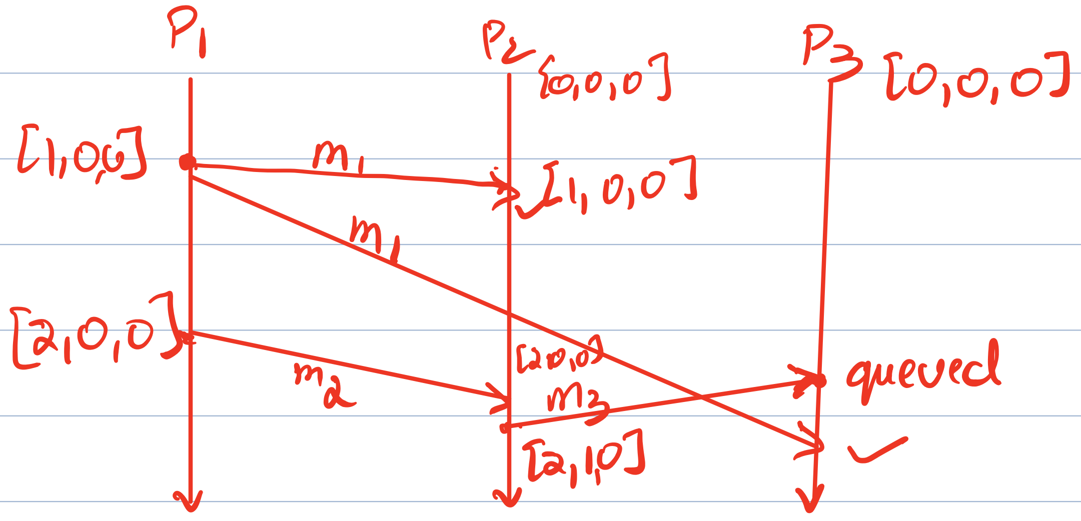
Agenda:

- exam review
- total order vs determinism
- consistency models
    - RYW, FIFO, causal, strong
- dealing with failure in (strongly consistent) replication protocols
- intro to consensus
- FLP result.

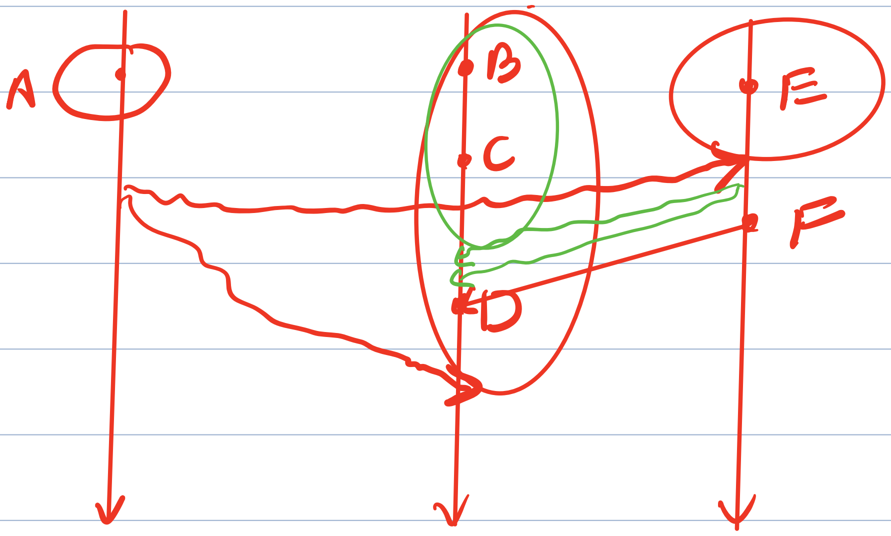
Exam Review:



Can't use causal broadcast algorithm because
- m_2 & m_3 are not broadcast messages!

m_3 is waiting on m_1 & m_2. Hence, m_3 cannot be delivered & it'll never be possible!

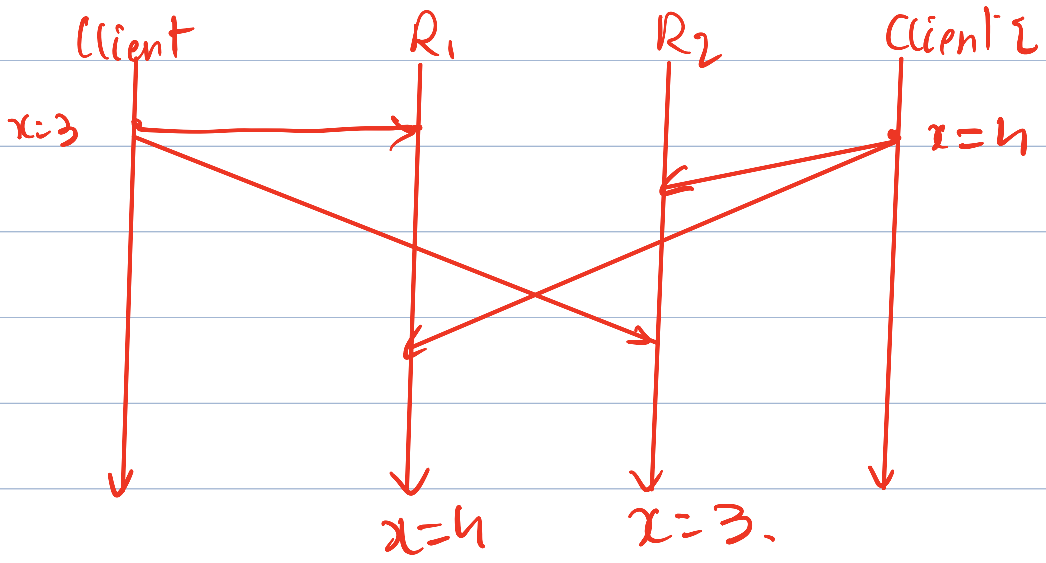
Snapshot question:



Invalid snapshot: F not in snapshot when it precedes D

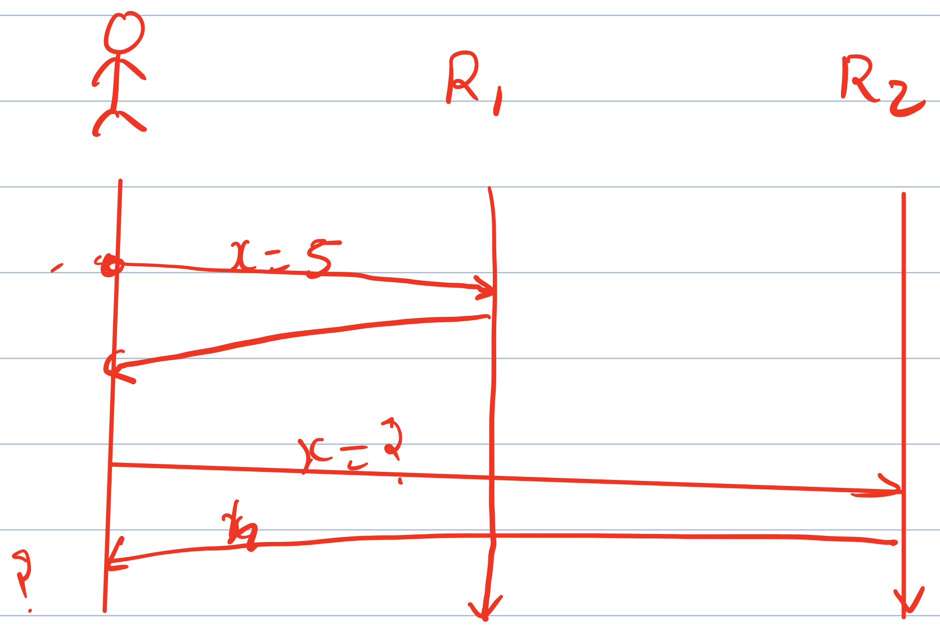
But this would never happen, because marker from E would reach before D. Hence, actual snapshot would be in green.

Total order vs determinism:

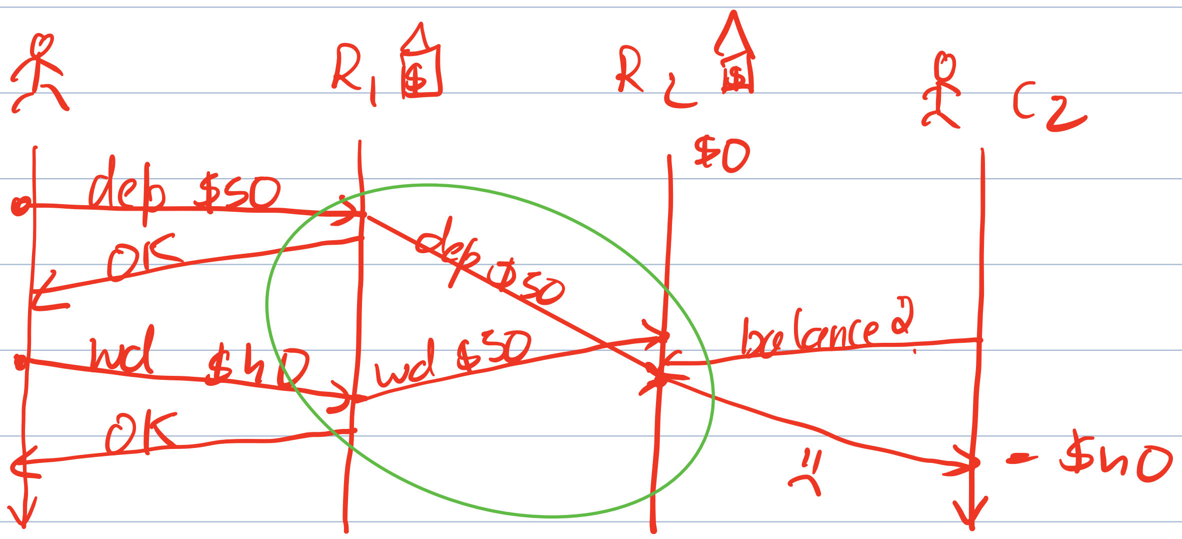


Even with one replica, we don't necessarily have determinism

Ways in which replicas can disagree.

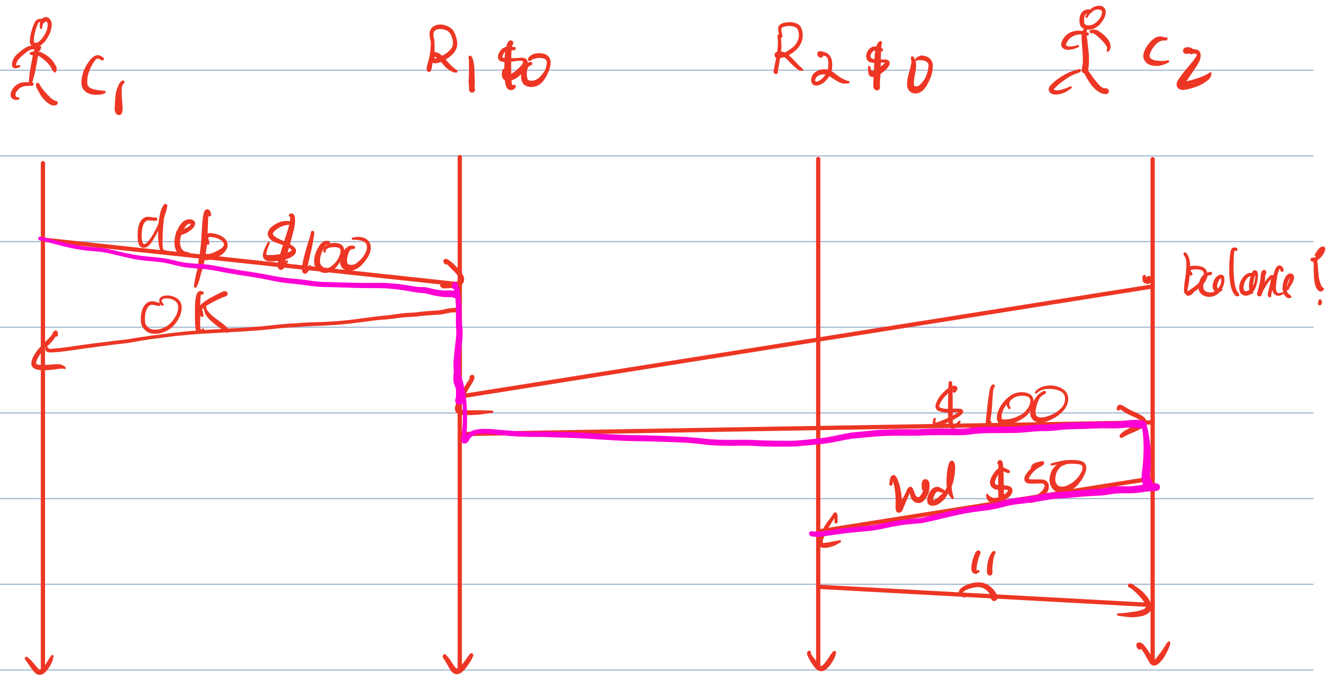


Property being violated: Read Your Writes (RYW)



Violation of FIFO consistency.

FIFO consistency: writes done by a process are seen by all processes in the order they were issued



Why did C2 think they can make a withdrawl?

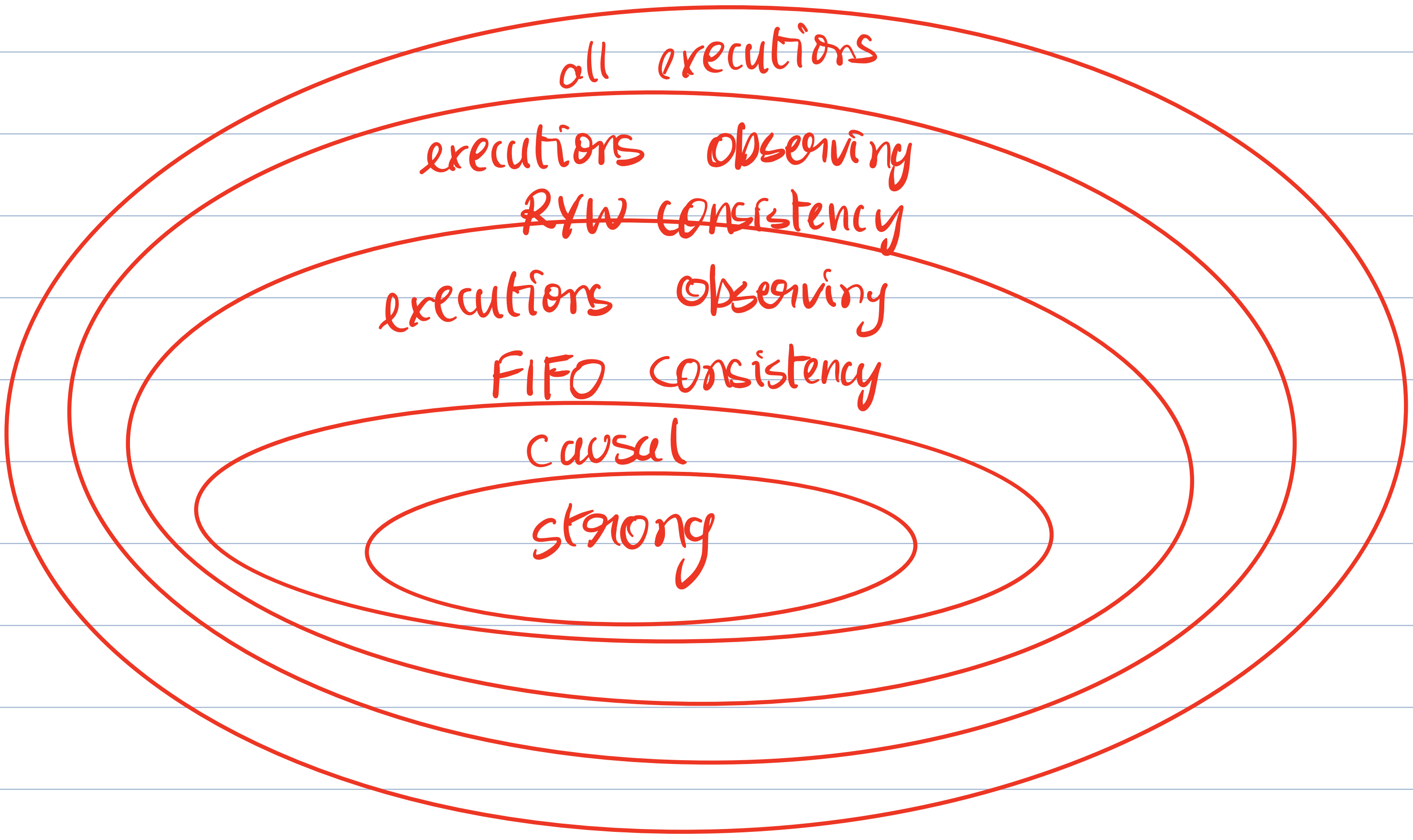
Pink shows chain of events using happens before relation.

R2 sees withdrawl BUT NOT the causal history of withdrawl.

Violation of causal consistency.

Causal Consistency: Writes that are related by happens-before must be seen in the same (causal) order by all processes

[illegible]


all executions
executions observing ~~RYW consistency~~
executions observing FIFO consistency
causal
strong

Why not always take strong ?

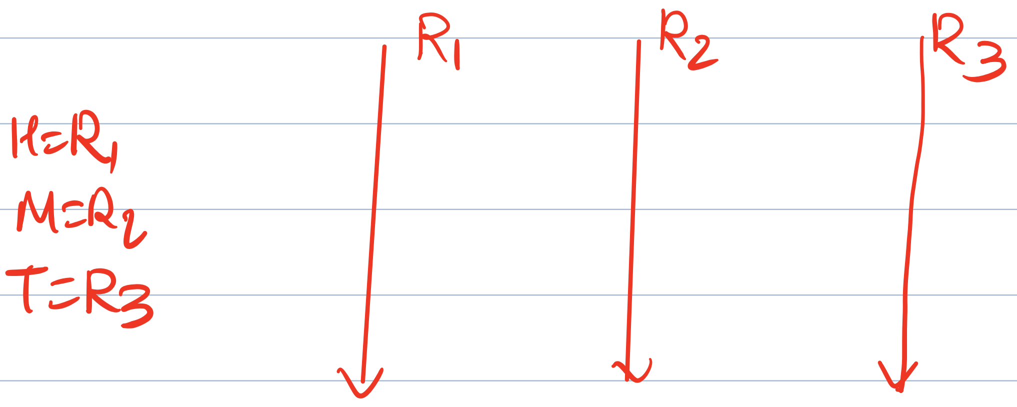
- Replicas could crash
- More effort (more msgs passed around)

Generally, causal consistency is a good compromise between right & fast

Model: set of assumptions.

Strongly consistent replication protocols
- Primary backup
- Chain replication

Need a coordinator process.



Fail stop fault model: crashes can occur & can be detected by the environment.

Perfect failure detection is IMPOSSIBLE in async distsys!

If Head (H) fail, what does coordinator need to do?

- Replace R_1 with R_2 (successor)

when T process fails?

- Replace R_3 with R_2 (predecessor)

So, R_2 is both Head & Tail, which is OK!

Read chain replication paper: Van Renesse & Schneider, 2004

Object storage on CRAQ, 2009

What if coordinator fails?

- Have a bunch of coordinators? Need to keep them consistent !!!

This will be solved with consensus

Consensus

When do you need consensus?
You have bunch of processes & ..

totally ordered broadcast
- you need to make sure they deliver the same messages in same order

group membership
- you need them all to know what other process exist & keep those lists updated

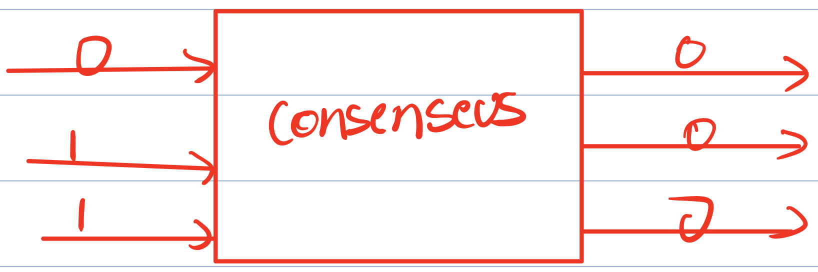
leader election
- you need one of them to play a particular special role, & everyone

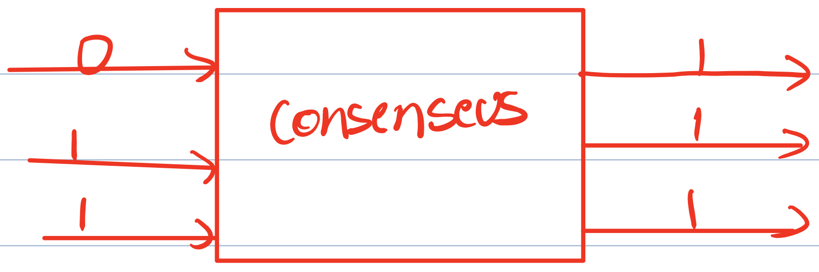
else needs to know about it

Distributed mutual exclusion
- you need them to take turns getting access to a shared resource

distributed transaction commit
- they're participating in a transaction & need to agree on a commit labord decision.

Process try to agree on one bit, 0 or 1





Have to agree on one value

Paxos!