Due: Sept. 27, 2 p.m.

Problem 1

Ring heartbeating may not detect simultaneous multiple failures of processors.

1. What is the maximum number of simultaneous processor failures that can be detected by ring heartbeating protocol?

   Answer: Only one process failure is guaranteed to be detected by ring heartbeating protocol. If for example two consecutive processes in the ring fail simultaneously, only failure of the last process is detected by the protocol.

2. Modify the ring heartbeating protocol to detect up to 4 simultaneous processor failures.

   Answer: In the ring heartbeating protocol, each process sends its heartbeat to only one process. In order to be able to detect up to 4 **simultaneous process failures, each process should send its heartbeat to** 4 **other processes.**

Problem 2

(Problem 2.14 from the book) Consider two communication services for use in asynchronous distributed systems. In service A, messages may be lost, duplicated or delayed and checksums apply only to headers. In service B, messages may be lost, delayed, or delivered too fast for the recipient to handle them, but those that are delivered arrive with the correct contents.

1. Describe the classes of failure exhibited by each service.

   Answer:
   Service A:
   Message loss : communication omission failure.

   Duplicate message : arbitrary/Byzantine failure.

   Message delay : in time-sensitive systems, this would be timing failure, but since we are considering asynchronous systems, message delay does not matter as long as message gets delivered and it is not considered a failure. If only message delay happens, and we have the asynchronous system, delay will not cause drop of the message.

   Header checksum only : this scenario can cause that the payload be corrupted during the communication, hence we may see arbitrary failure.

   Service B:

   Message loss : communication omission failure.

   Message delay : since we deal with asynchronous systems, delayed messages do not cause

Problem 2 continued on next page. . .

any failures.

Too fast message delivery : If the receiver drops the messages because they are delivered too fast, it would be communication omission failure.

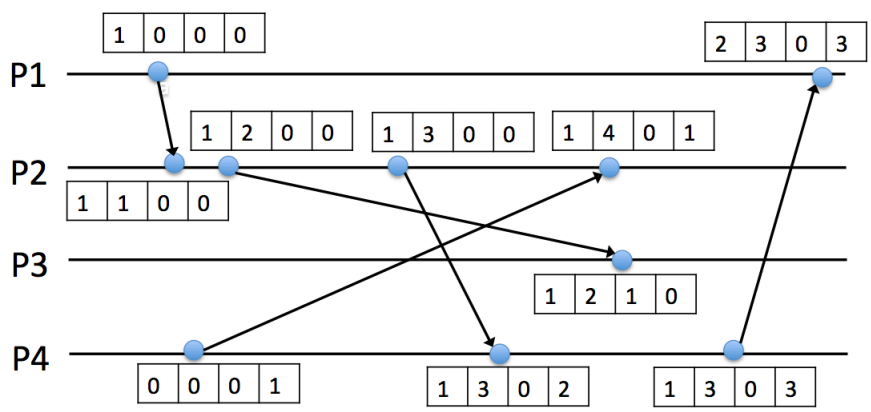
2. Can service B be described as a reliable communication service?

No. Message loss violates validity, and so service B is not reliable.

Problem 3

P1, P2, P3, and P4 are four processes. Write down the vector logical timestamps in the boxes attached to each event.

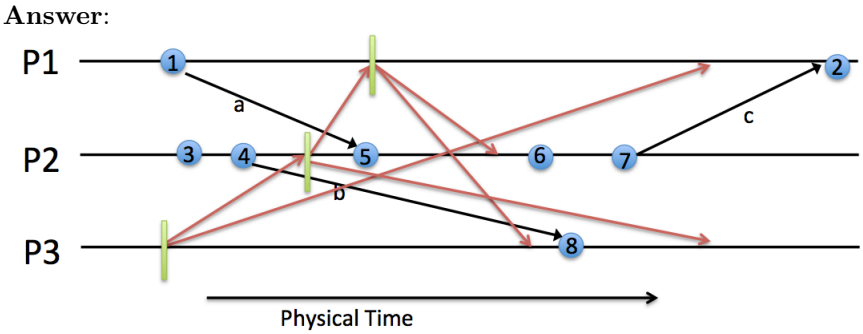
Answer:



Problem 4

Process P3 initiates the snapshot algorithm. The black arrows are messages sent and received. The red arrows are marker messages. Find out the consistent cut corresponding to this global snapshot and mark the states of each process and channel.

Answer:



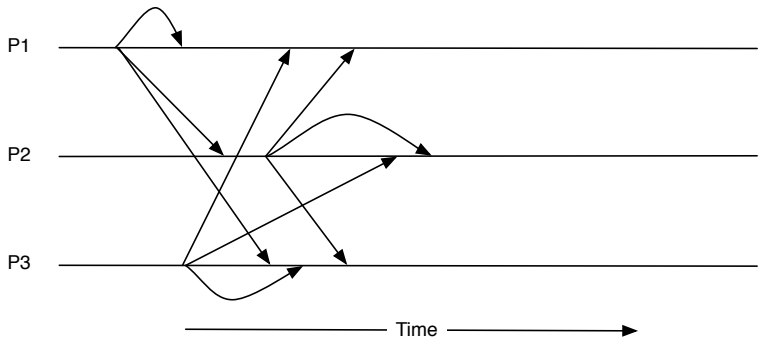
The cut joins the green points on the figure. The states captured by this snapshot algorithm are:

- P1: 1

- P2: 3,4
- P3 : {}
- C12: a
- C13: {}
- C21: {}
- C23: b
- C31:{}
- C32: {}

Problem 5

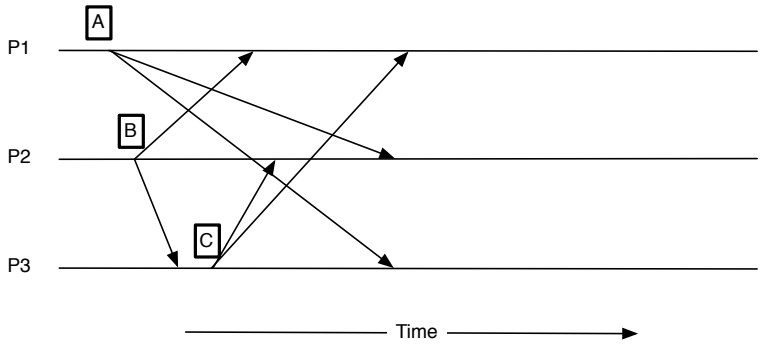
Consider multicast messages sent and received using the order illustrated below. What ordering does this example follow? (a) FIFO (b) Causal (c) Total (d) FIFO-Total (e) Causal-Total.



Answer: Answer is (e) Causal-Total. It is FIFO since the condition is not violated. It is causal, total and also FIFO-Total. You got points if you mentioned all the answers or (e).

Problem 6

The figure below illustrates message flow in a multicast group that uses the Isis total ordering algorithm. Processes P1, P2, P3 each multicast message A,B,C, respectively. The figure shows the transmission and reception times of each initial message transmission (i.e., step 1 of the Isis algorithm).



1. Show the priority queues at each process at the completion of the messages shown and the proposed priority that each process assigns to each message.

   P1: A:1, B:2, C:3
   P2: B:1, C:2, A:3
   P3: B:1, C:2, A:3

2. What is the (total) order that the messages will be delivered in?

   max(B) = B2.1
   max(A) = A3.3
   max(C) = C3.1

   So order is B, C, A

3. Prove or disprove: *The Isis total ordering system* always *produces a FIFO total order.*

   The statement is false. The following figure shows a counterexample that disproves this statement.
   In this figure, priority queues at two processes are:
   P1: A:1, B:2
   P2: B:1, A:2

   The agreed priorities are:
   max(A) = 2.2
   max(B) = 2.1

   So, the total order is B,A. This violates FIFO order since A is issued before B.