Homework 2

CS425/ECE428 Spring 2022

Due: Wednesday, Feb 23 at 11:59 p.m.

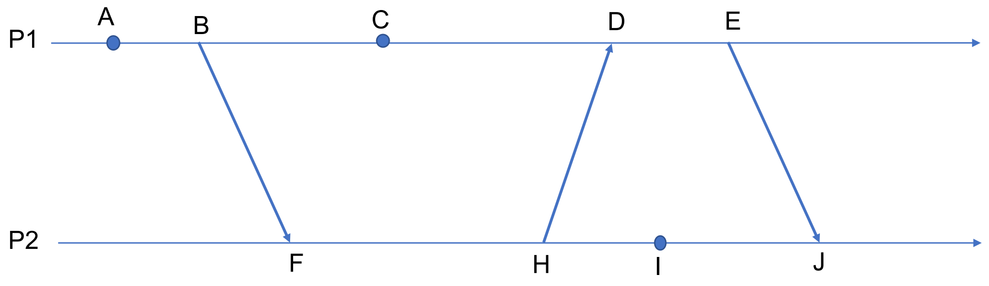

Figure 1: for question 1

1. (a) (4 points) Consider the timeline of events {A,B,...J} across two processes as shown in Figure 1. List all possible linearizations for this system that includes each event.

   (b) (5 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.

   (c) (1 point) Provide an example of an unstable global safety property (which results in unstable non-safety). How can it be made stable? *(This is unrelated to 1(a) and 1(b).)*

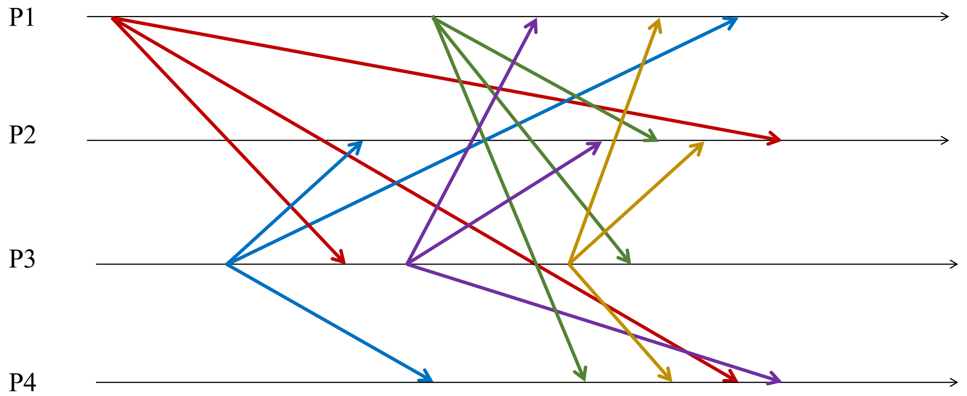

Figure 2: for question 2(a)

2. (a) (6 points) In the execution in Figure 2, processes send messages to each other to implement *FIFO ordered* multicast. To simplify the picture, messages sent by each process to itself are not shown, but assume that such messages are received and delivered instantaneously. For the questions below, you may use printed or hand-drawn figure with hand-drawn responses, or digitally edit the figure from the homework PDF.

   (i) Identify the messages that are buffered at the processes to ensure FIFO multicast delivery. (Circle the receive event for the buffered messages to identify those messages.) *(3 points)*

(ii) For each message buffered as above, determine the earliest instant of time at which the message may be delivered, while ensuring FIFO multicast. (To identify the instant of time draw an arrow that begins at the time when the message is received to the time at which the message may be delivered.) *(3 points)*

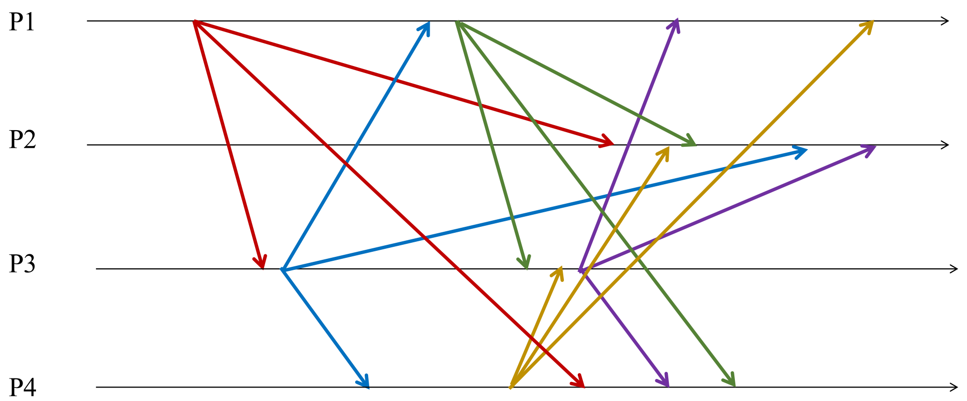

Figure 3: for question 2(b)

(b) (6 points) In the execution in Figure 3, processes send messages to each other to implement *causal multicast*. To simplify the picture, messages sent by each process to itself are not shown, but assume that such messages are received and delivered instantaneously. For the questions below, you may use printed or hand-drawn figure with hand-drawn responses, or digitally edit the figure from the homework PDF.

(i) Identify the messages that are buffered at the processes to ensure causally-ordered multicast delivery (Circle the receive event for the buffered messages to identify those messages.) *(3 points)*

(ii) For each message buffered as above, determine the earliest instant of time at which the message may be delivered, while ensuring causally-ordered multicast. (To identify the instant of time draw an arrow that begins at the time when the message is received to the time at which the message may be delivered.) *(3 points)*

3. For each of the statements below, identify whether it is *true* or *false*. If it is false, present a counter-example. If it is true, prove why.

(a) (2 points) A total ordered multicast is also causal.

(b) (3 points) If the processes in a system use R-multicast, and each *channel* follows FIFO order, then causal ordering is satisfied.

(c) (3 points) We can implement the ISIS algorithm for total ordering on top of (or using) causal-ordered multicast, to achieve a total causal multicast.

Process ID	Time when “enter” is called (since start of system)	Time spent in critical section after “enter” returns, before calling “exit”.
P_1	250ms	50ms
P_2	30ms	50ms
P_3	100ms	30ms
P_4	20ms	60ms
P_5	15ms	20ms

Table 1: Timings for Q4

4. Consider a distributed system of five processes $\{P_1, P_2, P_3, P_4, P_5\}$. Each process needs mutually exclusive access to a critical section. Table 1 lists the time when each process first makes a blocking call to “enter” the critical section (since the start of the system). It also lists the time each process spends in the critical section after “enter” succeeds, before calling “exit”.

   (a) (5 points) Suppose the system uses the central server algorithm for mutual exclusion, electing P_4 as the leader. Assume that message latency at P_4 for communicating with the leader (itself) is zero, i.e. it takes negligible time for P_4's token request to reach the leader upon calling enter, to receive the token after its request has been granted, and for the token to be released back to the leader upon calling exit. For all other processes, assume the one-way network latency for communicating with the leader (P_4) is fixed at 10ms, i.e. it takes 10ms each for the token request to reach P_4 after calling “enter”, 10ms for the token to reach the process after the leader has granted the request, and 10ms for the token to reach the leader after the process has called “exit”. The leader grants requests in the order in which it receives them. When will each process start executing its critical section?

   (b) (5 points) Now suppose that the system uses ring-based algorithm for mutual exclusion, with the ring structured as shown below (P1 to P2 to P3 to P4 to P5 to P1).

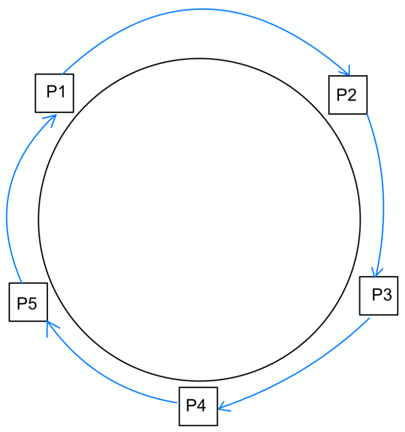

Figure 4

At time 0ms (when the system starts up), the token is at P_1. The network latency for passing the token from a given process to its ring successor is fixed at 10ms. When will each process start executing its critical section?