We'll post an outline of each recitation after the fact so you know what was covered in case you had to miss class that day. The recitation outlines are not a full replacement for recitation!

This week's recitation was largely a problem-solving session (good for exam prep). These are the problems we went over; some sections may not have gotten to every problem.

DCTCP

We have a large-scale web application using the partition/aggregate design pattern with many workers and a requirement for real-time responsiveness. Assume there is NO background traffic (no other applications in the network) and all responses from workers are small (1-2 packets each).

Question 1: Which of these problems is likely to be a concern in our network? Circle all that apply.

a. Incast
b. Queue building
c. Buffer pressure
d. None of the above

For the same application, Max has a bright idea of changing the aggregator structure. For example, instead of having one aggregator to aggregate the answer of 1000 workers, Max proposes using 10 aggregators each aggregating the answer from 100 workers, and then one aggregator to aggregate the answer from the ten aggregators.

Question 2: What is the most significant potential problem with this approach? Circle the best answer.

a. It substantially increases the number of machines required
b. Doubling levels of aggregation might add unacceptable delay
c. It creates longer queues
d. It intensifies incast

Question 3: TCP with RED/ECN has some similarities to DCTCP. What distinguishes DCTCP from TCP with RED/ECN? Circle all that apply.

a. DCTCP implements a central service to actively optimize the queue length fo each switch.
b. DCTCP can be implemented in commercially-available switches
c. DCTCP automatically determines the packet-marking threshold K
d. DCTCP does not adjust the size of the congestion window
e. None of the above

DCTCP (again)

A sender S is in the midst of sending data to a receiver using a window-based transport protocol. Kevin observes how S's window changes over time. For each of the observations below, decide whether S could be using TCP or DCTCP (or neither). If the sender could be using *either* TCP or DCTCP, circle both of them. Assume the TCP sender is using an AIMD strategy to adjust its window.

a. S's window is currently 100 packets. It then sends a window's worth of data, and in response to the corresponding ACKs, adjusts its window to 70 packets.
b. S's window is currently 100 packets. It then sends a window's worth of data, an din response to the corresponding ACKs, adjusts its window to 50 packets.
c. S's window is currently 100 packets. It then sends a window's worth of data, and in response to the corresponding ACKs, adjusts its window to 10 packets.

TCP

Consider a reliable transport protocol KTP that uses sequence numbers and ACKs the same way that TCP does, but **only** retransmits packets after a timeout; there is no fast-retransmit/fast-recovery mechanism. KTP also does not have a congestion control mechanism, and instead uses a fixed window size.

Phoebe is testing out two different configurations of KTP. In the first — call it C1 — she sets the retransmission timeout value (the RTO) to .95 seconds. In the second — call it C2 — she sets the RTO to 1.05 seconds.

Question 1: Consider a KTP connection between a sender S and a receiver R. Assume that the round-trip-time between S and R is one second and does not change over the course of the transmission. Which scenario **best** describes the difference between the two configurations of KTP?

a. S will send more packets in the configuration C1 and C2, but only if there is loss on the network.
b. S will send more packets in configuration C1 than C2 even if there is *not* loss, but it will only send a few more packets in C1 than in C2.
c. S will send more packets in configuration C1 than C2 even if there is *not* loss, and it will send significantly more packets in C1 than in C2.
d. There will be no difference between the transmissions in either configuration.

Design question (not from a past exam): What does this tell us about how TCP should estimate its RTO? Knowing that the actual RTT between a source and a destination will change over time, propose a way for TCP to estimate the RTO. How does your estimate work when there are a few RTT samples that are outliers? What about when the RTT actually changes?

Question 2: Sadhana is also experimenting with KTP. She decides to do away with the fixed window size and add TCP's AIMD congestion control mechanism, but she's skeptical about slow-start and decides to skip it (i.e., for an entire connection, her senders will adjust their window according to AIMD). Which of the below scenarios best describes Sadhana's KTP setup?

a. Short connections that only need to send a few packets will take longer to complete than if she used slow-start.
b. In general, senders will see more loss than if she used slow-start
c. Connections will reach the optimal window size more quickly than if she were using slow-start.

Queues

Consider a network where two senders, S1 and S2, are sending packets into a switch R, which then sends them along to a destination D.

R has two queues: one for packets from S1 and another for packets from S2. Each of these queues can hold 1000 packets. The switch uses a scheduling algorithm to decide how to send packets from these queues to D, and attempts to give each sender an equal share of the bandwidth over time.

S1 sends 1000 packets to R. Each of those packets is between 5 and 15 bytes long; the average packet size is ten bytes. S2 sends 100 packets to R. Each packet is between 50 and 150 bytes long; the average packet size of 100 bytes.

Question 1: You are able to observe packets on the link between R and D. For each of the following observations, decide whether R could be using Deficit Round Robin (DRR), and explain why (either by giving an example DRR setup that works in this scenario or proving that one doesn't exist). Treat each observation independently. You can assume that in each case, the quantums for both queues are equal, but you do not know the precise value. The value may also be different from one observation to the next (i.e., the quantum in part A may not be the same as in part B).

a. Ten packets from S1, followed by one packet from S2, followed by ten packets from S1, followed by one packet from S2.
b. One packet from S1, followed by one packet from S2, followed by one packet from S1, followed by one packet from S2.
c. 1000 packets from S1, followed by 100 packets from S2.

Design question: What are the consequences of the scenario in part C, with the large quantum? When does this work poorly? What would the effects be? What does this tell us about how we should set quantums in DRR?