CS4450 Problem Set #6

1

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 approximately constant round-trip times, how long does it take for cwnd to increase from 6 MSS to 12 MSS (assuming no loss events)?

(b) Suppose MSS = 1000bytes. What is the average throughout (in terms of MSS and RTT) for this connection up through time = 6 RTT after connection setup, where thoughput is defined as the total amount of data sent over the specified time?

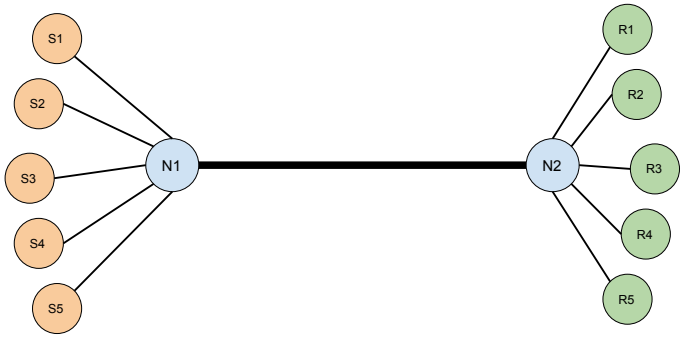
2

Consider a resource allocation situation as follows:

- We have a shared resource of capacity C.
- This shared resource is requested by requestors $R_1, R_2, R_3, ..., R_n$.
- Each R_i has a demand d_i for the resource.
- For instance, consider this:
  - Resource: Your time in a day
  - Requestors: Tasks which you have to perform in the day, which take time.
- Now, we have to allocate the shared resource C among these requestors. Clearly, when $\Sigma d_i \leq C$, there is no problem - Every requestor's demand can be satisfied.
- However, things get interesting when $\Sigma d_i > C$. In this case, not all requestors can have their demands satisfied.
- In this resource allocation problem, **max-min fairness** is said to be achieved by a feasible allocation if an attempt to increase the share of resource allocated to a requestor would mean the decrease in allocation of resources to another requestor with an equal or smaller allocation.
- Here is a way one can make max-min fair allocations:
  - Shared Resource: C.
  - Demands: $d_1, d_2, ..., d_n$, such that $d_1 \leq d_2 \leq d_3 \leq ... \leq d_n$.
  - Requestors R_i has demand d_i.
  - Initally, none of the requestors have been allocated any share of the resource.
  - We increase every requestor's share equally, until either:

    - R_1's requested share (d_1) is fulfilled.
    - We run out of C. That is: $n.d_1 > C$. In this case, we are done. Each requestor gets C/n.
  - Now, R_1's request is fulfilled, and we still have some part of C left.
  - We now keep R_1 aside, and increase every other requestors share until R_2's share is fulfilled (or we run out of C).
  - In this way, we keep going ($d_3, d_4, ...$), until we run out of C.
  - The final resource allocations are **max-min fair**.

In case of multiple TCP flows sharing a single link, it is a well-known result that the bandwidth allocation per flow that TCP achieves is **max-min fair**. Consider the situation we have as shown below:



- Each link has a capacity of 1000 bps.
- In this situation, we have 5 flows. Each S_i communicates with R_i.
- Let the bandwidth demands by each S_i be d_i.

Calculate the final bandwidth allocations for each TCP flow in the following cases. Also verify that these allocations are max-min fair.

1.

Source S_i	**Demand d_i (bps)**	**Allocation (bps)**
S_1	200	
S_2	200	
S_3	200	
S_4	200	
S_5	200	

2.

Source S_i	**Demand d_i (bps)**	**Allocation (bps)**
S_1	100	
S_2	200	
S_3	300	
S_4	400	
S_5	500	

3.

Source S_i	**Demand d_i (bps)**	**Allocation (bps)**
S_1	100	
S_2	400	
S_3	200	
S_4	400	
S_5	100	

3

(a) A sender and a receiver are communicating using a TCP-based protocol in which data is sent immediately after connection. The bandwidth of the link is 100Mbps, the round-trip time is 10ms and the maximum segment size is 1500 bytes (ignore header overhead). Draw a packet exchange diagram showing how long it takes to transfer 15000 bytes.

(b) Consider that only a single TCP 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 RTT for this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is, ignore slow start.

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

4

(a) Consider two TCP flows, one over link X, and the other over link Y. X is a 100Mbps Ethernet link with a 10ms RTT, and Y is a 1Gbps satellite link with a 1000ms RTT. Assuming both links have the same loss probability p , and the flows always use the same packet size and never saturate the links, what's the link utilization ratio of X to Y?

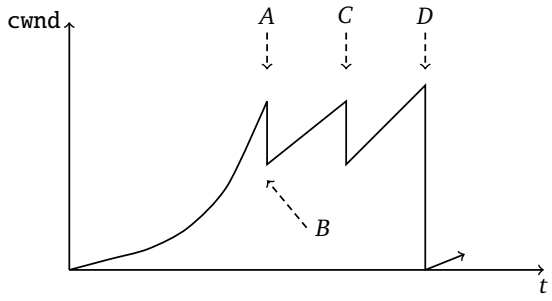
(b) Consider two TCP flows, one over wireless links X, and the other over wireless link Y. Both links have 25ms RTT. If the interference causes a 1% packet loss on X, and a 4% packet loss on Y, respectively, what's the link throughput ratio of X to Y? (Assume there's no forward error correction, and there is no link layer retransmission.)

(c) From the above answers, we know that TCP does not perform very well in the presence of high RTT or lossy links. Give one solution to improve TCP throughput on links with high RTT, and one solution to improve TCP throughput on lossy links.

(d) Suppose a TCP flow and a UDP flow are sharing the same link, and the UDP sends at a rate equal to the link capacity. Assuming the flows stay for long enough, how would they split the bandwidth eventually? If they would equally share the bandwidth, rationalize it. If a particular one would monopolize the bandwidth, explain which one and why.

5

Consider the plot below, where x-axis denotes the time and y-axis denotes the TCP window size for TCP.



(a) Describe the slow start mechanism (from time 0 to event A), and the AIMD mechanism (between event A and C) in at most two lines each.

(b) Name the event that triggers the window size reduction at A.

(c) Does the event at A necessitate a packet drop? Why or Why not?

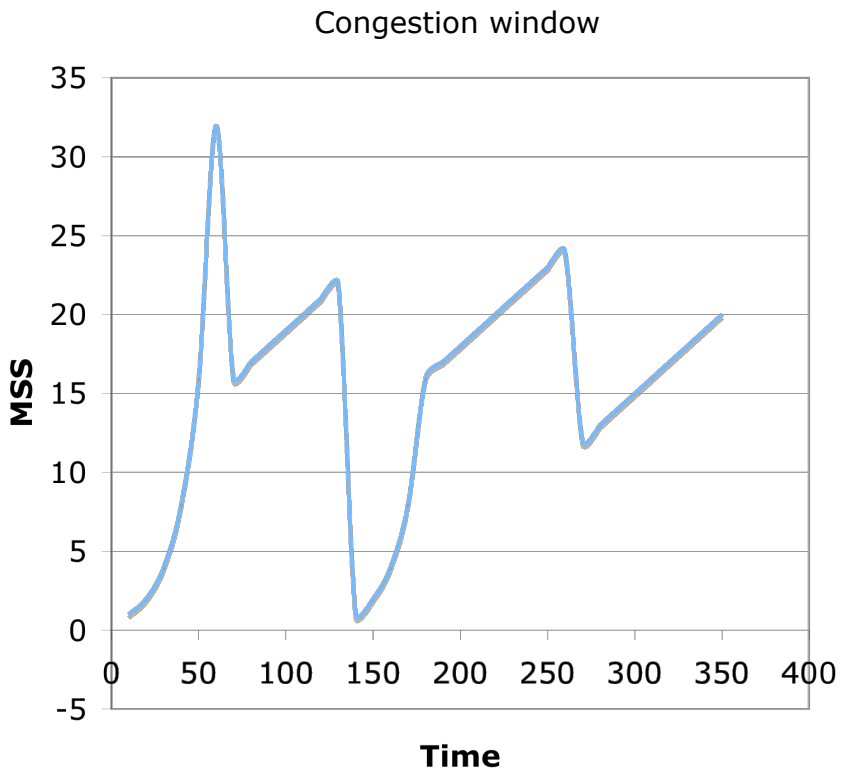
(d) Name the event that triggers the window size reduction at D.

(e) Does the event at D necessitate a packet drop? Why or Why not?

(f) Suppose the window size right before event D is 16. In the absence of congestion after event D, would the window size ever be larger than 16? Why or why not?

6

(a) Describe what is happening in this TCP trace at the following times.(One or two word answers will suffice.)



- $t = 0$–60:
- $t = 60$:
- $t = 60$–140:
- $t = 140$:
- $t = 140$–180:
- $t = 180$–270:
- $t = 270$:
- $t = 270$–350:

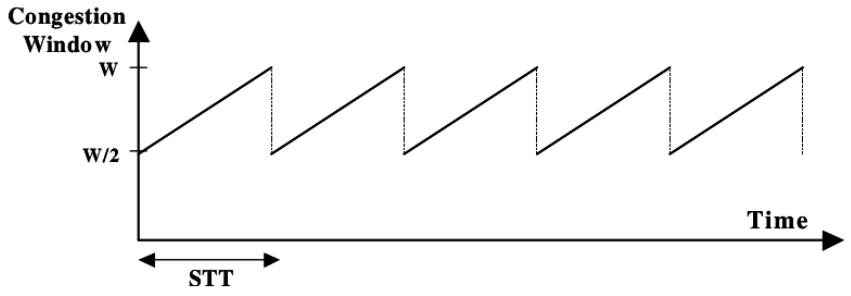
(b) Recall that one reason why congestion leads to poor performance is the large queueing delays that are associated with router queues. Would the Internet perform better if router queues were eliminated (or made much smaller)? Why or why not?

(c) In TCP, each endpoint picks a random initial sequence number (ISN). Why is this done, instead of starting each sequence number at 0?

(d) RFC 7323 defines a **TCP Window Scale Option** that allows the specification of larger window sizes, up to 2^{30} bytes instead of the normal 2^{16}. When would this option be useful?

7

The picture below shows the famous TCP saw tooth behavior. We are assuming that fast retransmit and fast recovery always work, i.e. there are no timeouts and there is exactly one packet lost at the end of each "tooth". We are assuming that the flow control window is large and that the sender always has data to send, i.e. throughput will be determined by TCP congestion control. In the picture, W represents the congestion window size at which a congestion packet loss occurs (expressed in maximum transfer units). You can assume that W is large, so feel free to approximate (W-1) or (W+1) by W. STT represents the "saw tooth time" expressed in seconds.



In this question, you will calculate the average throughput T for this connection as a function of the roundtrip time (RTT), the maximum transfer unit size (MTU), and packet loss rate p for this connection. Please use the notation suggested by the figure, i.e. W and STT, as intermediate values if you need them

(a) Express STT in terms of RTT and W.

(b) Based on the shape of saw tooth, how many units can be sent within each STT?

(c) Since exactly one packet (unit) loss event happens at the end of each STT, express p in terms of W?

(d) TCP throughput (T) is in generallly defined as number of bytes divided by the duration of time to send them. Given the results in the previous questions, please calculate the average TCP throughput as a function of RTT, MTU and p.