George Suarez
CSE 461

Homework 4

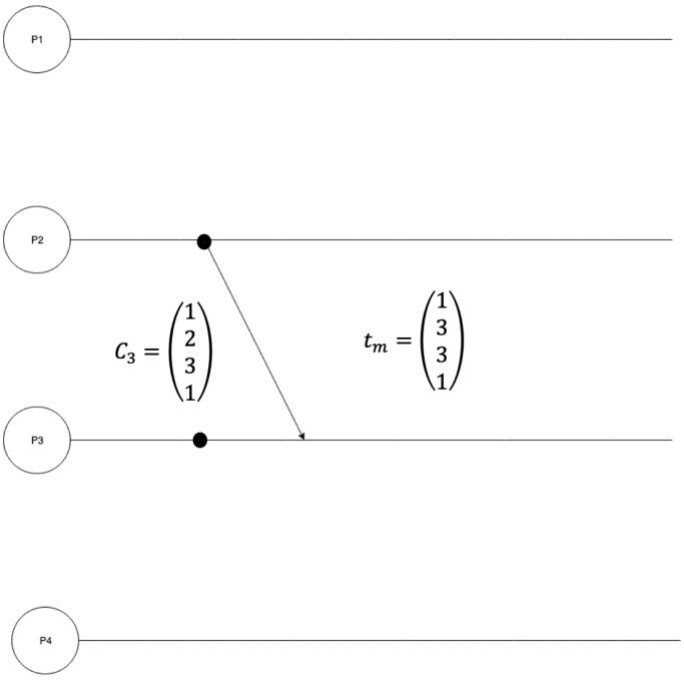
1. Suppose the "Birman-Schiper-Stephenson Protocol" is used to enforce "Causal Ordering of Messages" of a system that has four processes P_1, P_2, P_3, and P_4. With the help of diagrams. Explain clearly what the process would do in each of the following cases.

   a) The current vector time of Process P_3 was C_3 when it received a message from P_2 along with vector time stamp t_m, where:

   $$C_3 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 1 \end{pmatrix} \qquad t_m = \begin{pmatrix} 1 \\ 3 \\ 3 \\ 1 \end{pmatrix}$$

   Should P_3 deliver the message immediately? Why? If not, what should it do?

   Solution:



   Process P_3 should deliver the message immediately

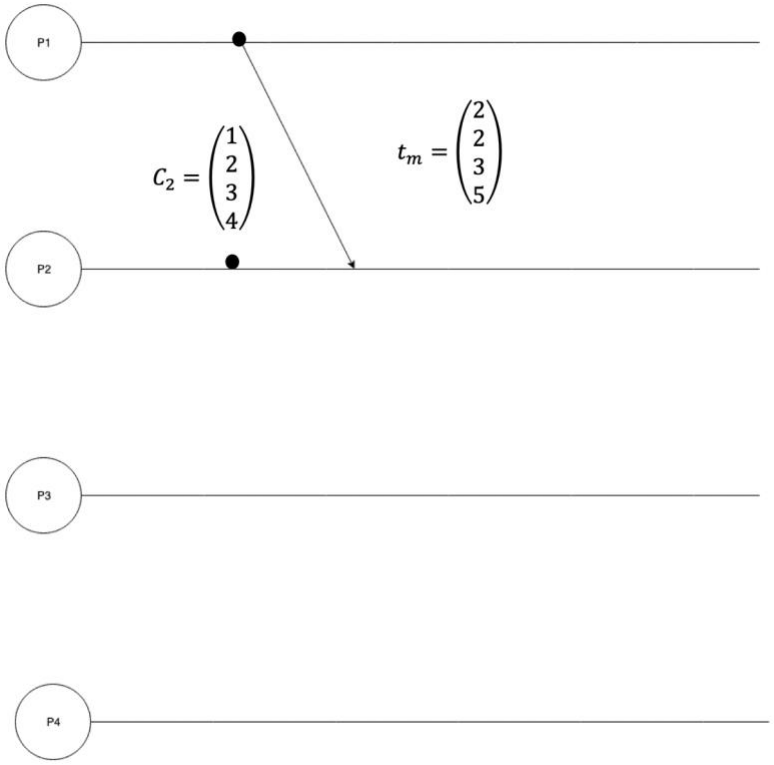
   $C_3[2] = t_m[2] - 1$ and $C_3[k] \geq t_m[k]$ where $k = 1, 3, 4$

b) Process P_2 with current vector time C_2 received a message from P_1 along with vector time stamp t_m, where

$$C_2 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \qquad t_m = \begin{pmatrix} 2 \\ 2 \\ 3 \\ 5 \end{pmatrix}$$

Should P_2 deliver the message immediately? Why? If not, what should it do?

Solution:



$C_2[1] = t_m[1] - 1 = 1 => 1 \geq 1$
$C_2[2] = t_m[2] => 2 \geq 2$
$C_2[3] = t_m[3] => 3 \geq 3$
$C_2[4] = t_m[4] => 4 \geq 5$

Since $C_2[4] \geq t_m[4]$ is false, then the message is being held in a buffer until P_2 receives a message from P_4 first. Therefore, P_2 should not deliver the message immediately.

2. Consider a cut **C**:

$$\boldsymbol{C} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

Where c_1, c_2, and c_3 are the cut events with vector clocks $\boldsymbol{C_1}, \boldsymbol{C_2}, \boldsymbol{C_3}$ respectively:

$$\boldsymbol{C_1} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \quad \boldsymbol{C_2} = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} \quad \boldsymbol{C_3} = \begin{pmatrix} 0 \\ 1 \\ 3 \end{pmatrix}$$

Calculate $T_C = sup(C_1, C_2, C_3)$. Is **C** a consistent cut? Why?

Solution:

$$T_c = max \begin{pmatrix} C_1[1] \\ C_2[2] \\ C_2[3] \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 3 \end{pmatrix}$$

Since $C_1[1] = 1 => 1 \neq 2$, then **C** is an inconsistent cut because P_2 received a message from P_1, but P_1 has not sent the message yet.

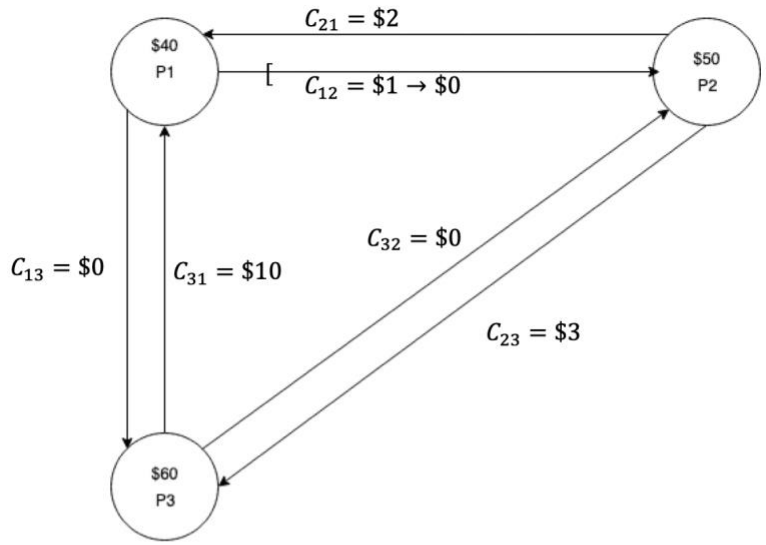
3. A banking system uses Chandy-Lamport global state recording protocol (Snapshot Algorithm) to record its global state; markers are sent along channels where FIFO is assumed. The system has three branches P_1, P_2, and P_3 and are connected by communication channels C_{ij}, where i, $j = 1, 2, 3$. Suppose LS_i denote the local state (the money the branch possesses at the time of recording) of branch P_i.

   P_1 initiated the recording process. Right before P_1 sent out the marker, a \$1 transaction was in transit on C_{12}, a \$2 transit on C_{21}, a \$3 transit on C_{23}, and a \$10 transit on C_{31}(assume that the units are in million dollars) and branches P_1, P_2, and P_3 had \$40, \$50, and \$60 respectively (not including any money in transit). Assume that the branches do not send out any other money during the whole recording process and the markers from P_1 arrived at other banks earlier than other markers. With the help of diagrams, find out the state LS_i of P_i and channel states C_{ij} where $i, j = 1, 2, 3$. Tabulate your results in the following format:

State	Money
LS_1	..
LS_2	..
LS_3	..
C_{12}	..
C_{13}	..
C_{21}	..
C_{23}	..
C_{31}	..
C_{32}	..

Show your steps clearly.

Solution:



State	Money
LS_1	$40
LS_2	$51
LS_3	$60
C_{12}	$0
C_{21}	$2
C_{23}	$3
C_{32}	$0
C_{13}	$0
C_{31}	$10

4. In Lamport's algorithm for mutual exclusion, Process P_i enters CS when the following 2 conditions are satisfied:

    1) $P_i's$ request is at the head of $request - queue_i$
    2) P_i has received a (REPLY) message from every other process time-stamped later than t_{s_i}

    Condition 1) can hold concurrently at several sites. Why then is 1) needed to guarantee mutual exclusion?

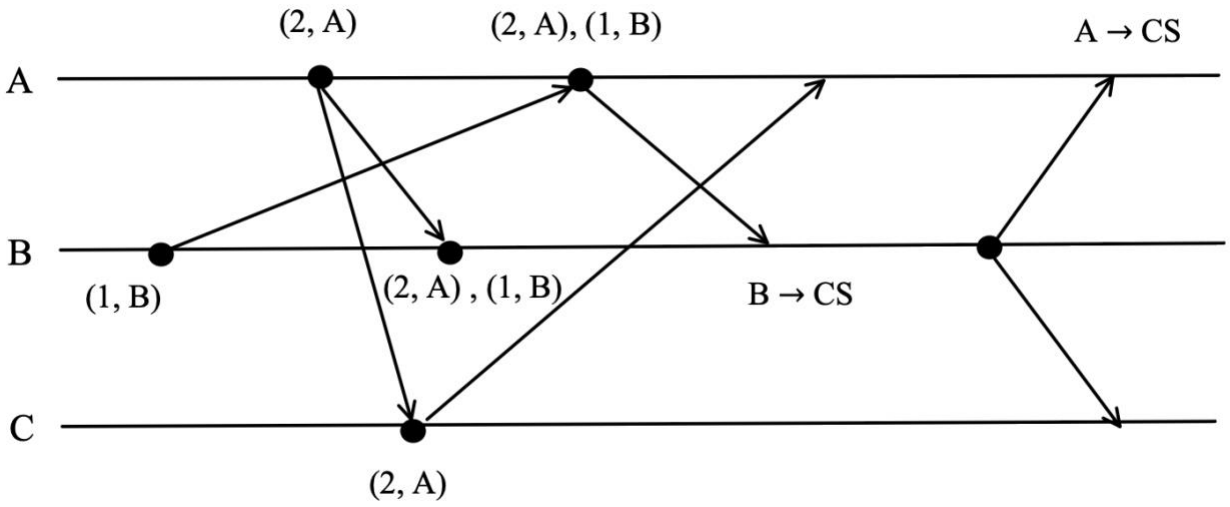
    Solution:

    We need condition 1 to guarantee mutual exclusion because each process needs to compare the time stamp of its' own request with the time stamps of the other processes to check that only one process is at the head of the request-queue. Without the request-queue, two different processes might think that they are at the head of the queue and make condition 2 not possible.

    Does the algorithm work if condition 2) is removed? Why? Give an example with illustrations (drawings) to support your argument.

    Solution:

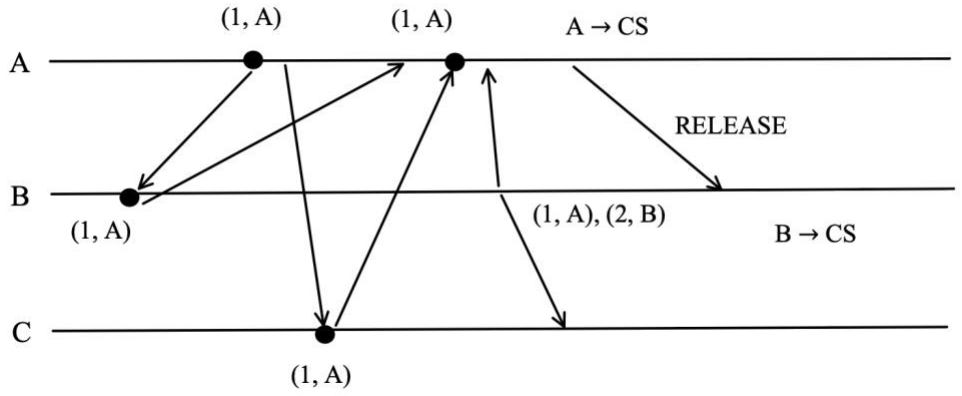
    The algorithm still works if condition 2 is removed because a RELEASE message can be interpreted as a REPLY message.



5. In Lamport's algorithm of mutual exclusion, if a site S_i is executing the critical section, is it necessary that $S_i's$ request need to be always at the top of the request-queue at another site S_j? Explain and give an example (with diagrams) to support your argument.

   Solution:

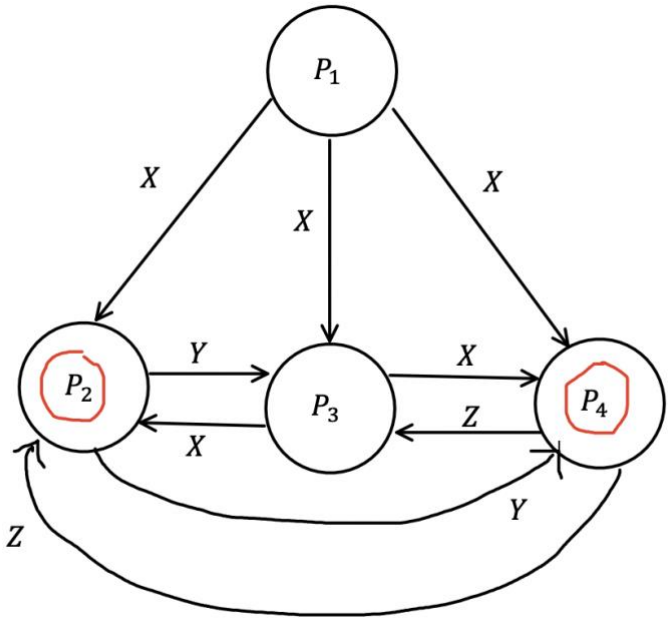
   It is not necessary that S_i's request always need to be at the top of the request-queue at another site S_j because S_i already visited the critical section, and then process P_i removes its' request from the head of its' request-queue and sends a time stamp RELEASE to every other process.



6. Can Byzantine agreement be always be reached among four processors if two processors are faulty? With the help of diagrams, explain your answer.

   Solution:

   The Byzantine agreement cannot always be reached among four processors if two processors are faulty since the number of faulty processors in this case is $< \frac{1}{3}$ of the total number of processors. In this condition, processer $P_2's$ majority = $\{x, y, z\} = 0$. Therefore, no agreement can be reached.


P_1
X
X
X
P_2
Y
X
P_3
X
Z
P_4
Y
Z