 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, 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?

Answer: a)

Given data:

Assuming cwnd increases by 1 MSS every time a batch of ACKs is received and assuming approximately constant round-trip times.

cwnd increases by 1 MSS if every batch of ACKs received.

The below steps are take for cwnd to increase from 6 MSS to 12 MSS:

- 1 RTTs to 7 MSS.
- 2 RTTs to 8 MSS.
- 3 RTTs to 9 MSS.
- 4 RTTs to 10 MSS.
- 5 RTTs to 11MSS.
- 6 RTTs to 12 MSS.

b)

Given data:

Connection up through time = 6 RTT

Average throughout (in terms of MSS and RTT) = (6+7+8+9+10+11)/6

=8.5 MSS/RTT

46. 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.

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?

c. How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?

Answer:

a) Let W denote the max window size measured in segments. Then, W*MSS/RTT=10Mbps, as packets will be dropped if the maximum sending rate exceeds link capacity. Thus, we have W*1500*8/0.15=10*10^6, then W is about125 segments.

b) As congestion window size varies from W/2toW, then the average window size is 0.75W=94 (ceiling of93.75) segments. Average throughput is 94*1500*8/0.15=7.52Mbps.

c) When there is a packet loss, W becomes W/2, i.e., 125/2=62. (125-62)*0.15=9.45 seconds, as the number of RTTs (that this TCP connections needs in order to increase its window size from 62to125) is 63. Recall the window size increases by one in each RTT.

54. In our discussion of TCP congestion control in Section 3.7, we implicitly assumed that the TCP sender always had data to send. Consider now the case that the TCP sender sends a large amount of data and then goes idle (since it has no more data to send) at t 1. TCP remains idle for a relatively long period of time and then wants to send more data at t 2. What are the advantages and disadvantages of having TCP use the cwnd and ssthresh values from t 1 when starting to send data at t 2? What alternative would you recommend? Why?

Answer: An advantage of using the earlier values of cwnd and ssthresh at t2 is that TCP would not have to go through slow start and congestion avoidance to ramp up to the throughput value obtained at t1. A disadvantage of using these values is that they may be no longer accurate. In particular, if the path has become more congested between t1 and t2, the sender will send a large window's worth of segments into an already (more) congested path.

53. In our discussion of TCP futures in Section 3.7, we noted that to achieve a throughput of 10 Gbps, TCP could only tolerate a segment loss probability of 2 · 10 -10 (or equivalently, one loss event for every 5,000,000,000 segments). Show the derivation for the values of 2 · 10 -10 (1 out of 5,000,000) for the RTT and MSS values given in Section 3.7. If TCP needed to support a 100 Gbps connection, what would the tolerable loss be?

For 10Gbps:

Throughput (or) Bandwidth(B) = 10 Gbps $= 10\times10^{9}$ bps $= 10^{10}$ bps

$MaximumSegmentSize(MSS)$ = 1500 bytes = 1500×8 bits

$Round\ Trip\ Time(RTT)$ = 100ms = 100×10^{-3} sec = 0.1 sec

Let L denotes the loss rate and B denotes the average bandwidth (or) throughput.

Average bandwidth (or) throughput $(B) = \dfrac{1.22\times MSS}{RTT\sqrt{L}}$

From the above formula, loss rate L is as shown below:

$$B=\frac{1.22\times MSS}{RTT\sqrt{L}}$$
$$\sqrt{L}=\frac{1.22\times MSS}{RTT\times B}$$
$$L=\left(\frac{1.22\times MSS}{RTT\times B}\right)^2$$

Substitute the values of MSS, B and RTT in the above formula.

$$\begin{aligned}L&=\left(\frac{1.22\times MSS}{RTT\times B}\right)^2\\&=\left(\frac{1.22\times1500\times8}{0.1\times10^{10}}\right)^2\\&=\left(\frac{14640}{10^{9}}\right)^2\\&=(0.00001464)^2\\&=0.0000000002143296\\&=2.14\times10^{-10}\end{aligned}$$

Hence, the tolerable loss L will be 2.14×10^{-10} **or** 2×10^{-10} **(approx).**

For 100Gbps:

Throughput(or)Bandwidth(B)= 100 Gbps $=100\times10^{9}$ bps $=10^{11}$ bps

$Maximum\ Segment\,Size(MSS)$ = 1500 bytes=1500×8 bits

$Round\ Trip\ Time(RTT)$=100 ms=100×10^{-3} sec=0.1sec

Let L be loss rate and B be average throughput (or) bandwidth.

Average throughput (or) bandwidth $(B)=\dfrac{1.22\times MSS}{RTT\sqrt{L}}$

From the above formula, loss rate L is as shown below.

$$B=\frac{1.22\times MSS}{RTT\sqrt{L}}$$
$$\sqrt{L}=\frac{1.22\times MSS}{RTT\times B}$$
$$L=\left(\frac{1.22\times MSS}{RTT\times B}\right)^{2}$$

Substitute the values of MSS, B and RTT in the above formula.

$$\begin{aligned}L&=\left(\frac{1.22\times MSS}{RTT\times B}\right)^{2}\\&=\left(\frac{1.22\times1500\times8}{0.1\times10^{11}}\right)^{2}\\&=\left(\frac{14640}{10^{10}}\right)^{2}\\&=(0.000001464)^{2}\\&=0.000000000002143296\\&=2.14\times10^{-12}\end{aligned}$$

Hence, the tolerable loss L will be 2.14×10^{-12} **or** 2×10^{-12} **(approximately).**