Solutions for quiz 1 M2024.CS3401

Question 1

In words, any attempt to convey the equivalent of any of the following has been accepted.

- Messages destined for the same destination are not reordered.
- For all pairs of send events *send(m_{ij})* and *send(m_{kj})* destined for the same process, where *i* may be equal to *k*, if *send(m_{ij})* causally precedes *send(m_{kj})*, then *recv(m_{ij})* also causally precedes *recv(m_{kj})*.

In symbols,

for any two messages m_{ij} and m_{kj},
if $send(m_{ij}) \rightarrow send(m_{kj})$ then $recv(m_{ij}) \rightarrow recv(m_{kj})$.

Question 2

Conditions for Consistent Global States

- A global state GS is a consistent global state iff it satisfies the following two conditions :
- C1 states the law of conservation of messages.
  - Every message that is recorded as sent in the local state of some process is either captured in the state of the channel or is captured in the local state of the receiver.
  - $send(m_{ij}) \in LS_i \Rightarrow m_{ij} \in SC_{ij} \oplus rec(m_{ij}) \in LS_j$. ($\oplus$ is the Exclusive-OR operator)
- C2 states that for every effect, its cause must be present.
  - If a message is not recorded as sent in the local state of a process P_i, then the message cannot be included in the state of the channel C_{ij} or be captured as received by P_j.
  - $send(m_{ij}) \notin LS_i \Rightarrow m_{ij} \notin SC_{ij} \wedge rec(m_{ij}) \notin LS_j$.

Q3) Lamport's algorithm for mutual exclusion

a) The two conditions to be satisfied for a process P_i to enter the critical section are:

1. P_i has received messages with timestamp larger than (ts_i, i) from other processes.
2. P_i's request is at the top of its request queue , $request_queue_i$, ie. it has the smallest timestamp of all requests received.

b) Lamport's algorithm for mutual exclusion assumes the following:

- Pairs of bidirectional channels between processes
- All channels satisfy FIFO message delivery

Assuming channels do not have FIFO delivery, Lamport's algorithm cannot achive mutual exclusion.

If two processes P_i and P_j are taken, then assuming both want to enter the CS, they place their requests on their request queue.

Let both processes have their own requests at the top of their request queues.

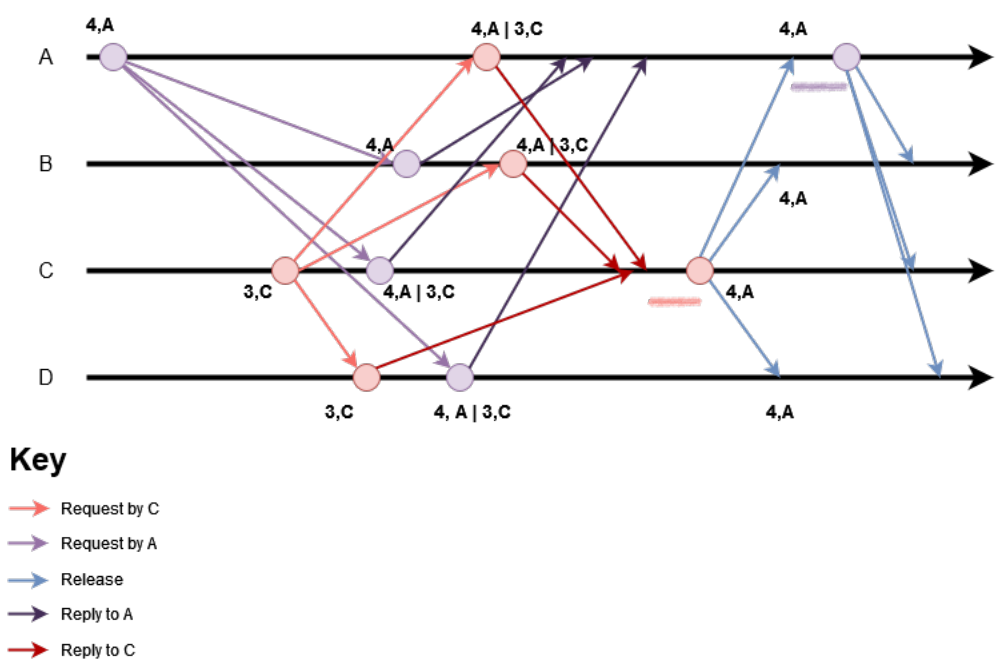
Let P_i's request have a lesser timestamp than P_j's.

Then at a time t, it is possible for the reply from P_i to beat the request from P_j, causing P_j to start executing.

Once the reply from P_j is received, P_i also begins executing as it has the lowest timestamp.

$\therefore$ Mutual exclusion does not happen

c)



Q4)

a) MPI_Send/MPI_Recv: Synchronous and blocking.

b) MPI_Isend/MPI_Irecv: Asynchronous and non-blocking.

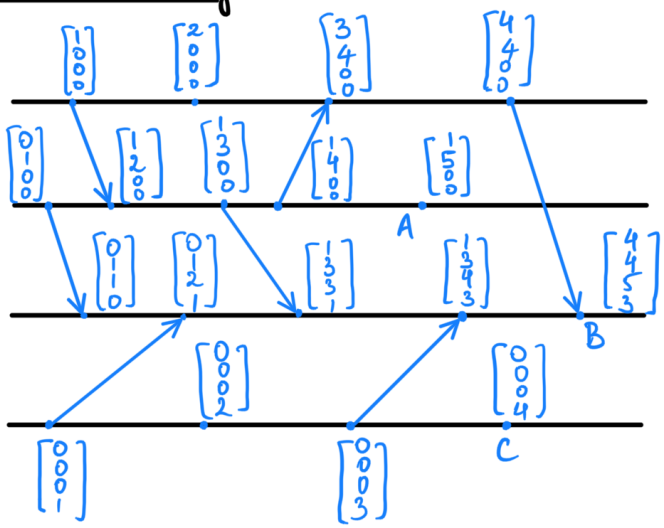
c) TCP Send/Receive: Synchronous and blocking (by default).\

d) UDP Send/Receive: Asynchronous and blocking (by default, but can be non-blocking).

Q5) See Below page

Quiz 1 Answer Key

Q5.



a. A - $\begin{bmatrix}1\\5\\0\\0\end{bmatrix}$, B - $\begin{bmatrix}4\\4\\5\\3\end{bmatrix}$, C - $\begin{bmatrix}0\\0\\0\\4\end{bmatrix}$

b. A||B , A||C , B||C (any other example)

c. Yes. In causal ordering, if send(m_1) → send(m_2) then recv(m_1) → recv(m_2) at all common destinations (that is recv(m_1) & recv(m_2) occur at the same process.)

In the given process time diagram, this cond^n is not violated and hence all messages are delivered in causal order.

d. No, each process can follow its own value of increment, d. This is because each process P_i only ever increments the ith index of its clock. For all other indices, it just maintains the max. value it has seen yet. Since all processes independently maintain their own local clock, they can use their own value of d, so long as it remains constant throughout the program runtime.