CSE 660 Operating Systems Concepts & Theory

Homework 4
Amit Lawanghare

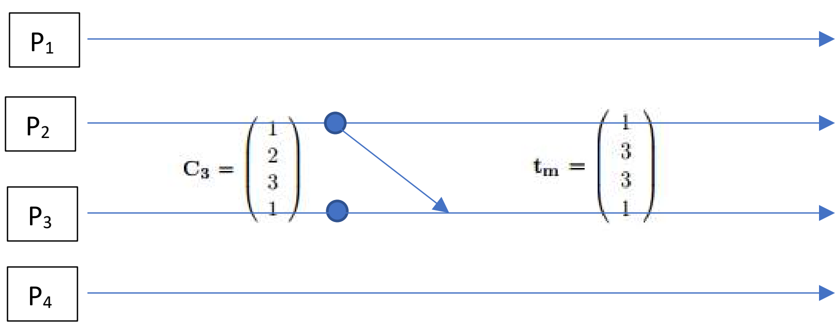
1. Suppose the "Birman-Schiper-Stephenson Protocol" is used to enforce "Causal Ordering of Messages" of a system that has four processes, P_1, P_2, P_3 and P_4. With the help of diagrams, explain clearly what the process would do in each of the following cases.

a) The current vector time of Process P_3 was C_3 when it recieved a message from P_2 along with vector timestamp tm, where

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

Should P3 deliver the message immediately? Why? If not, what should it do?

Ans - P_3 should deliver message immediately: $C_3[2] = t_m[2] - 1$ and $C_3[k] >= t_m[k]$ (k=1,3,4).

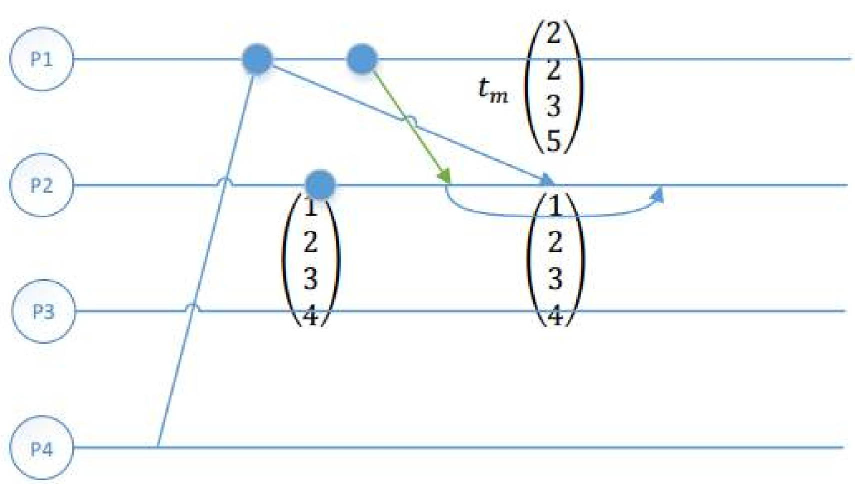


b) Process P_2 with current vector time C_2 received a message from P_1 along with vector time stamp t_m, where

$$C2 = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \qquad t_m = \begin{pmatrix} 2 \\ 2 \\ 3 \\ 5 \end{pmatrix}$$

Should P2 deliver the message immediately? Why? If not, what should it do?

Ans - We have $C_2[1] = t_m[2] - 1$, but $C_2[4] < t_m[4]$. Then P_2 should buffer message and until it receives previous message from P_1.



2. Consider a cut: $C = \begin{pmatrix} c1 \\ c2 \\ c3 \end{pmatrix}$ where c1, c2, and c3 are the cut events with vector clocks C1, C2, C3 respectively: $C1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$ $C2 = \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix}$ $C3 = \begin{pmatrix} 0 \\ 1 \\ 3 \end{pmatrix}$

Calculate TC = sup(C1, C2, C3). Is C a consistemt cut? Why?

Ans -

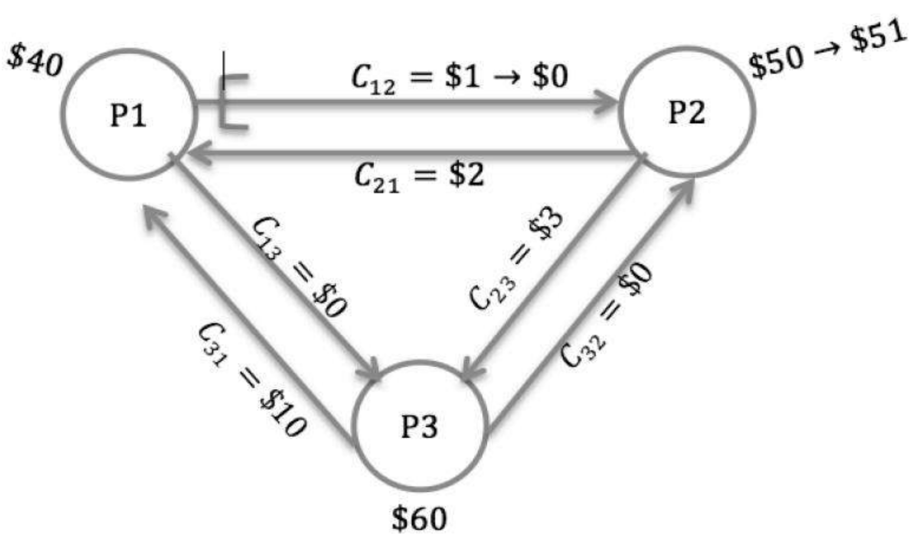
C is inconsistent cut because: Tc > $\begin{pmatrix} C1[1] \\ C2[2] \\ C3[3] \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 3 \end{pmatrix}$

And Since C[i] ≠ T_c[i] .

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:

Ans –



State	Money
LS1	**$40**
LS2	**$51**
LS3	**$60**
C12	**$0**
C13	**$0**
C21	**$2**
C23	**$3**
C31	**$10**
C32	**$0**

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

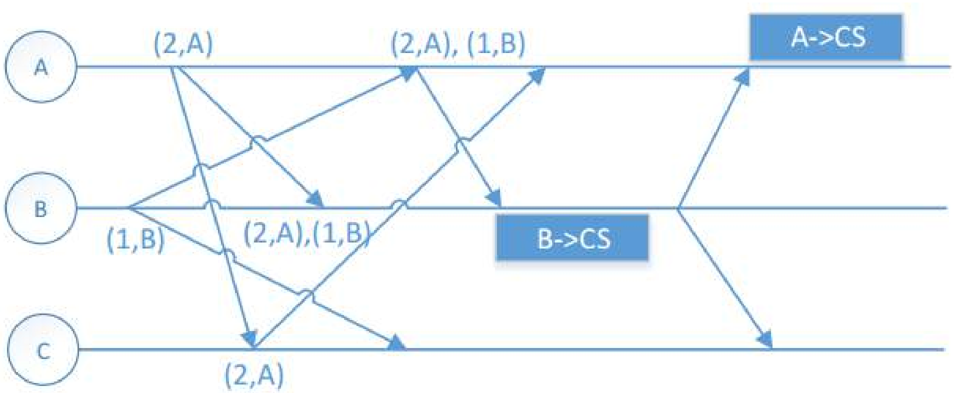
    1) P_i's request is at the head of $requestiqueue_i$
    2) P_i has received a (REPLY) message from every other process time-stamped later than t_{s_i}

    Condition 1) can hold concurrently at several sites. Why then is 1) needed to guarantee mutual exclusion?
    Does the algorithm work if condition 2) is removed? Why? Give an example with illustrations (drawings) to support your argument.

Ans –

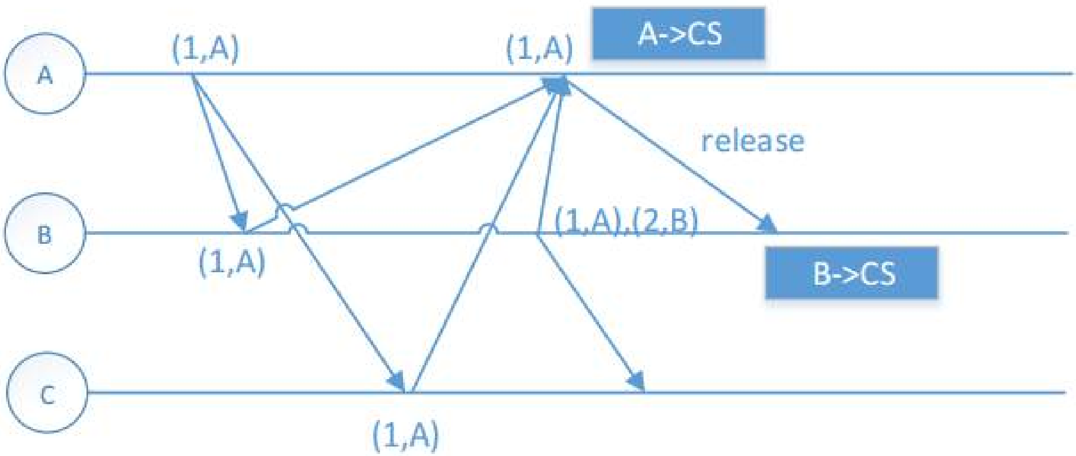
Condition **(1**) is needed to guarantee the mutual exclusion because all the queues that each process has needs to be in synchronization since the each of the process's own queue might not be updated the same as the other (two processes might think that they are at the top of the queue when they are not). Condition **(1)** serves as a substitution of shared memory since they don't have a shared memory.

If condition **(2)** is removed, the algorithm still works under certain case. A Release message serves a reply. A Release message from the process who entered the CS can signal that if a process whose process is second in the queue after the one who entered the CS), then that process can enter the CS next.



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.

It is not necessary. If it is executing the CS, it is been already to CS, then if it can be on the top or not. When exiting the CS, process Pi removes its request from the head of its request-queue and sends a time stamped RELEASE to every other process.



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

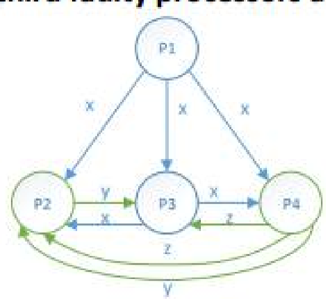
<u>**Answer:**</u>

According to Lamport-Shostack-Pease algorithm the agreement cannot be reached if the number of faulty processors is < 1/3 of the total number of processors. As a solution, a Byzantine agreement can only be reached among four processors iff there is less than one third faulty processors and here the maximum of faulty process should be one.

P2's majority = {x,x,z} = x, but P2 is a traitor so it will retreat

P3's majority = {x,y,z} = retreat

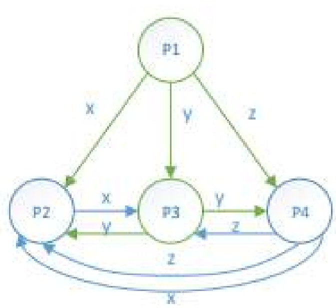
P4's majority = {x,x,y} = x, but P4 is a traitor, so it will retreat



P2's majority = {x,y,z} = Ø

P3's majority = {y,x,z} = Ø

P4's majority = {z,x,y} = Ø



7. Maekawa's Algorithm is used to achieve mutual exclusion for 13 sites. Suppose the sites are labeled 1, 2, ..., 13. Find the request sets R1, R2, ..., R13.

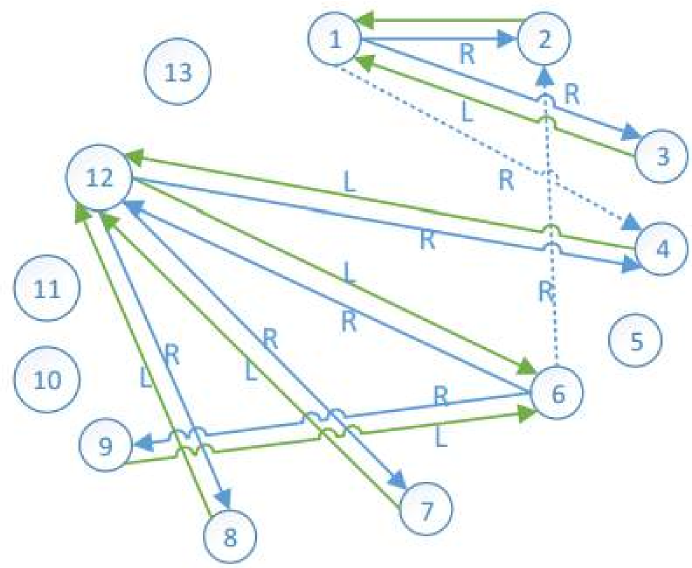
Suppose sites 1, 6, 12 want to enter a critical section (CS) and they have sent requests in the order 1, 6, 12. The following sequence of events have occurred in the order listed:

1. The requests of site 1 have arrived at site 2, and site 3. Its request to site 4 is on the way.
2. The requests of site 6 have arrived at site 9, and site 12. Its request to site 2 is on the way.
3. The requests of site 12 have arrived at site 4, and site 7 and 8.

Draw a diagram to show which sites have been locked and locked by whom. Suppose the transit requests have arrived at their destinations. At this point can any site enter the CS? Why? If not, how do the sites resolve the problem?

Answer:

13 nodes, 13 = 4(4-1) + 1, thus K = 4
R1 = { 1, 2, 3, 4 }
R2 = { 2, 5, 8, 11 }
R3 = { 3, 6, 8, 13 }
R4 = { 4, 6, 10, 11 }
R5 = { 1, 5, 6, 7 }
R6 = { 2, 6, 9, 12 }
R7 = { 2, 7, 10, 13 }
R8 = { 1, 8, 9, 10 }
R9 = { 3, 7, 9, 11 }
R10 = { 3, 5, 10, 12 }
R11 = { 1, 11, 12, 13 }
R12 = { 4, 7, 8, 12 }
R13 = { 4, 5, 9, 13 }



No site can enter the CS because it need to release the deadlock first.

To resolve deadlock, #4 send inquire message to #12 -> #6. But #6 have lower priority than #1 which has R from #4. So #4 should release itself from #12 by letting #12 reply a yield message. After #4 get yield message, it will be available for #1 -> #1 can enter CS -> #1 release locks for #2,3,4. Then #6 can enter CS -> #12 enter CS.

8. In a distribution system, there are 20 servers. Suppose the utilization of a server is 60%. Calculate the probability that the system has at least one task waiting and at least one server lying idle. Show your steps clearly.

Answer:

p: Utilization, R = 1 – p: probability processor idle

P: probability at least one task waiting and one server idle

$$P = \sum_{i=1}^{N} \binom{N}{i} Q_i H_{N-i}$$

where Q is probability that i servers idle and H_{N-i} is probability that set of (N-i) servers are not idle and at least one has a task waiting
$Q = R^i$, $H_{N-i} = (1-R)^{N-i} - [(1-R)R]^{N-i}$
Substituting and simplifying,
$P = 1 - (1-R)^N (1-R^N) - R^N(2-R)^N$

R = 1 – 0.6 = 0.4

N = 20

$P = 1 - (0.6)^{20} (1 - 0.4^{20}) - 0.4^{20}\, 1.6^{20} = 0.99983$