Homework 1

CS425/ECE428 Spring 2025

Due: Friday, Feb 21 at 11:59 p.m.

1. Consider a distributed system of four processes as shown in Figure 1. The system is synchronous, and the minimum and maximum network delays (in seconds) between each pair of processes are shown in the figure as $[min, max]$ against each channel. Assume no messages are lost on the channel, and the processing time at each process is negligible compared to network delays.

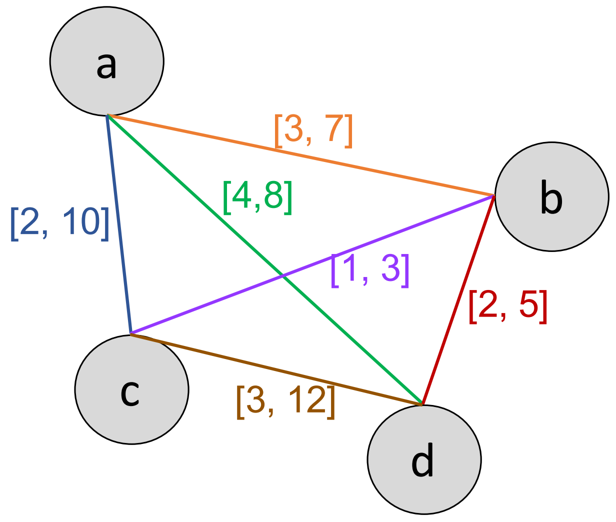

Figure 1: Figure for question 1.

   (a) (2 points) Consider an all-to-all heartbeat protocol, where each process sends a heartbeat to each other process periodically every T=200s, and each process sets a timeout (computed appropriately from the known bounds on network delay) to detect failure of other processes.

   Suppose that process a crashes. For every other process, calculate how long it will take to detect a's failure, in the worst case.

   (b) (2 points) Now consider a small extension of the protocol in Q1(a) – as soon as a process detects that another process p has crashed via a timeout, it sends a notification to all other processes about p's failure. Suppose that process a is the only process that crashes. For every other process, calculate how long it will take to detect a's failure, in the worst case.

   (c) (3 points) Suppose it is known that *no more than two* processes in the above system can crash simultaneously. (i) What is the smallest number of processes that each process p must send heartbeats to, so as to ensure that p's failure is detected by at least one alive process? (ii) Now specifically consider process a. List the smallest *set* of processes that a must send heartbeats to, such that the worst-case time to detect a's failure by at least one alive process is the same as what it would have been with all-to-all heartbeats.

2. Consider Figure 2. The client has an option of using any of the three authoritative sources of real time (s1, s2, or s3) for external synchronization via Cristian algorithm. The round-trip times (RTT) between the client and the three servers are shown in the figure. Assume that the observed RTT to each server remains constant across all synchronization attempts. There are no known bounds on minimum one-way network delay (assume one-way delay ≥ 0).

   (a) (1 point) Which server should the client choose to achieve the lowest accuracy bound right after synchronization? What is the value of this bound, as estimated by the client right after synchronization?

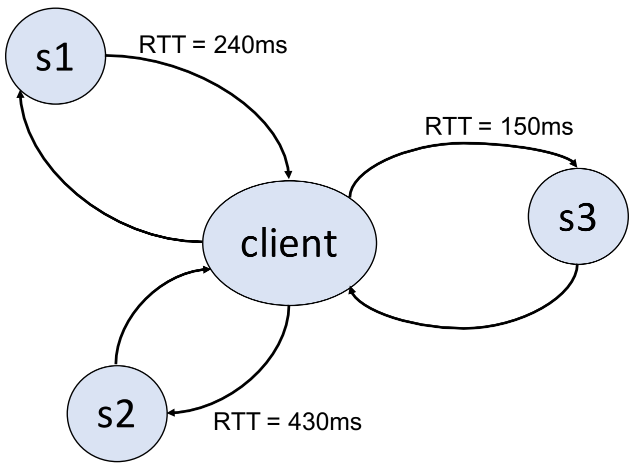

Figure 2: Figure for question 2.

(b) (2 points) If the client's local clock drifts at the rate of 15μs every second, what is the longest time period (in seconds) at which the client must initiate synchronization with the server it chose in part (a), so as to maintain an accuracy bound within 450ms at all times? In other words, you need to compute the longest time difference between two consecutive synchronization requests sent by the client to the server, such that the accuracy bound is always within 450ms.

3. Consider the series of messages exchanged between two servers A and B as shown in Figure 3. The local timestamps (in seconds) at the servers, when sending and receiving each message, are shown in the figure. A wishes to compute its offset relative to B using NTP's symmetric mode synchronization, based on its knowledge of the send and receive timestamps for all six messages. (Note, A's offset relative to B is defined as "local time at A - local time at B".)

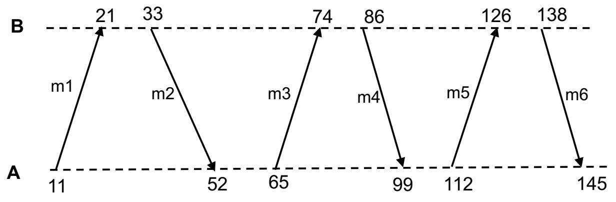

Figure 3: Figure for question 3.

(a) (1 point) Using the send and receive timestamps of which message pair will result in the lowest synchronization bound (as estimated by A)?

(b) (2 points) What is the corresponding synchronization bound ($d_i/2$)?

(c) (2 points) What is the corresponding estimated offset value (o_i)?

(d) (2 points) Now assume that A uses the same series of messages for synchronization via Cristian algorithm: messages sent from A to B are requests, and messages from B to A are responses. A uses its local send and receive timestamps to compute RTT (round-trip time). What is the tightest synchronization bound (as estimated by A) with which A can compute its offset relative to B?

4. The timeline in Figure 4 shows 16 events (A to P) across four processes. The numbers below indicate real time.

(a) (3 points) Write down the Lamport timestamp of each event.

(b) (5 points) Write down the vector timestamp of each event.

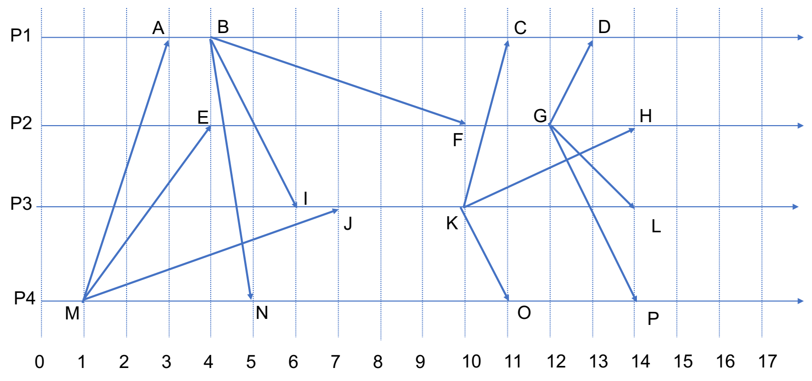

Figure 4: Timeline for questions 4 and 5.

(c) (2 points) List all events considered concurrent with (i) C, and (ii) F.

5. (a) (5 points) Consider the timeline and events in Figure 4 again. Suppose that P3 initiates the Chandy-Lamport snapshot algorithm at (real) time 8. Assuming FIFO channels, write down *all* possible consistent cuts that the resulting snapshot could capture. You can describe each cut by its frontier events.

   (b) (3 points) Write *all* possible states of the incoming channels at P2 that the above snapshot could record, as asked below:

   (i) All possible channel states from P1 to P2.

   (ii) All possible channel states from P3 to P2.

   (iii) All possible channel states from P4 to P2.

   You can denote the pending messages on a channel by their send and receive event ids, and denote a channel state with no pending messages as "empty".

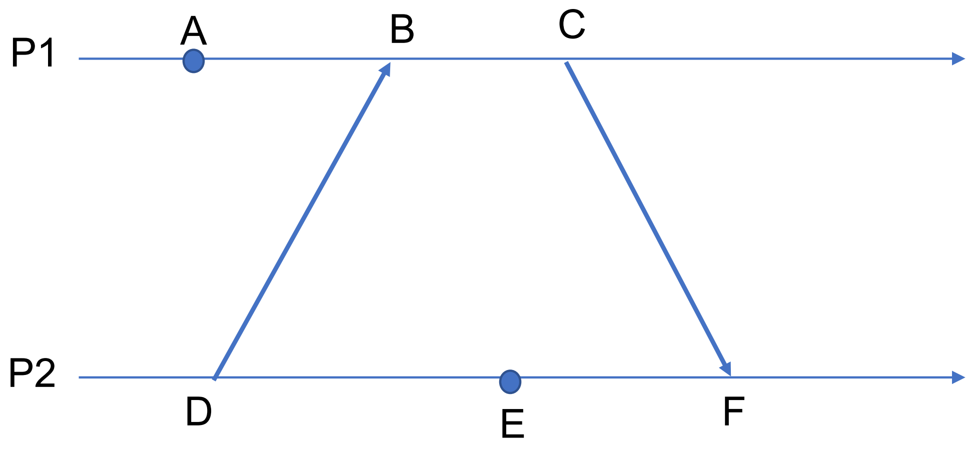

Figure 5: for question 6

6. (a) (2 points) Consider the timeline of events {A,B,...F} across two processes as shown in Figure 5. List all possible linearizations for this system that includes each event.

   (b) (3 points) What is the total number of consistent global states that can be possibly captured for the above system? Identify each of them by the frontier events of the corresponding cuts.