Homework 2

CS425/ECE428 Spring 2021

Due: Thursday, March 4 at 11:59 p.m.

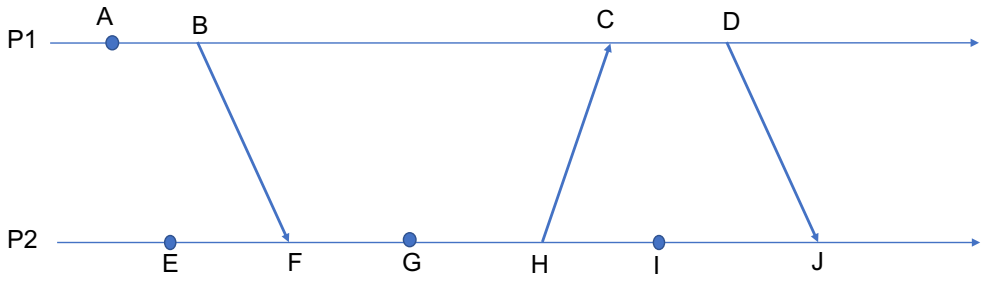

Figure 1: for question 1

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.

2. (a) (1 point) Consider an implementation of Chandy-Lamport algorithm where the local state of each process (which includes a detailed log of all send and receive events in the corresponding cut) is gathered at a central server. Assume all requirements for Chandy-Lamport algorithm hold, i.e. each channel follows FIFO order, no messages are dropped, and no process fails. It is possible to analyze the event logs at the central server to determine the channel state, instead of recording the incoming channel states at each process – *state whether this is True or False (explanation not required).*

   (b) (3 points) Consider a modification to the Chandy-Lamport algorithm where it is desirable to find a consistent cut, and only record the local state at each process for the cut, and not the channel state. In this modification, the initiator process records its state and sends a marker to all other processes. Each process records its state upon receiving the marker from the initiator process. The algorithm terminates when all processes have received a marker from the initiator process. Will this modification result in a consistent cut? If yes, prove why. If not, present a counter-example. Assume all requirements for Chandy-Lamport algorithm hold (i.e. each channel follows FIFO order, no messages are dropped, and no process fails).

   (c) (2 points) Provide an example of an unstable global safety property (which results in unstable non-safety). How can it be made stable?

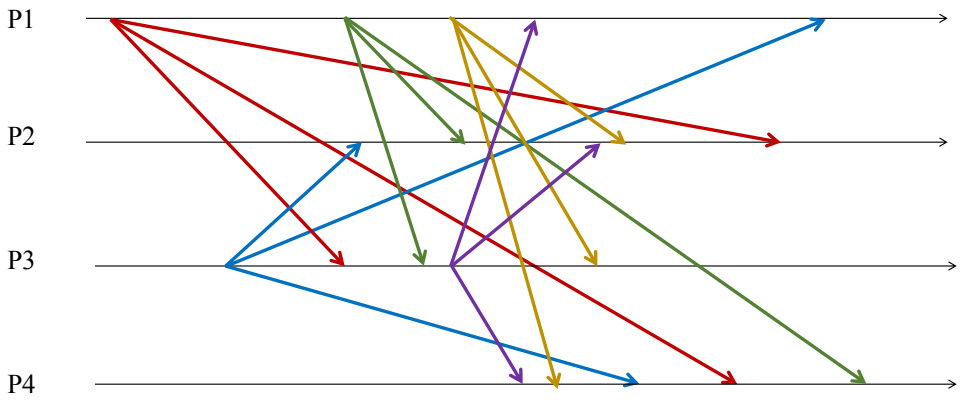

Figure 2: for question 3(a)

3. (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)*

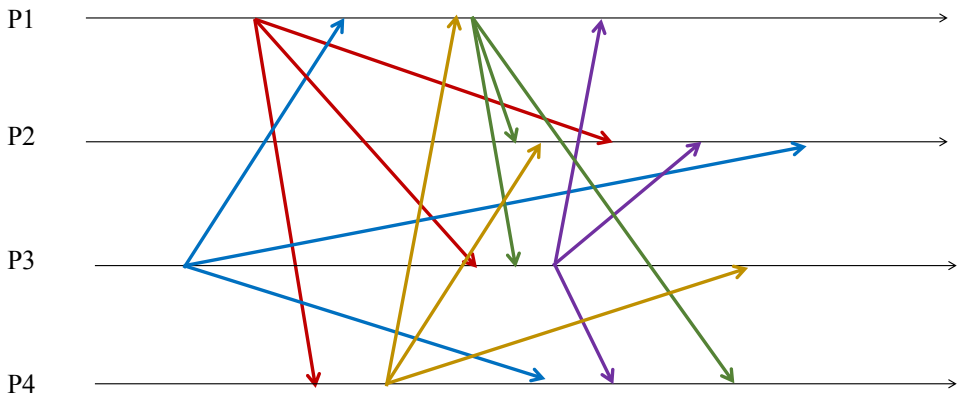

Figure 3: for question 3(b)

(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)*

4. 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 causal-ordered multicast is also FIFO ordered.

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

   (c) (3 points) A FIFO + total multicast is also causal, given that all processes in the system communicate only via the multicast messages, and do not exchange any other messages outside of multicasting.

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

   (e) (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.