

- • Answers only -

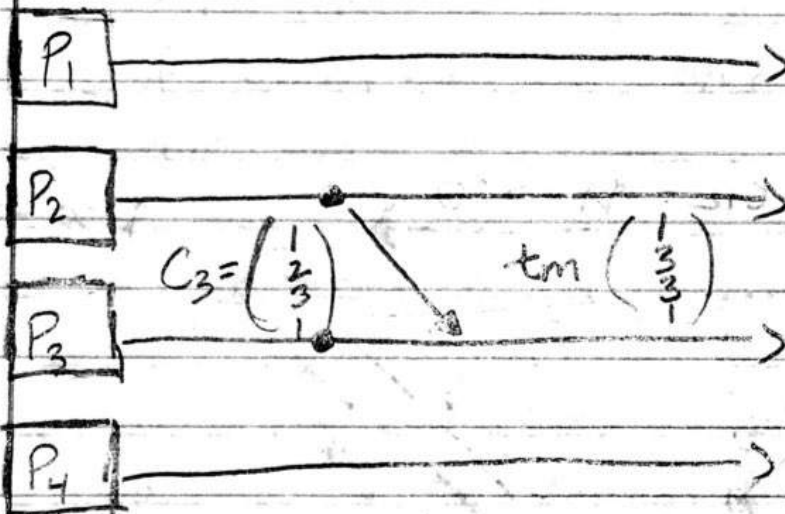
CSE 461 - HW 4 - Yousef Jarraf

3/11/19

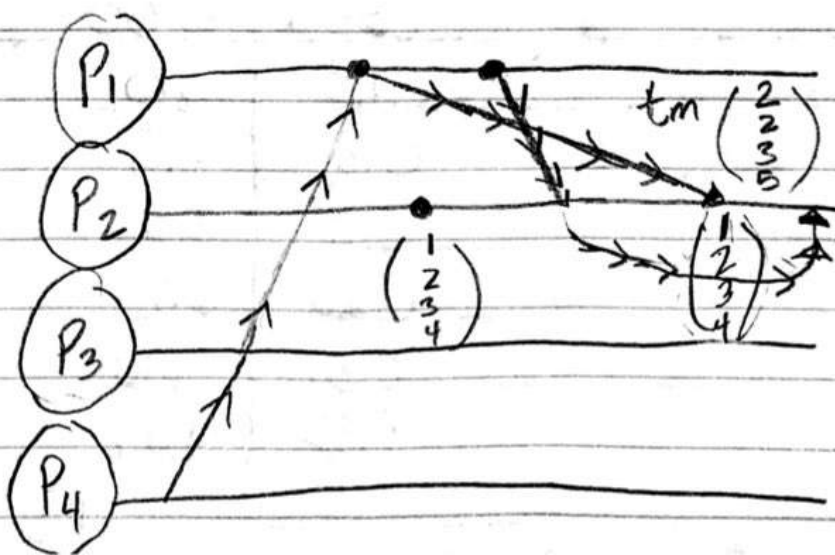
(cse 461 only do 6 questions, any more is extra points!)

Dr. Tong Yu

#1A) P_3 should deliver the message immediately
 $C_3[2] = tm[2] - 1$ and $C_3[k] \geq tm[k]$
 where $(k = 1, 3, 4)$



#1B) We know that $C_2[1] = tm[2] - 1$, but $C_2[4] < tm[4]$. P_2 should buffer message until it receives the previous message from P_1 .

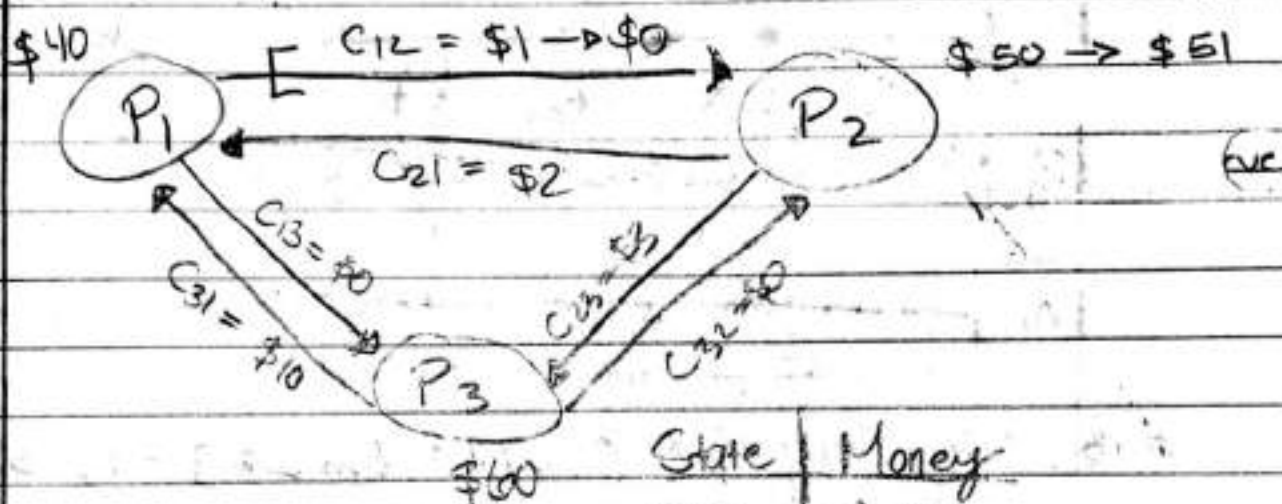


#2 C is inconsistent cut because:

$$t_c > \begin{pmatrix} C_1 [1] \\ C_2 [2] \\ C_3 [3] \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 3 \end{pmatrix}$$

and since $C[i] \neq T_c[i]$

#3



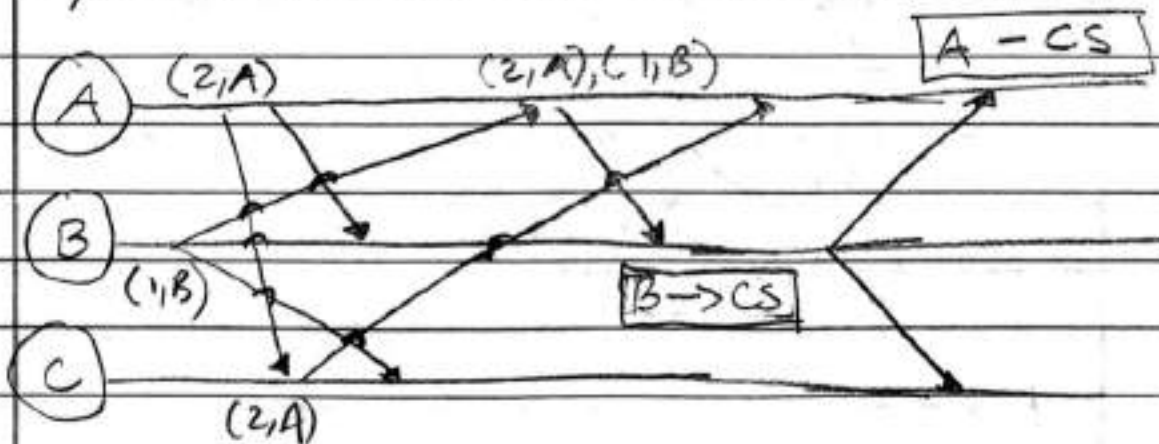
State	Money
LS1	\$40
LS2	\$51
LS3	\$60
C12	\$0
C13	\$0
C21	\$2
C23	\$3
C31	\$10
C32	\$0

#9

• Condition 1 is needed to guarantee the mutual exclusion b/c all the queues that each process needs to be in synchronization. Since each of the processes own queue might not be updated the same as the other. 2 processes might think that they are ^{at} the TOP of queue, when they are not.

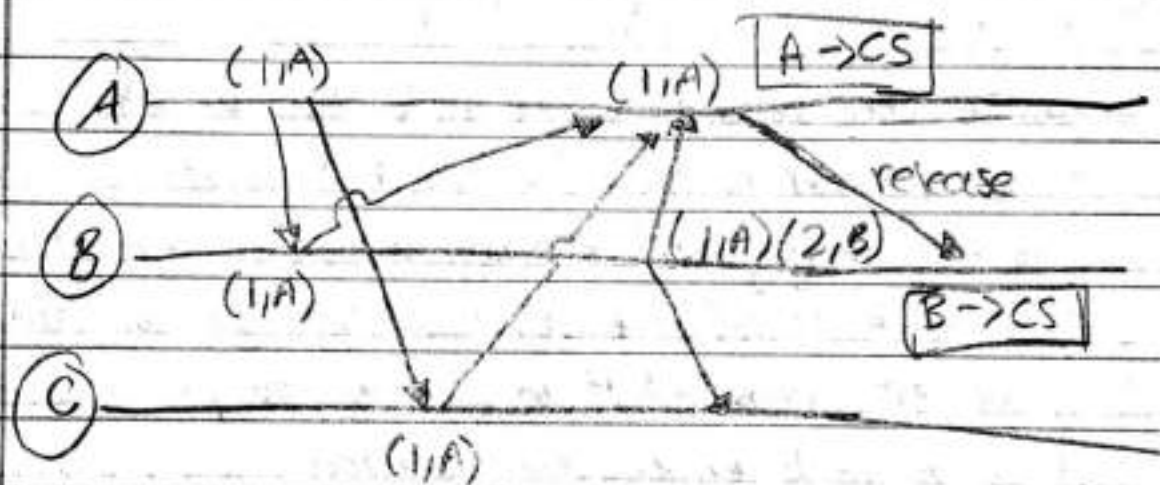
→ This condition (1) serves as a substitution of shared memory, since they do not have shared memory.

• If condition 2 is removed, the algorithm still works under a certain case. A "release message" serves as a reply. A release message from the process who entered the Critical Section can signal another process who's next in queue, to prepare another process to enter the Critical Section.



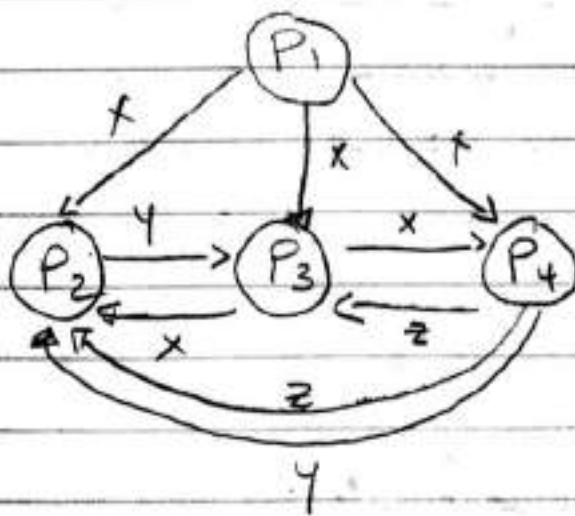
#5:

- It is not necessary. If it is executing the Critical section, then it already has visited the C.S.; then it doesn't need to be @ the top of the queue
- When we exit the critical section, process P_i removes its request from the head of its request - queue and sends a timestamp RELEASE to every other process.



#6 According to the Lamport-Shostack-Pease algorithm the agreement cannot be reached, if the number of faulty processors is $\geq 1/3$ of the total number of processors. A solution, would only be reached (Byzantine Agreement). Among 4 processors if \leq only if there is less than $1/3$ faulty processes and the maximum of faulty process should be one.

- P_2 's majority = $\frac{2}{3} x, x, z, z = x$, but P_2 is a traitor so it will retreat.
- P_3 majority = $\frac{2}{3} x, y, z, z = \text{retreat}$
- P_4 majority = $\frac{2}{3} x, x, y, z = x$, but P_4 is a traitor so it will retreat

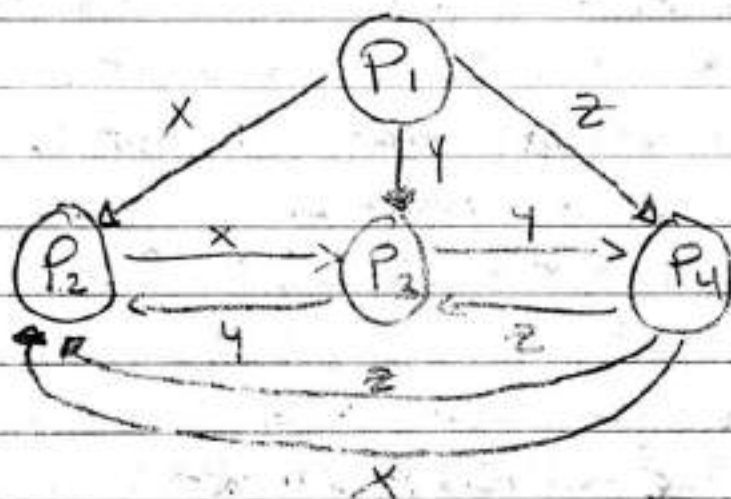


cont on back →

P_2 's majority = $\{x, y, z\} = \emptyset$

P_3 's majority = $\{y, x, z\} = \emptyset$

P_4 's majority = $\{z, x, y\} = \emptyset$



#7 13 nodes, $13 = 4(4-1) + 1$, therefore $K = 4$

$R_1 = \{1, 2, 3, 4\}$ No site can enter the C.S.

$R_2 = \{2, 5, 8, 11\}$ b/c it needs to release the

$R_3 = \{3, 6, 8, 13\}$ dead lock first.

$R_4 = \{4, 6, 10, 11\}$

$R_5 = \{1, 5, 6, 7\}$ → To resolve the deadlock,

$R_6 = \{2, 6, 9, 12\}$ #4 sends inquire message to

$R_7 = \{2, 7, 10, 13\}$ #12 → #6. But #6 has a

$R_8 = \{1, 8, 9, 10\}$ lower priority than #1 which has

$R_9 = \{3, 7, 9, 11\}$ a "K" from #4. So #4 should

$R_{10} = \{3, 5, 10, 12\}$ release itself from #12 by letting #12

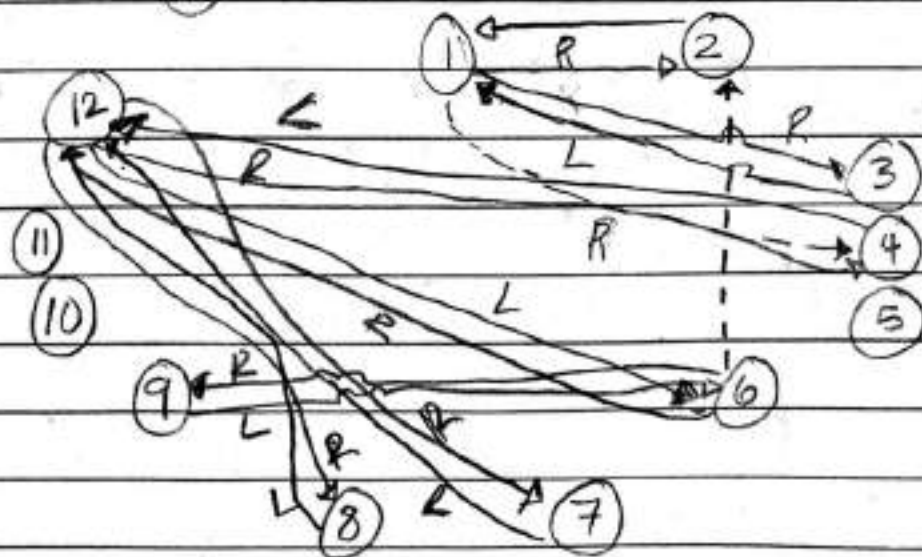
$R_{11} = \{1, 11, 12, 13\}$ reply a yield message. After

$R_{12} = \{4, 7, 8, 12\}$ #4 gets a yield message so it will

$R_{13} = \{4, 5, 9, 13\}$ be available #1 → #1 can enter

CS, #1 can enter CS. → #1 release locks

(13) #2, 3, 4. Then #6 can enter CS → #12 enter CS



#8

p : Utilization, $1-p$: 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

$$Q = p^i, H_{N-i} = (1-p)^{N-i} - [(1-p)p]^{N-i}$$

• Substituting and simplifying,

$$P = 1 - (1-p)^N (1-p^N) - p^N (2-p)^N$$

$$p = 1 - 0.6 = 0.4$$

$$N = 20$$

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

$$= 0.99983$$