Homework 1

CS425/ECE428 Spring 2026

Due: Friday, Feb 20 at 11:59 p.m.

1. Consider a distributed system of five processes as shown in Figure 1. The system is synchronous, and all processes have a communication channel with every other process. The min and max network delay (in seconds) from process 'a' to every other process is shown in the figure (network delays between other processes are not needed for this question, and are therefore not shown in the figure). Assume no messages are lost on the channel, and the processing time at each process is negligible compared to network delays.

   Consider a failure detection protocol, where each process sends a heartbeat to **three** other processes periodically every T=100s. Specifically, each process sends heartbeats to the same set of three other processes each time.

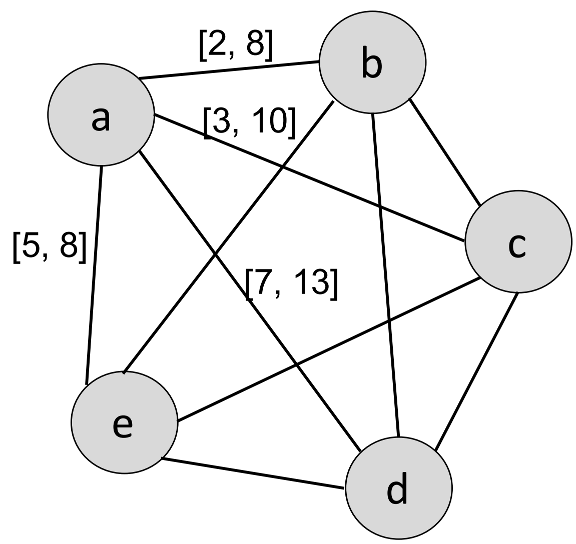

Figure 1: Figure for question 1.

   (a) (2 points) What is the maximum number of simultaneous process failures that the system can tolerate, while ensuring complete failure detection?

   (b) (2 points) Suppose process a crashes, and it is the only crashed process. Process a sending heartbeats to which set of three processes would result in highest worst-case failure detection time? What is the value of this highest worst-case failure detection time?

2. A client repeatedly synchronizes its clock with an authoritative server using Cristian algorithm. It sends a synchronization request every 6 hours and resets its local clock as per the Cristian algorithm as soon as it receives the corresponding response from the server. The observed round-trip times (RTT) between the client and the server is 250ms (it remains the same across all synchronization attempts). There are no known bounds on minimum one-way network delay (i.e. assume one-way delay ≥ 0). The client's local clock drifts at the rate of 20μs every second. Assume that the authoritative server has negligible drift rate. Consider the 6 hour time interval between the k^{th} and the $(k+1)^{th}$ requests sent by the client, for any $k > 1$.

   (a) (1 point) At what point of time during this interval is the skew between the client and the server lowest? [Answer in words. Precise time value not expected.]

   (b) (1 point) What is the upper (worst-case) bound of this lowest skew? [Round your response to closest integral milliseconds.]

(c) (1 point) At what point of time during this interval is the skew between the client and the server highest? [Answer in words. Precise time value not expected.]

(d) (2 points) What is the upper (worst-case) bound of this highest skew? [Round your response to closest integral milliseconds.]

3. Consider the series of messages exchanged between two servers A and B as shown in Figure 2. 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".)

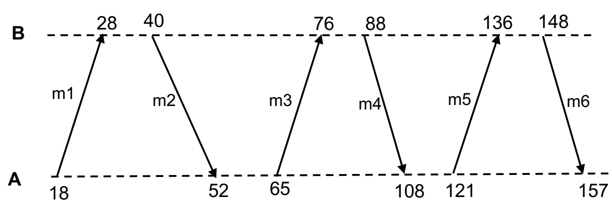

Figure 2: Figure for question 3.

(a) (1 point) Using the send and receive timestamps of which message pair will result in the lowest synchronization bound (as estimated by A)? [Hint: the two messages need not be consecutive.]

(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 series of messages shown in Figure 2 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 3 shows 16 events (A to P) across four processes. The numbers below indicate real time.

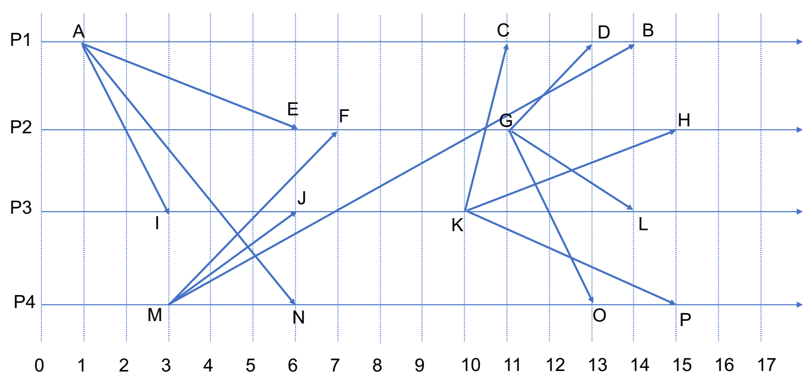

Figure 3: Timeline for questions 4 and 5.

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

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

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

5. (a) (6 points) Consider the timeline and events in Figure 3 again. Suppose that P2 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 P1 that the above snapshot could record, as asked below:

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

      (ii) All possible channel states from P3 to P1.

      (iii) All possible channel states from P4 to P1.

   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".

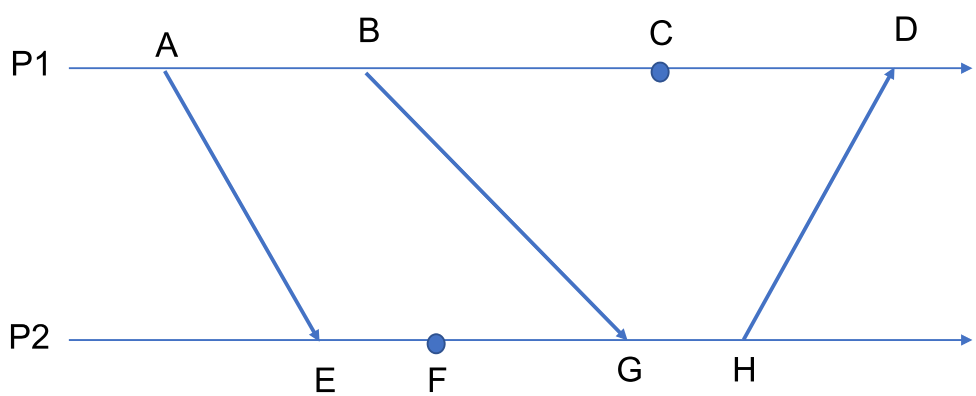

Figure 4: for question 6

6. (a) (3 points) Consider the timeline of events {A,B,...H} across two processes as shown in Figure 4. List all possible linearizations for this system that includes each event.

   (b) (2 points) What is the total number of consistent global states that can be possibly captured for the above system (including the null state with no events)? Identify each of them by the frontier events of the corresponding cuts.