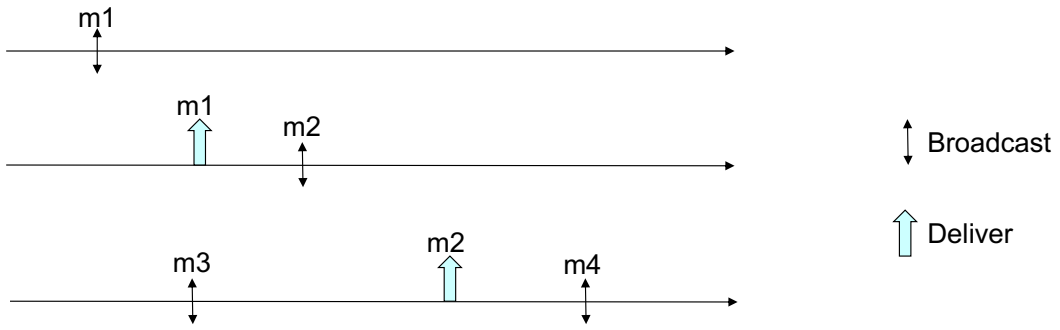
Dependable Distributed Systems
Master of Science in Engineering in Computer Science

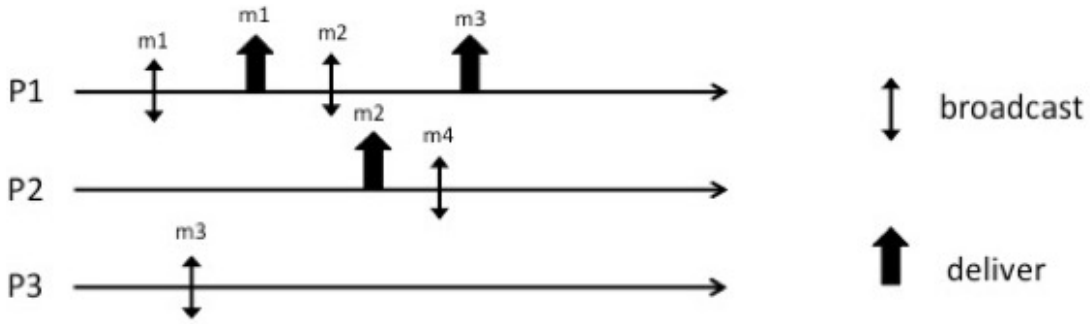
AA 2022/2023

Lecture 13 – Exercises
October 26th, 2022

Ex 1: Given the partial execution in Figure, provide all the delivery sequences such that both total order and causal order are satisfied



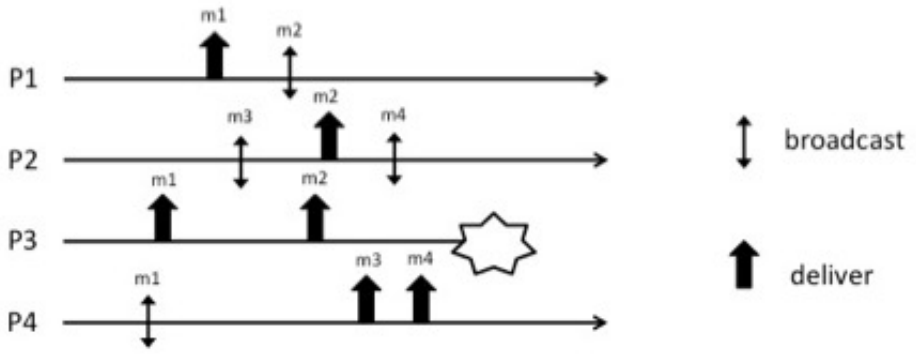
Ex 2: Let us consider the following partial execution



Answer the following points:

1. Provide all the possible sequences satisfying Causal Order
2. Complete the execution to have a run satisfying FIFO order but not causal order

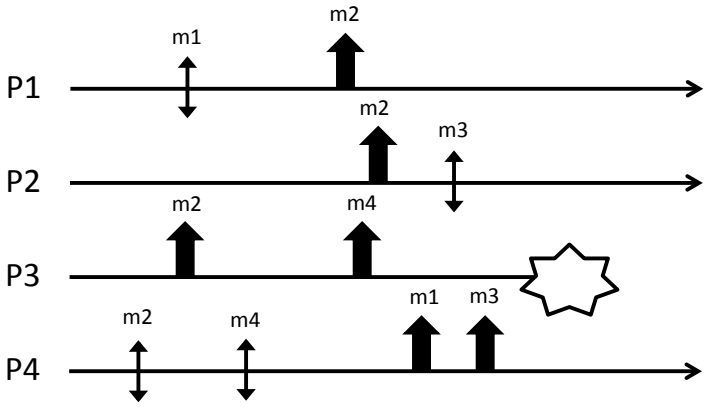
Ex 3: Let us consider the following partial execution



Answer the following points:

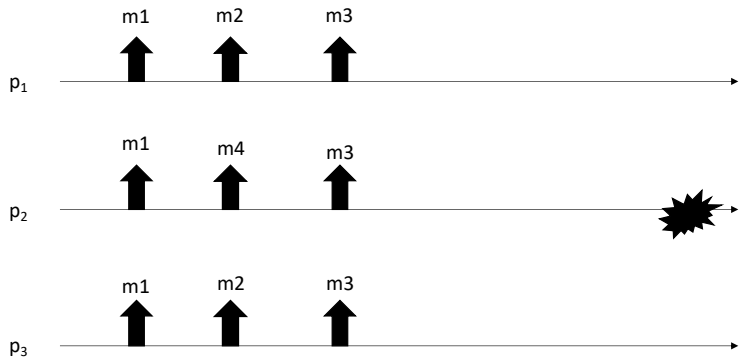
1. Provide the list al all the possible delivery sequences that satisfy both Total Order and Causal Order
2. Complete the history (by adding the missing delivery events) to satisfy Total Order but not Causal Order
3. Complete the history (by adding the missing delivery events) to satisfy FIFO Order but not Causal Order nor Total Order

Ex 4: Consider the message pattern shown in the Figure below and answer to the following questions:



1. Complete the execution in order to have a run satisfying Reliable Broadcast but not Uniform Reliable Broadcast.
2. Provide all the delivery sequences satisfying causal order and total order.
3. Provide all the delivery sequences violating causal order and satisfying TO(UA, WNUTO) but not satisfying TO(UA, SUTO)

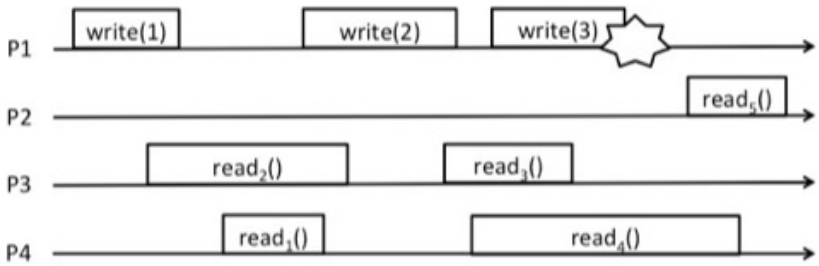
Ex 5: Consider the partial execution in the following figure



Given the run depicted in the figure state the truthfulness of the following sentences:			
a	The strongest agreement property satisfied is UA	T	F
b	The NUA agreement property is violated	T	F
c	The strongest ordering property satisfied is SUTO	T	F
d	The WUTO ordering property is satisfied	T	F
e	The SNUTO ordering property is violated	T	F
f	Let us assume we can add only one more delivery to p1 and p3, it is not possible to get a run satisfying TO(NUA, SUTO)	T	F
g	If p2 is not going to deliver m4 then the strongest specification satisfied by the resulting execution is TO(UA, SUTO)	T	F
h	Let us assume we can add only one more delivery to p1 and p3, it is possible to get a run satisfying TO(UA, WNUTO) but not satisfying TO(UA, WUTO)	T	F
i	If p2 is not faulty, the NUA agreement property is satisfied	T	F
l	If p2 is not faulty, the SUTO ordering property is satisfied	T	F

For each point, provide a justification for your answer

Ex 6: Consider the execution depicted in the following figure and answer the questions



1. Define ALL the values that can be returned by read operations (Rx) assuming the run refers to a regular register.
2. Define ALL the values that can be returned by read operations (Rx) assuming the run refers to an atomic register.

Ex 7: Consider a distributed systems composed by a set of *n* processes p_1, p_2,… p_n. Processes have a unique identifier and are structured as a binary tree topology. Messages are exchanged between processes over the edges of the tree which act like perfect point-to-point links. Each process p_i has stored the identifiers of its neighbors

into the local variables FATHER, R_CHILD e L_CHILD representing respectively the father of p_i, the right child and the left child (if they exists).
Assuming that processes are not going to fail, write the pseudo-code of an algorithm satisfying the following specification:

Events:

- **Request:** ⟨ *tob*, Broadcast | m ⟩: Broadcasts a message m to all processes.
- **Indication:** ⟨ *tob*, Deliver | p, m ⟩: Delivers a message m broadcast by process p.

Properties:

- *Validity:* If a correct process *p* broadcasts a message *m*, then *p* eventually delivers *m*.
- *No duplication:* No message is delivered more than once.
- *No creation:* If a process delivers a message *m* with sender *s*, then *m* was previously broadcast by process *s*.
- *Agreement:* If a message *m* is delivered by some correct process, then *m* is eventually delivered by every correct process.
- *Total order:* Let m_1 and m_2 be any two messages and suppose *p* and *q* are any two correct processes that deliver m_1 and m_2. If *p* delivers m_1 before m_2, then *q* delivers m_1 before m_2.