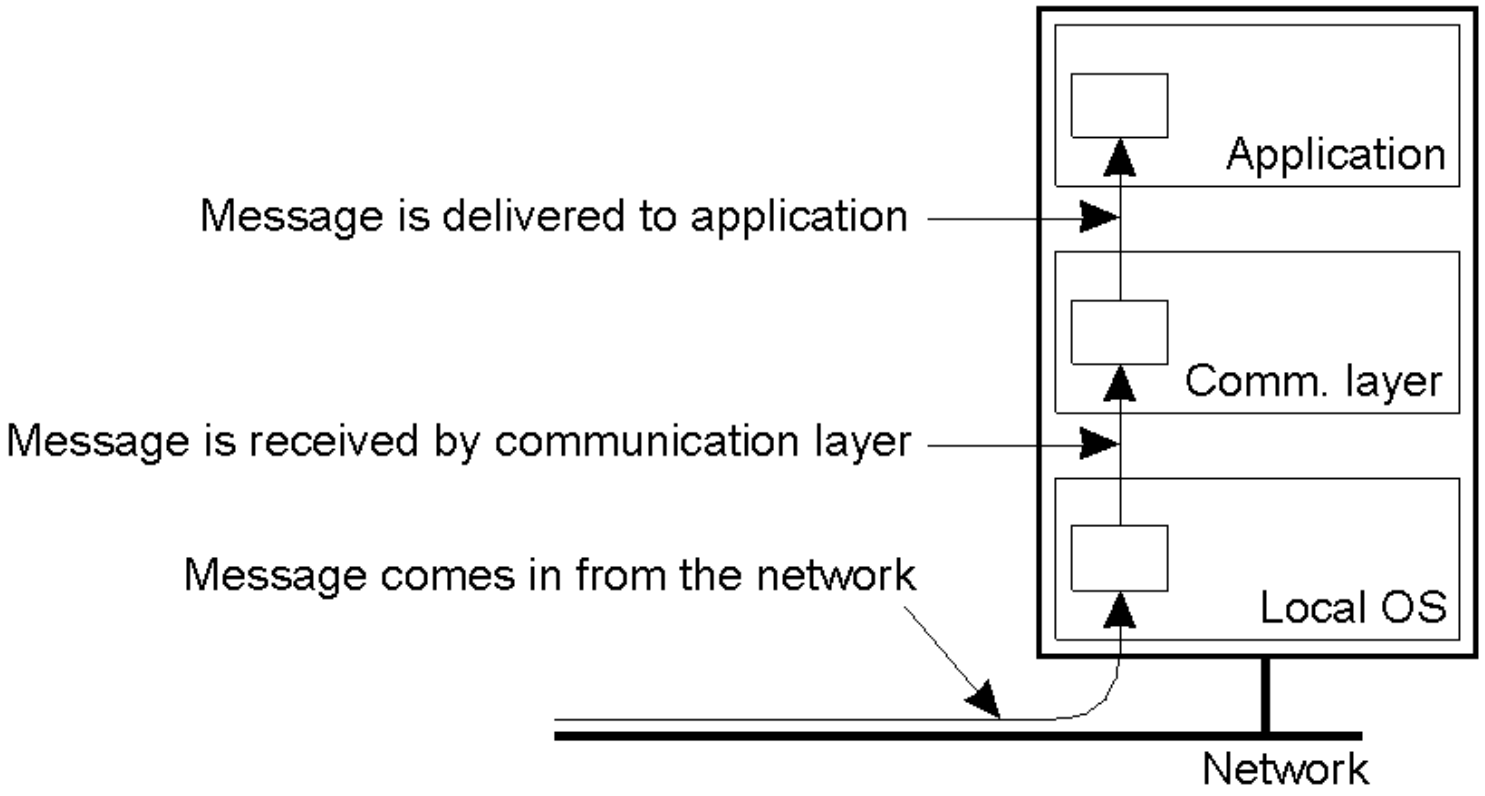
MYE017 Distributed Systems

Kostas Magoutis
magoutis@cse.uoi.gr
http://www.cse.uoi.gr/~magoutis

Message reception vs. delivery



The logical organization of a distributed system to distinguish between message receipt and message delivery

FIFO message ordering

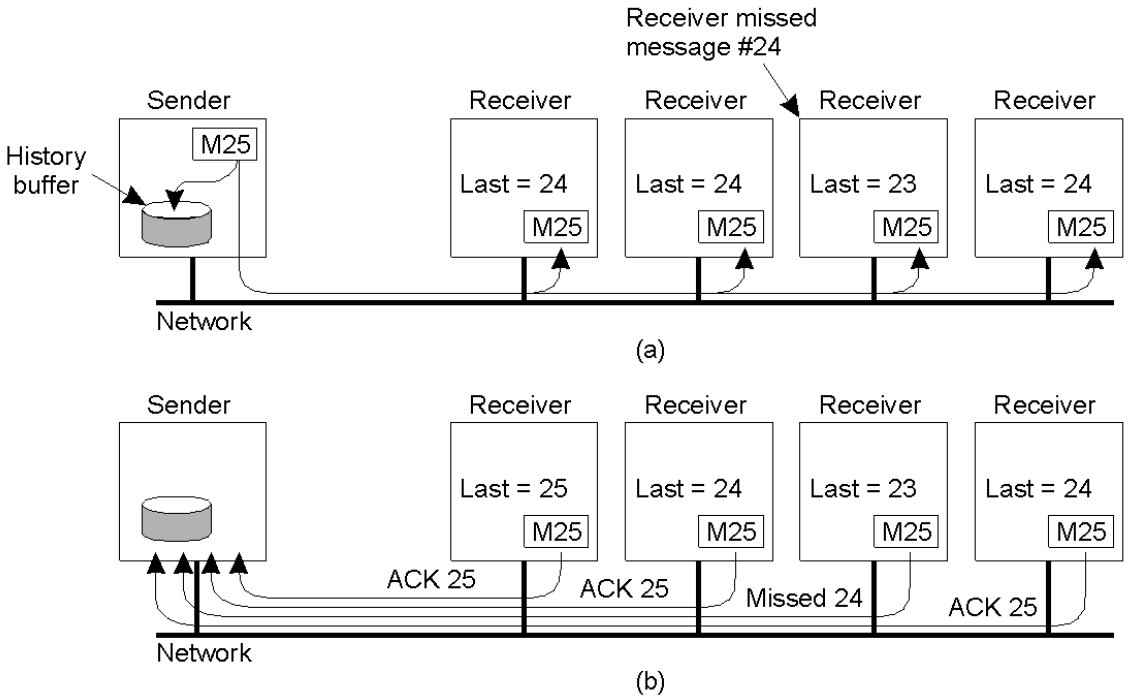
Process P1	Process P2	Process P3	Process P4
sends m1	delivers m1	delivers m3	sends m3
sends m2	delivers m3	delivers m1	sends m4
	delivers m2	delivers m2	
	delivers m4	delivers m4	

- Four processes in a group with two different senders
- A possible delivery order under FIFO multicasting

Versions of multicasting

Multicast	Basic Message Ordering	Total-ordered Delivery?
Reliable multicast	None	No
FIFO multicast	FIFO-ordered delivery	No
Causal multicast	Causal-ordered delivery	No
Atomic multicast	None	Yes
FIFO atomic multicast	FIFO-ordered delivery	Yes
Causal atomic multicast	Causal-ordered delivery	Yes

Reliable multicasting, basic schemes

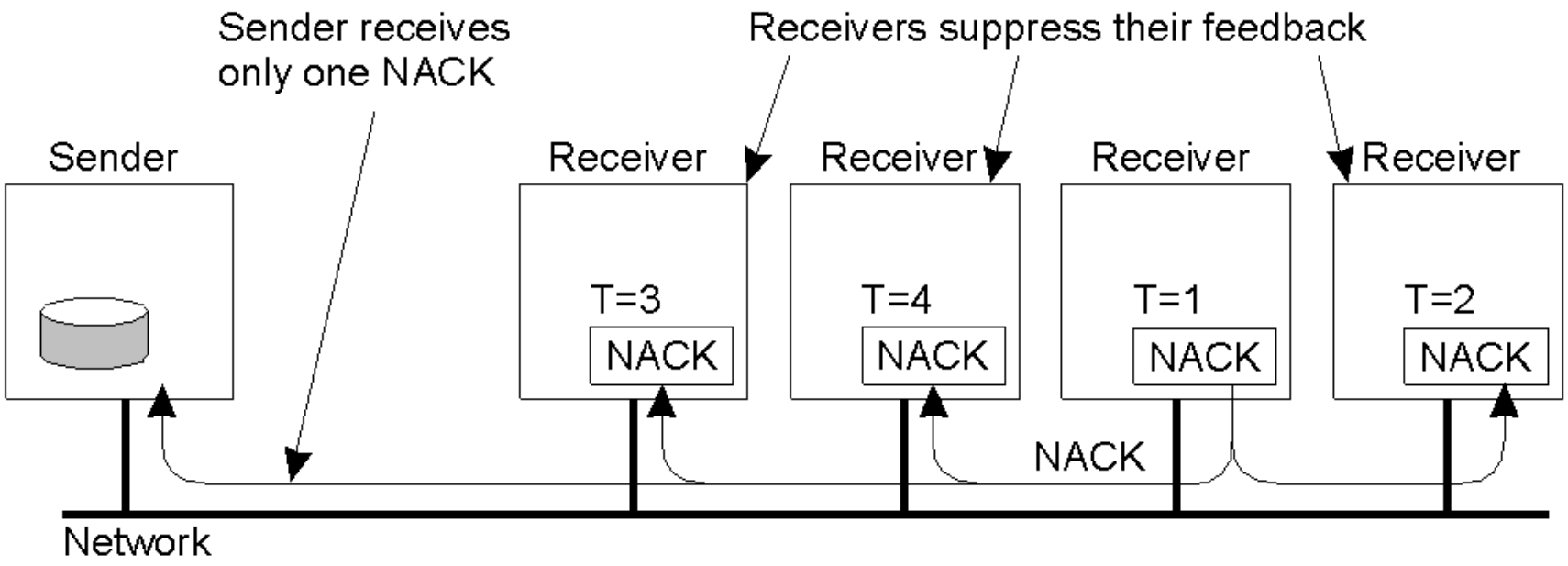


A simple solution to reliable multicasting when all receivers are known and assumed not to fail

a) Message transmission

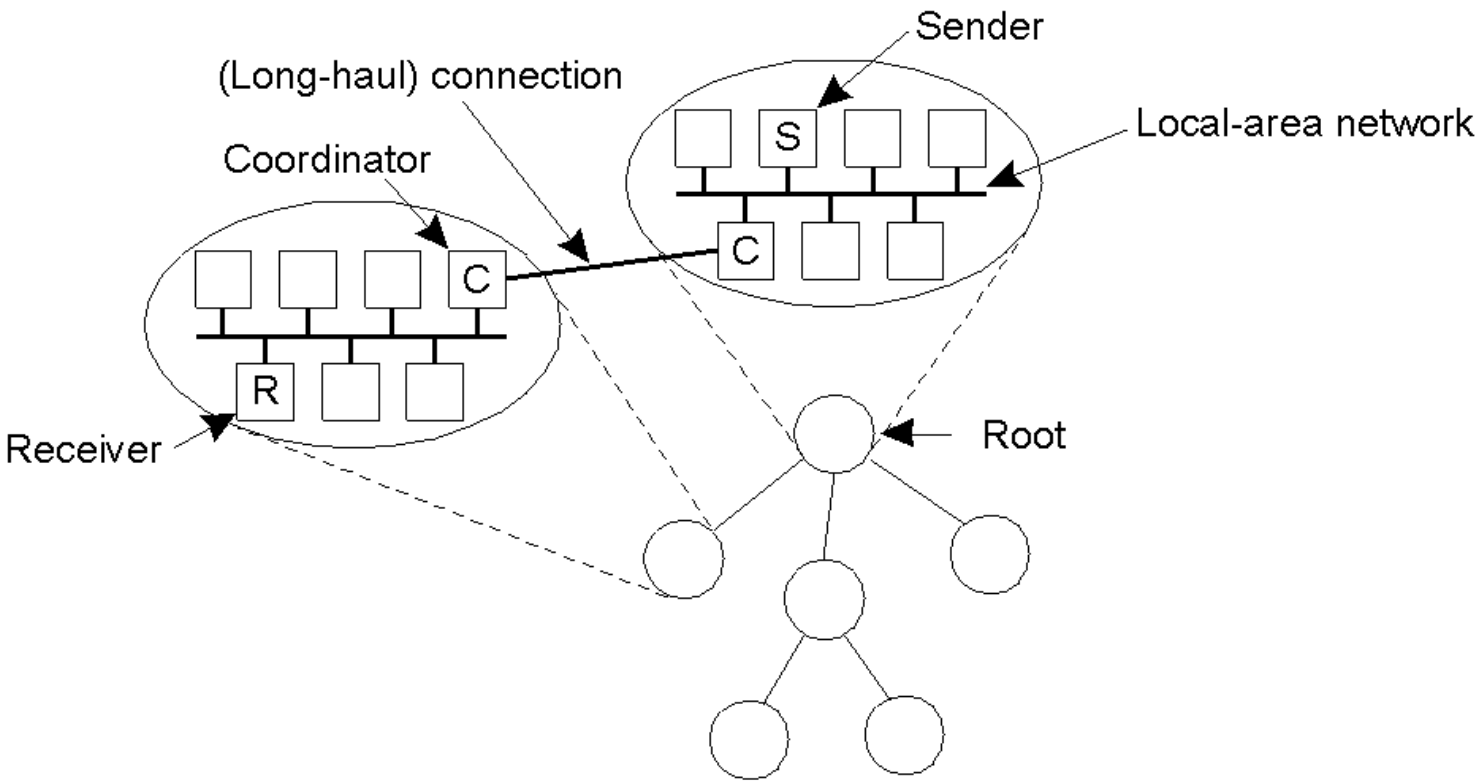
b) Reporting feedback

Nonhierarchical feedback control



Several receivers have scheduled a request for retransmission, but the first retransmission request leads to the suppression of others

Hierarchical feedback control

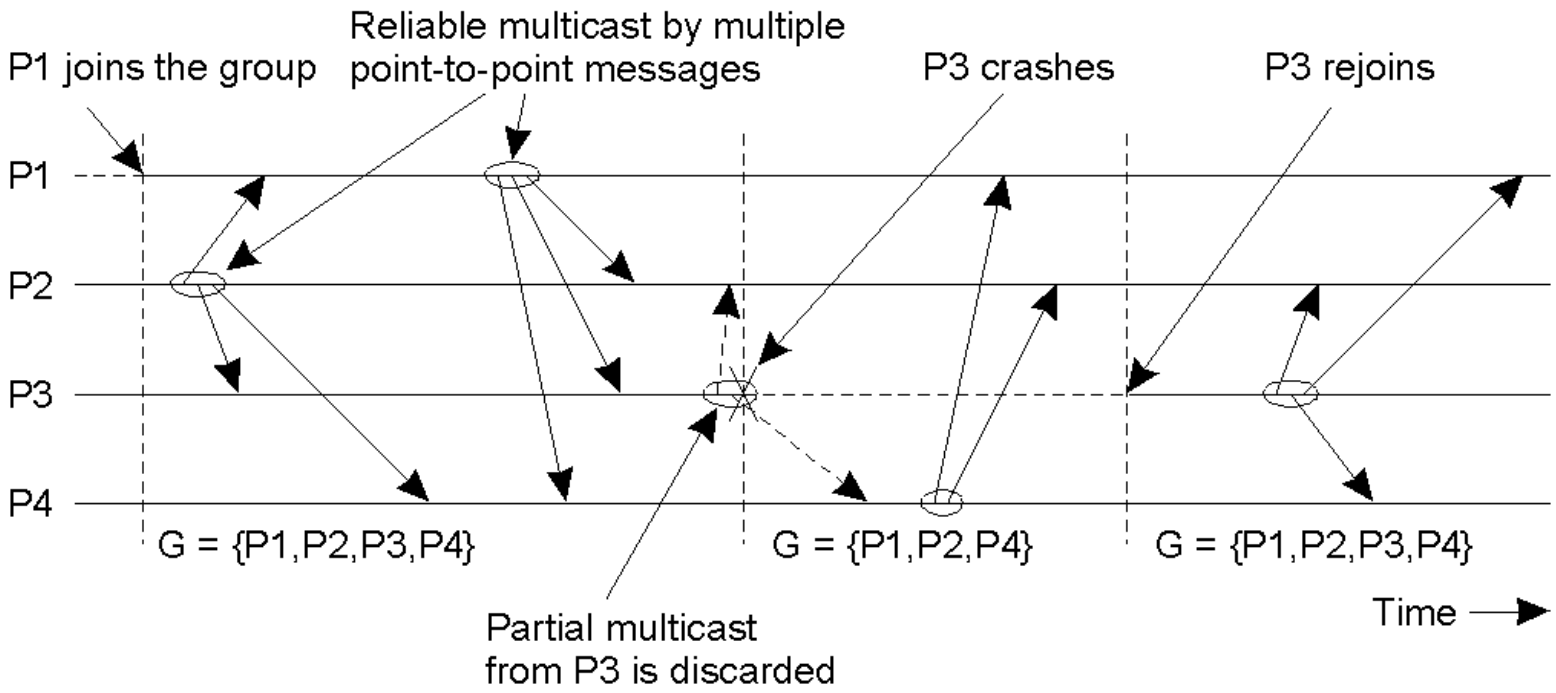


The essence of hierarchical reliable multicasting

a) Each local coordinator forwards the message to its children

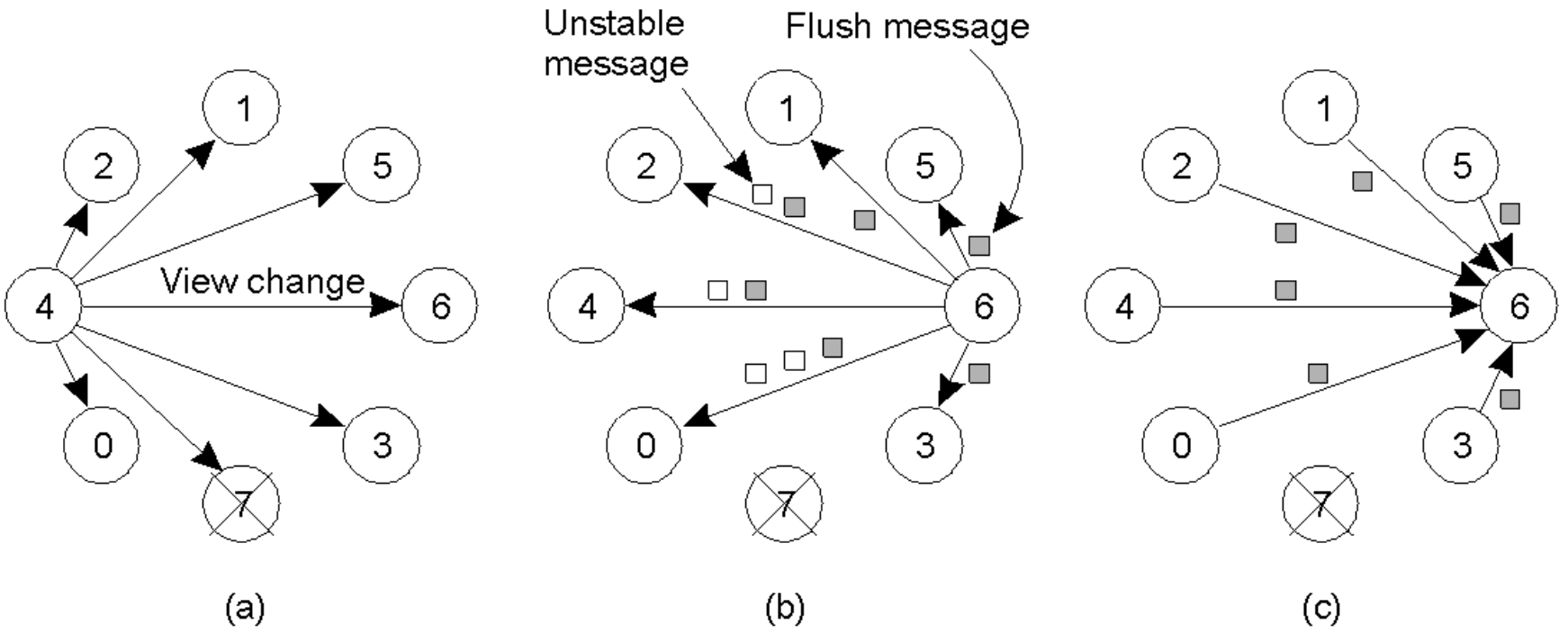
b) A local coordinator handles retransmission requests

Virtual Synchrony



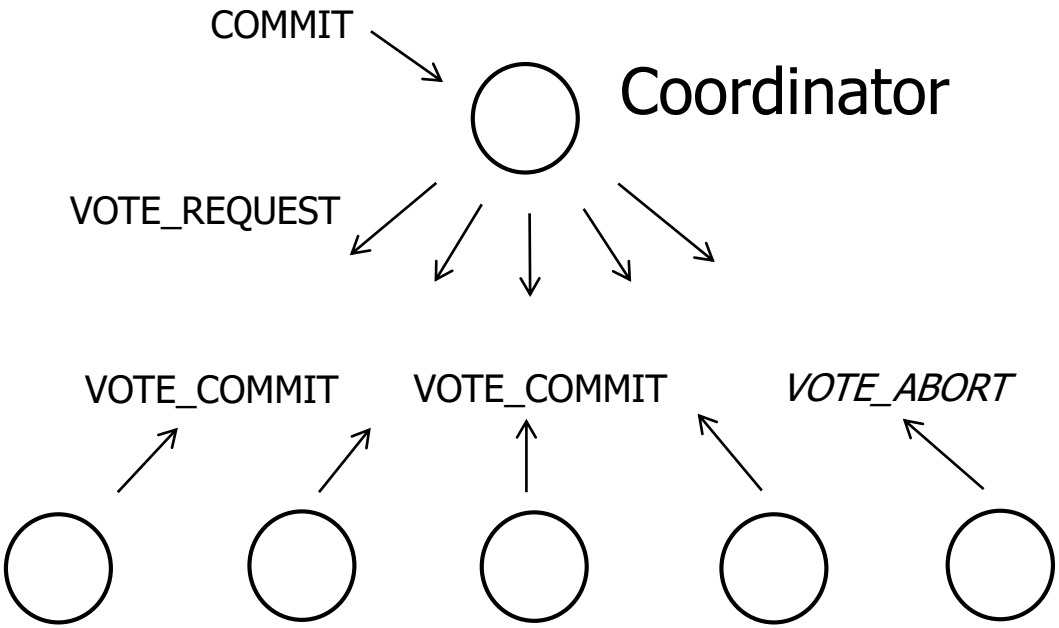
The principle of virtual synchronous multicast

Implementing Virtual Synchrony (2)



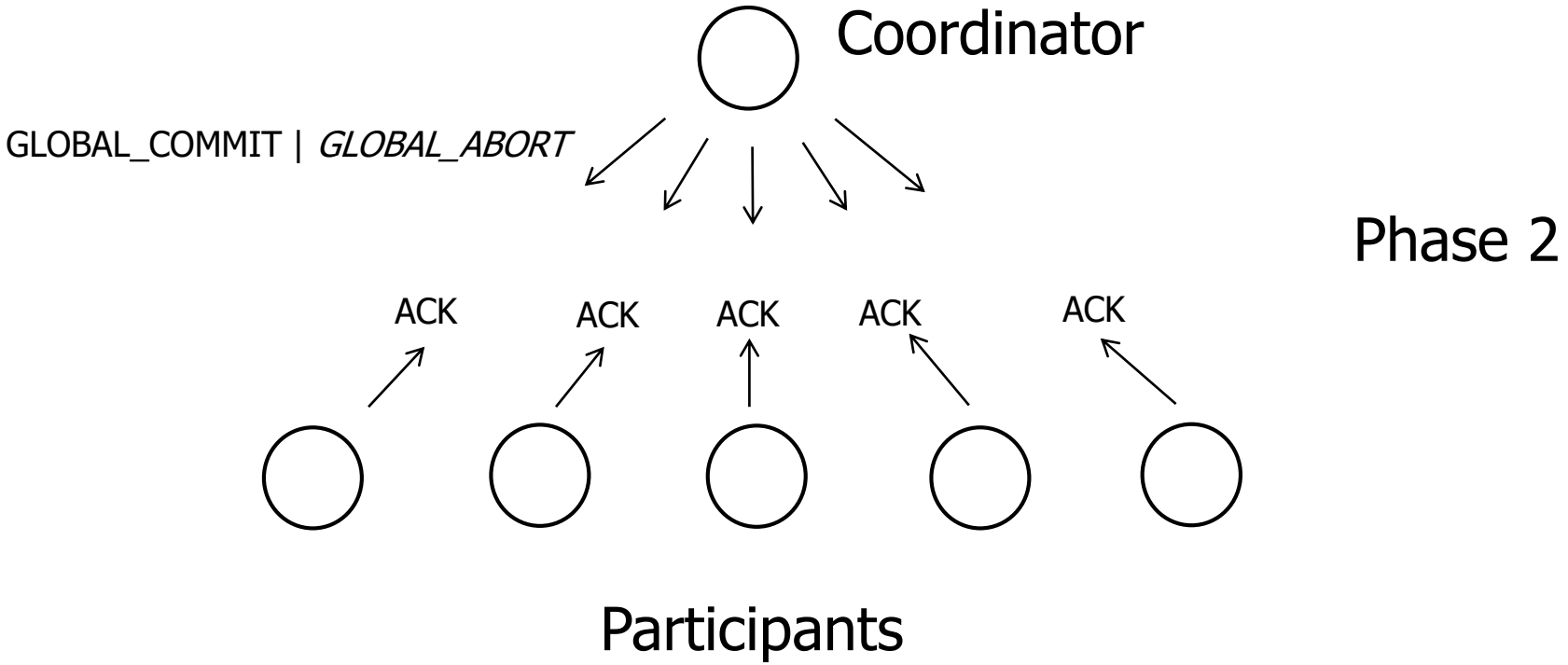
- Process 4 notices that process 7 has crashed, sends a view change
- Process 6 sends out all its unstable messages, followed by a flush message
- Process 6 installs the new view when it has received a flush message from everyone else

Two-phase commit

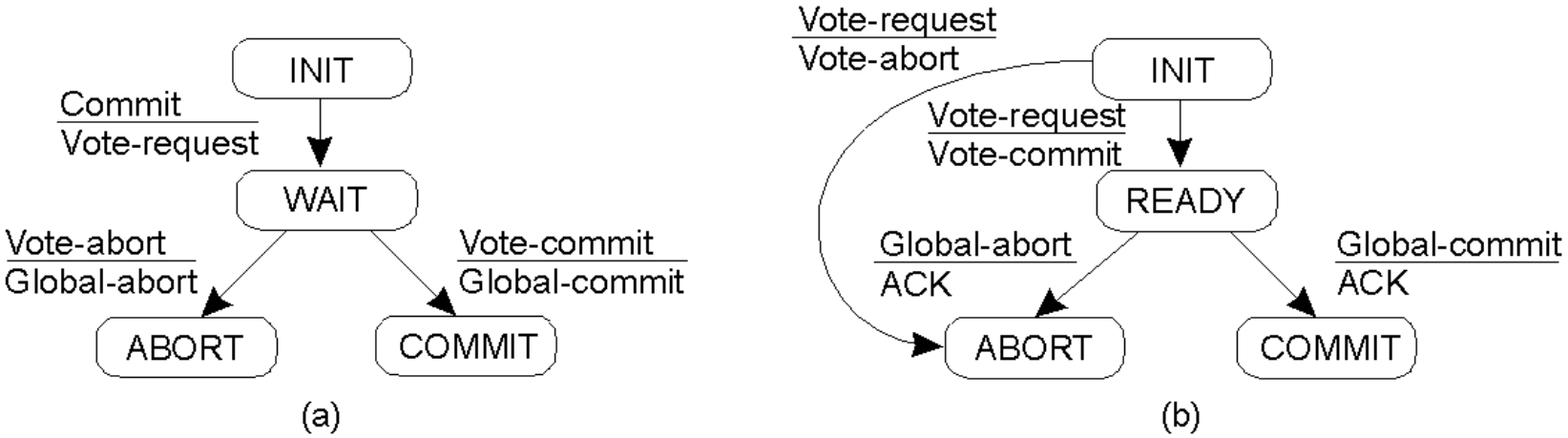


Phase 1

Two-phase commit

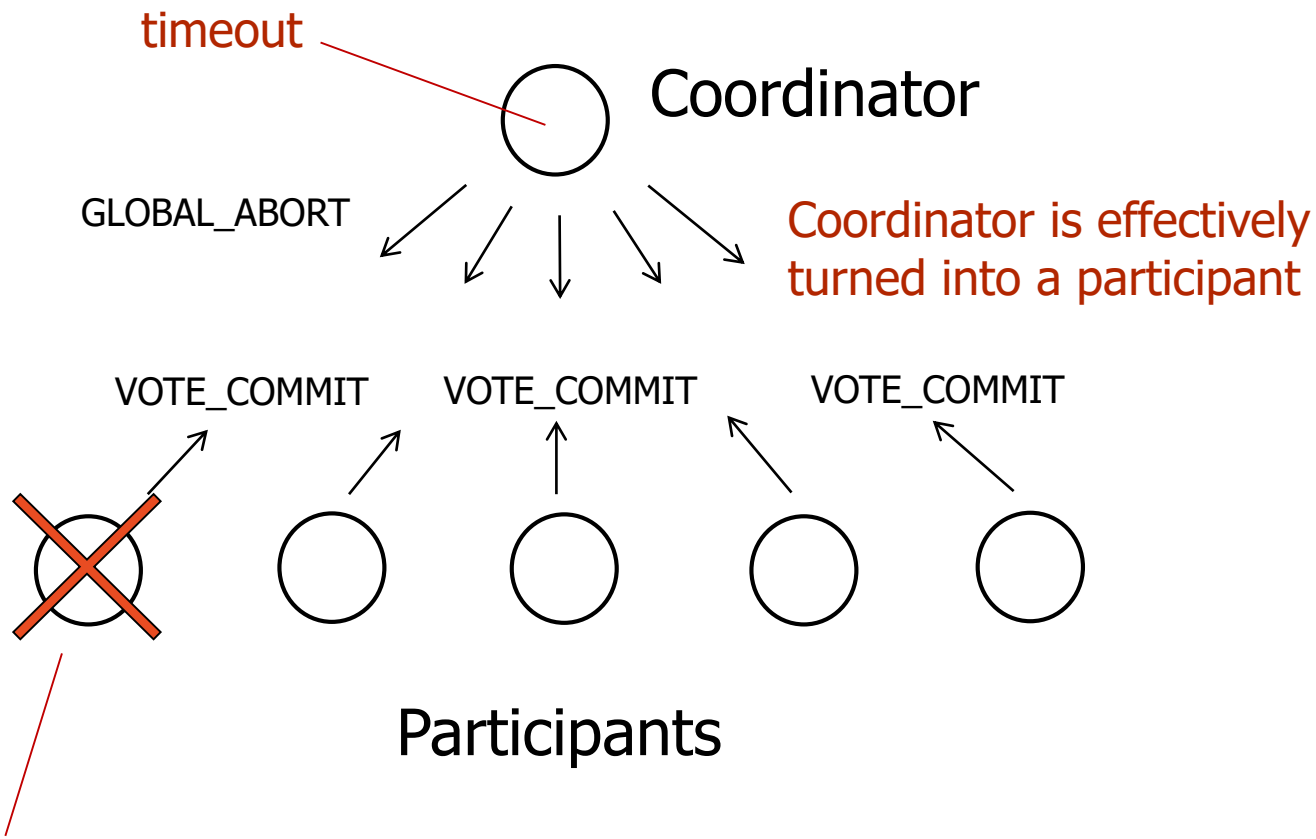


2PC - State machines



a) The finite state machine for the coordinator in 2PC
b) The finite state machine for a participant

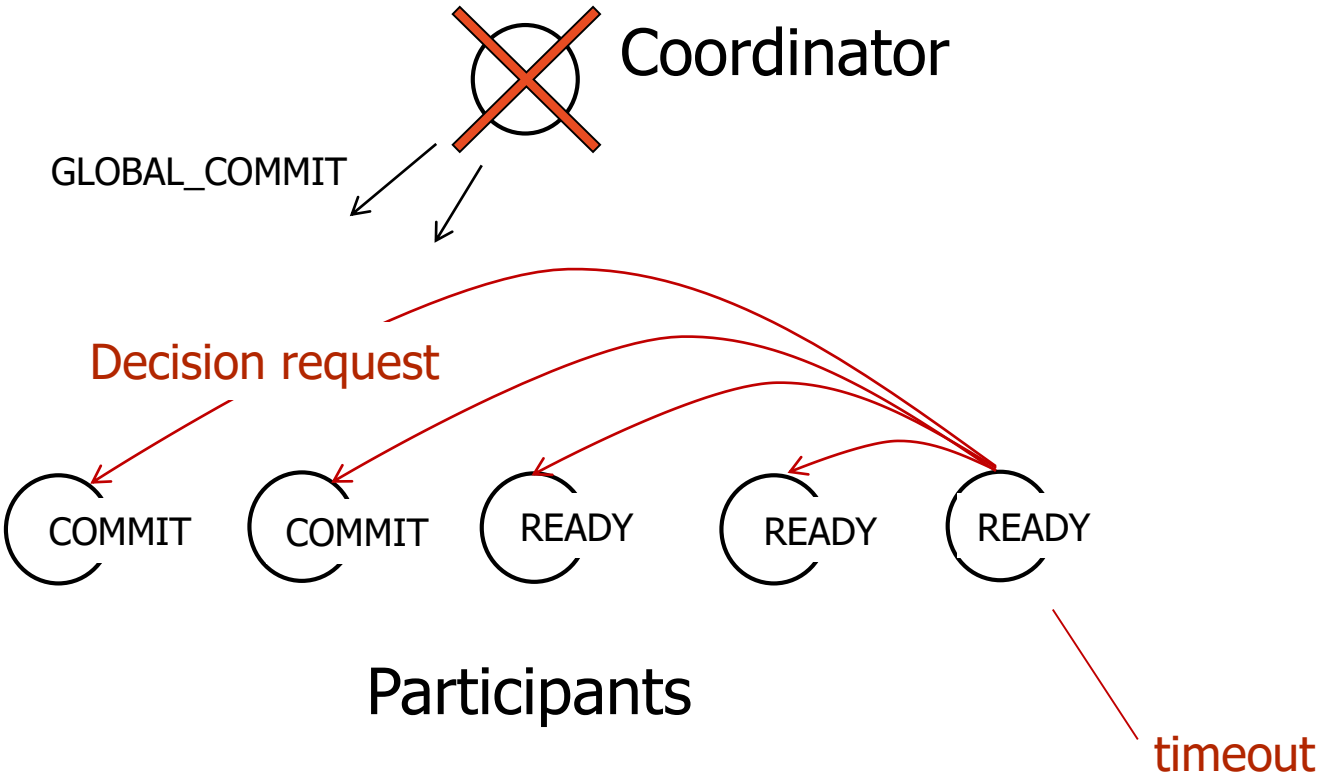
2PC – Operation under failures



A participant crashes, disconnects, or is too slow during a vote

2PC – Operation under failures

Coordinator crashes after sending a few (but not all) commit messages



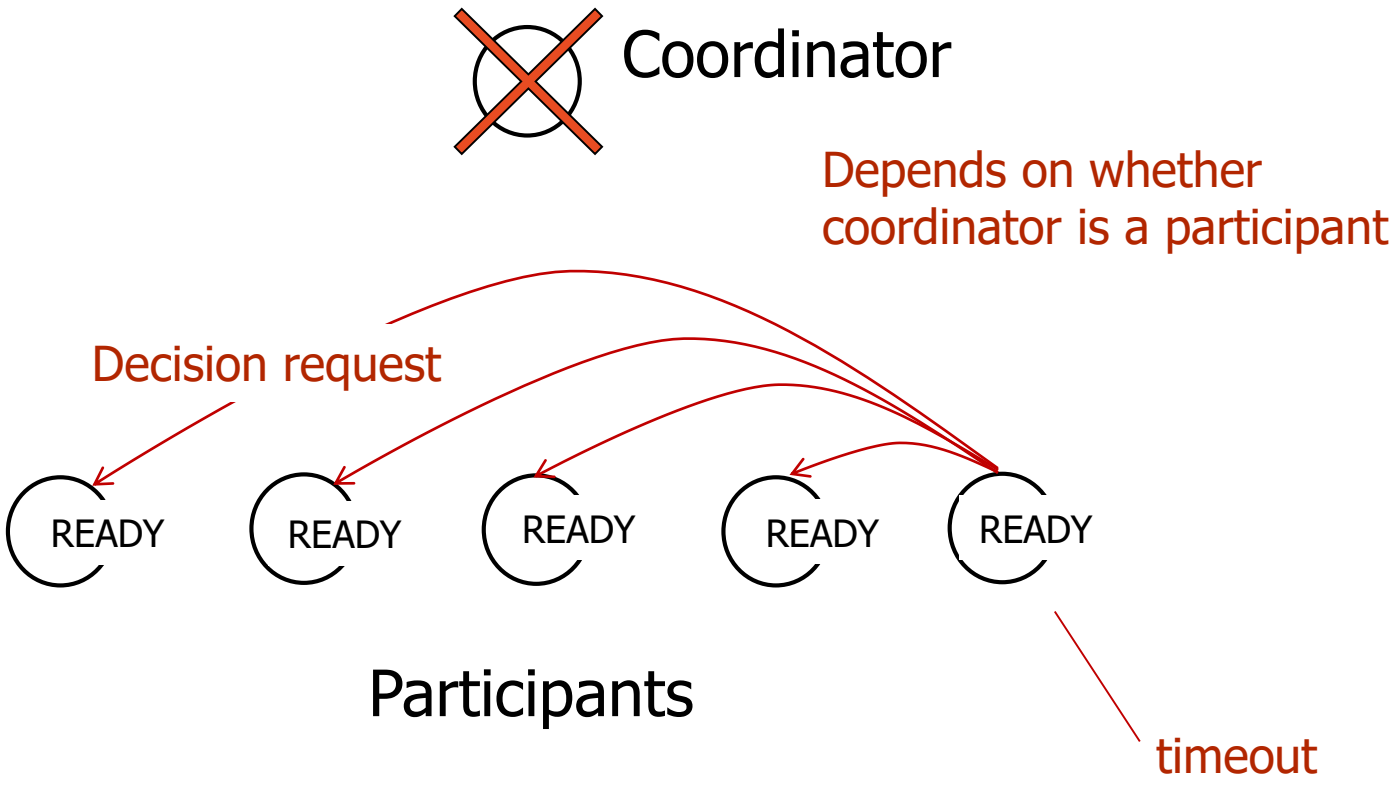
2PC – Coordinator crash

State of Q	Action by P
COMMIT	Make transition to COMMIT
ABORT	Make transition to ABORT
INIT	Make transition to ABORT
READY	Contact another participant

Actions taken by a participant *P* when residing in state *READY* and having contacted another participant *Q*.

2PC – What happens if all in READY state?

Coordinator crashes, disconnects, or is too slow



2PC – May block forever

Coordinator crashes, disconnects, or is too slow – but not a participant

Coordinator

Crashed participant may be the only one that knows decision

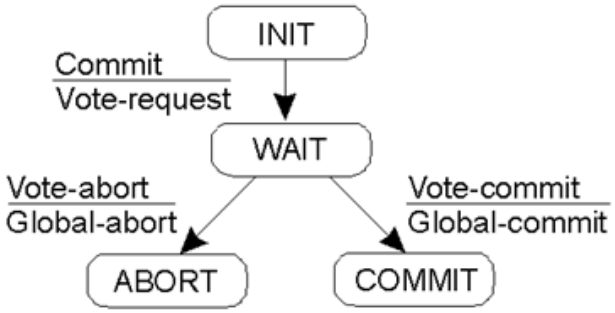
Participants

timeout

A participant crashes, disconnects, or is too slow

2PC – Actions by coordinator

Ensure that state is recoverable



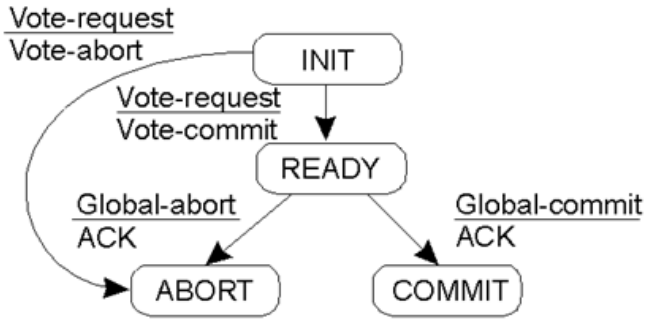
actions by coordinator:

```
write  START _2PC to local log;
multicast VOTE_REQUEST to all participants;
while not all votes have been collected {
    wait for any incoming vote;
    if timeout {
        write  GLOBAL_ABORT to local log;
        multicast  GLOBAL_ABORT to all participants;
        exit;
    }
    record vote;
}
if all participants sent VOTE_COMMIT and coordinator votes COMMIT{
    write GLOBAL_COMMIT to local log;
    multicast GLOBAL_COMMIT to all participants;
} else {
    write GLOBAL_ABORT  to local log;
    multicast GLOBAL_ABORT to all participants;
}
```

2PC – Actions by participant

actions by participant:

```
write INIT to local log;
wait for VOTE_REQUEST from coordinator;
if timeout {
    write VOTE_ABORT to local log;
    exit;
}
if participant votes COMMIT {
    write VOTE_COMMIT to local log;
    send VOTE_COMMIT to coordinator;
    wait for DECISION from coordinator;
    if timeout {
        multicast DECISION_REQUEST to other participants;
        wait until DECISION is received; /* remain blocked */
        write DECISION to local log;
    }
    if DECISION == GLOBAL_COMMIT
        write GLOBAL_COMMIT to local log;
    else if DECISION == GLOBAL_ABORT
        write GLOBAL_ABORT to local log;
} else {
    write VOTE_ABORT to local log;
    send  VOTE ABORT to coordinator;
}
```

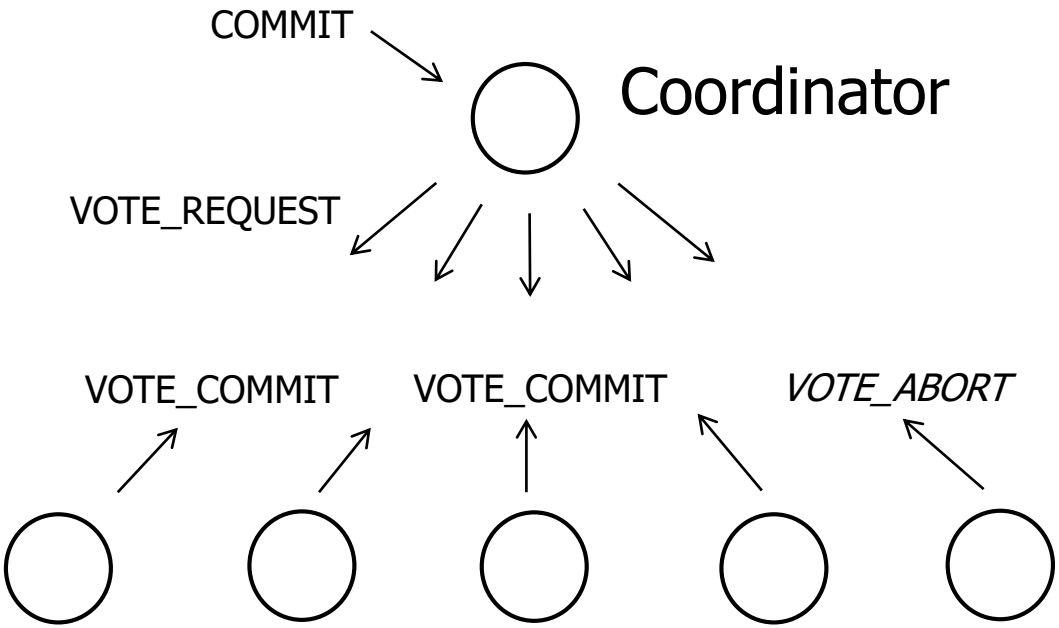


2PC – Handling incoming decision requests

actions for handling decision requests: /* executed by separate thread */

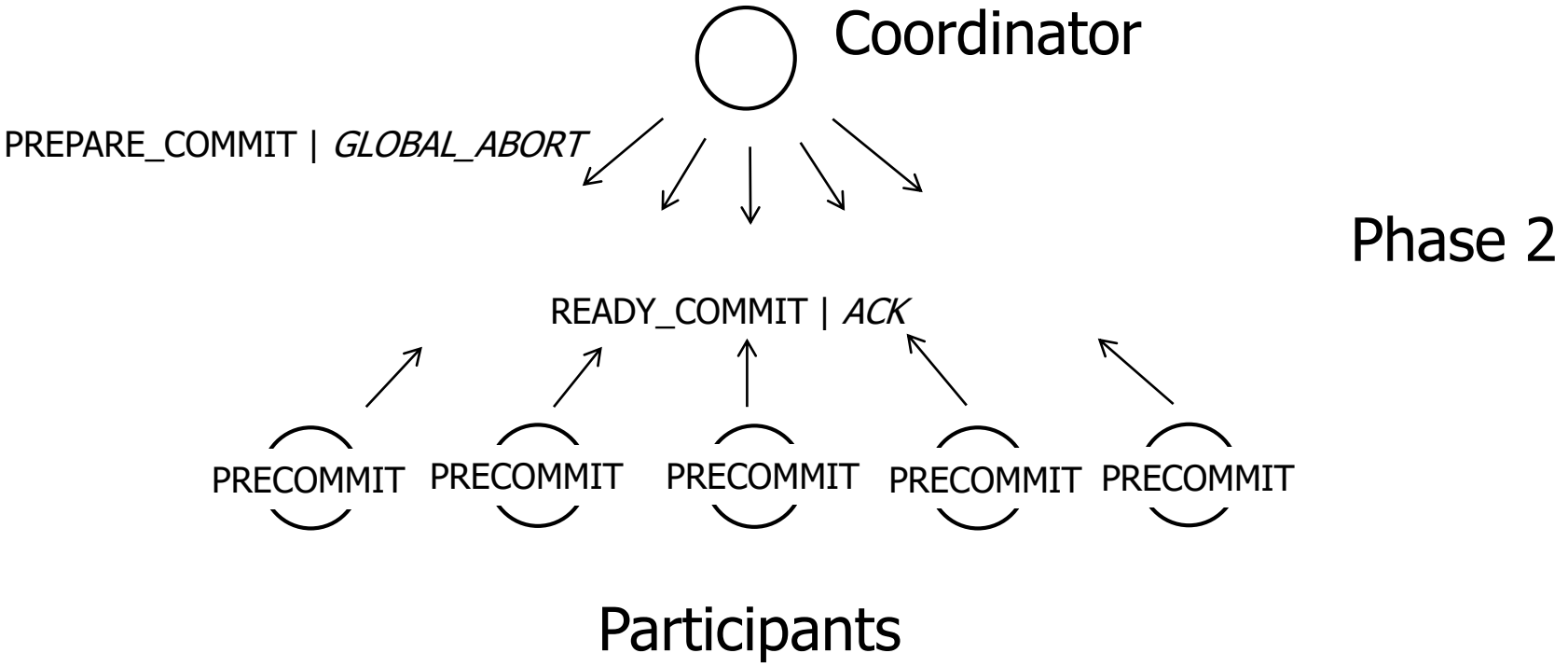
```
while true {
    wait until any incoming DECISION_REQUEST is received; /* remain blocked */
    read most recently recorded STATE from the local log;
    if STATE == GLOBAL_COMMIT
        send GLOBAL_COMMIT to requesting participant;
    else if STATE == INIT or STATE == GLOBAL_ABORT
        send GLOBAL_ABORT to requesting participant;
    else
        skip;  /* participant remains blocked */
```

Three-phase commit

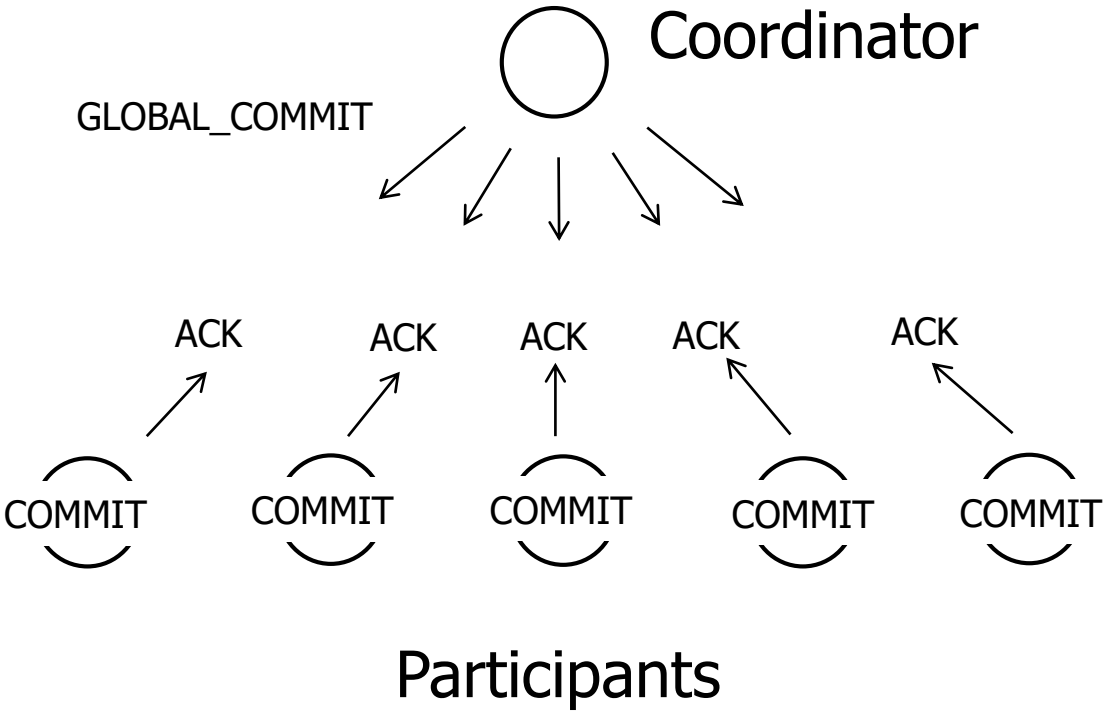


Phase 1

Three-phase commit

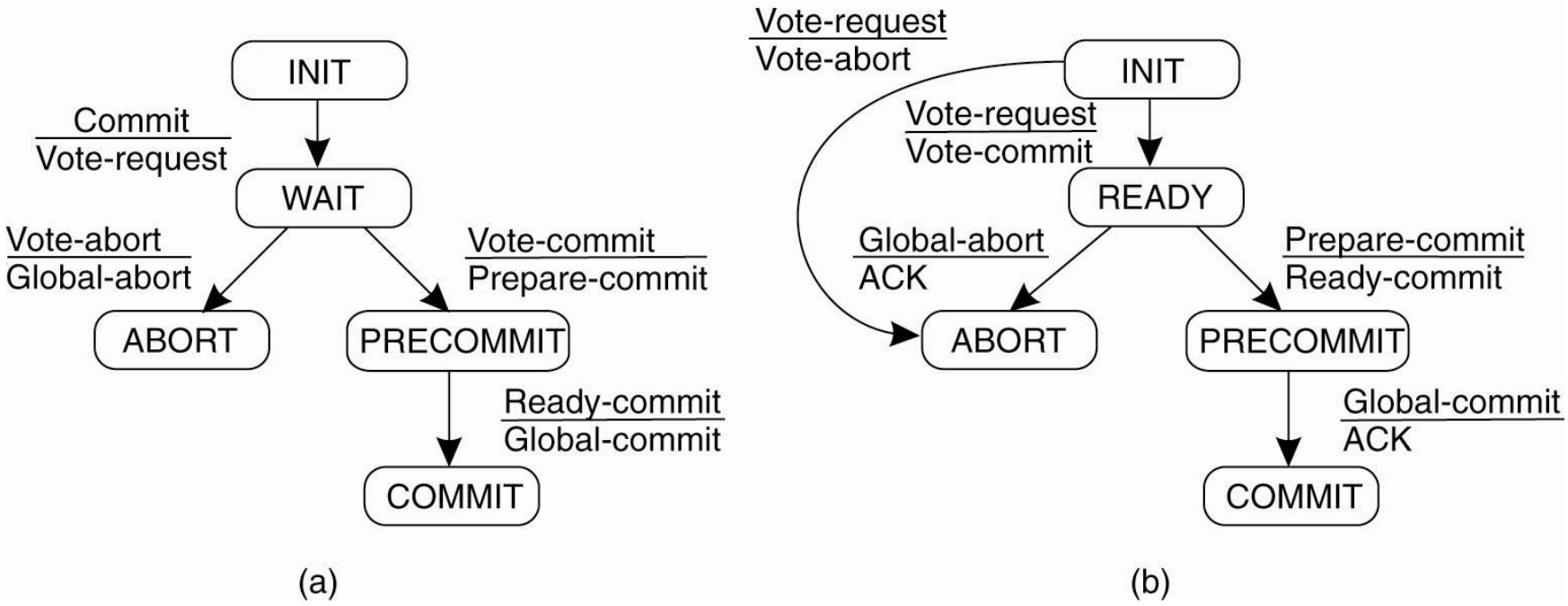


Three-phase commit



Phase 3

Three-phase commit (2)



(a) The finite state machine for the coordinator in 3PC

(b) The finite state machine for a participant

3PC is non-blocking

Coordinator crashes, disconnects, or is too slow – but not a participant

Coordinator

Crashed participant may not have voted abort or aborted, thus can commit

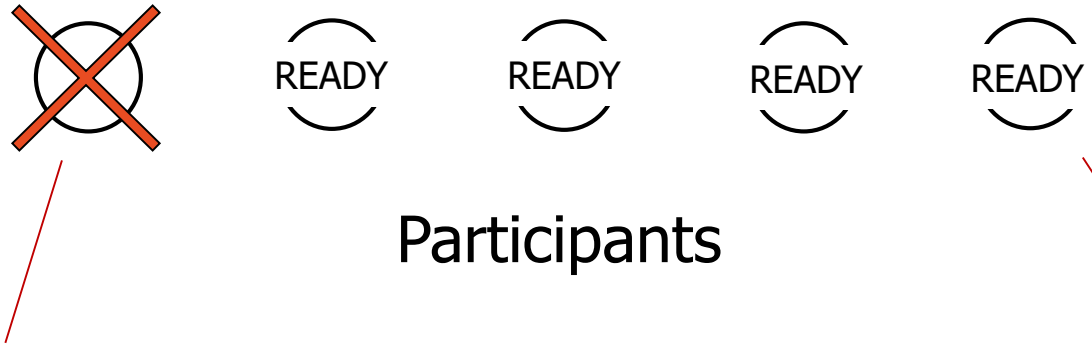
A participant crashes, disconnects, or is too slow

3PC is non-blocking

Coordinator crashes, disconnects, or is too slow – but not a participant

Coordinator

Coordinator cannot have issued a commit, thus they can abort



A participant crashes, disconnects, or is too slow

3PC is non-blocking

Coordinator crashes, disconnects, or is too slow – but not a participant

Coordinator

Crashed participant may not have voted abort or aborted, thus can commit

A participant crashes, disconnects, or is too slow

3PC is non-blocking

Coordinator crashes, disconnects, or is too slow – but not a participant

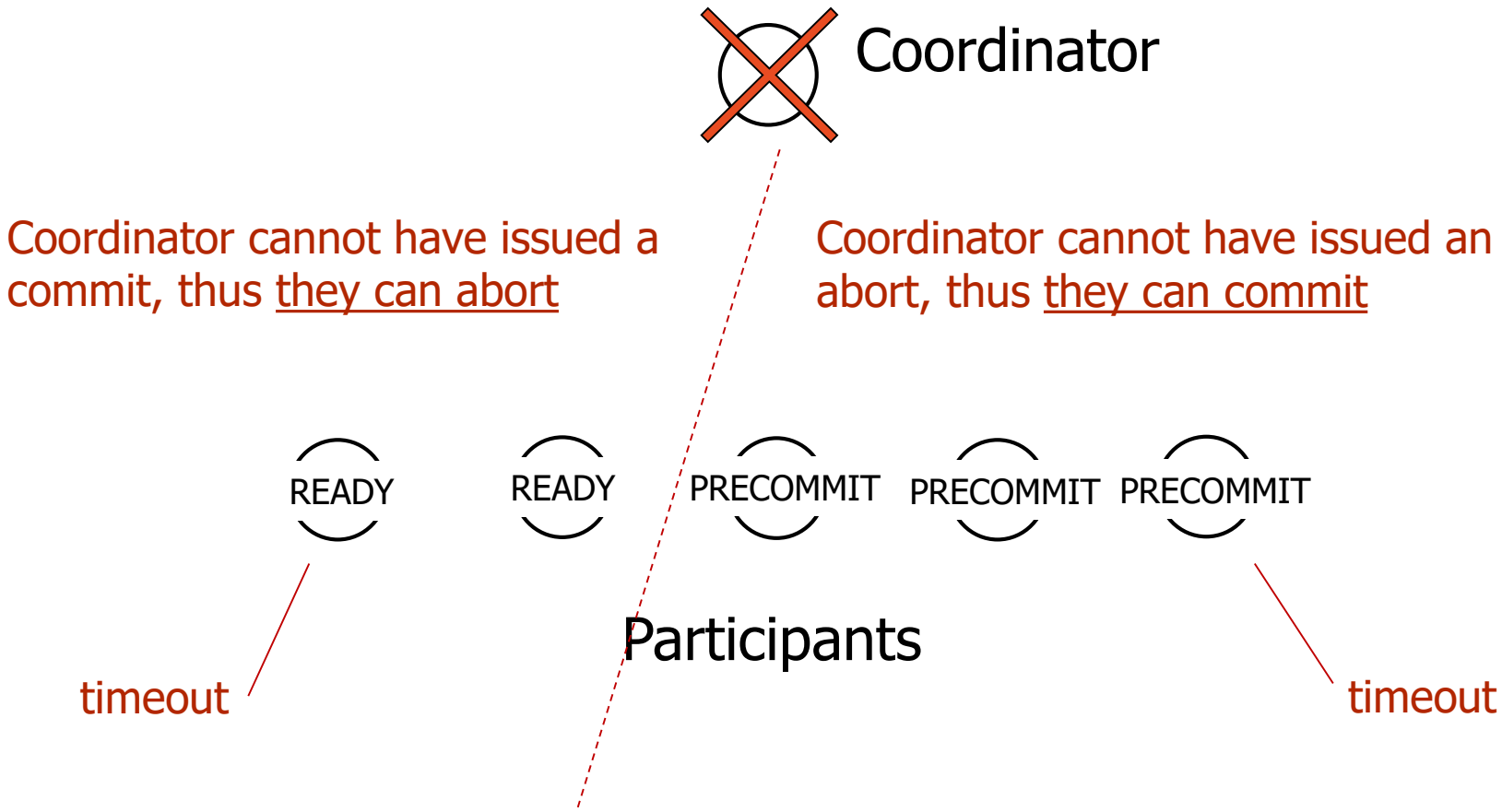
Coordinator

Coordinator cannot have issued an abort, thus can commit

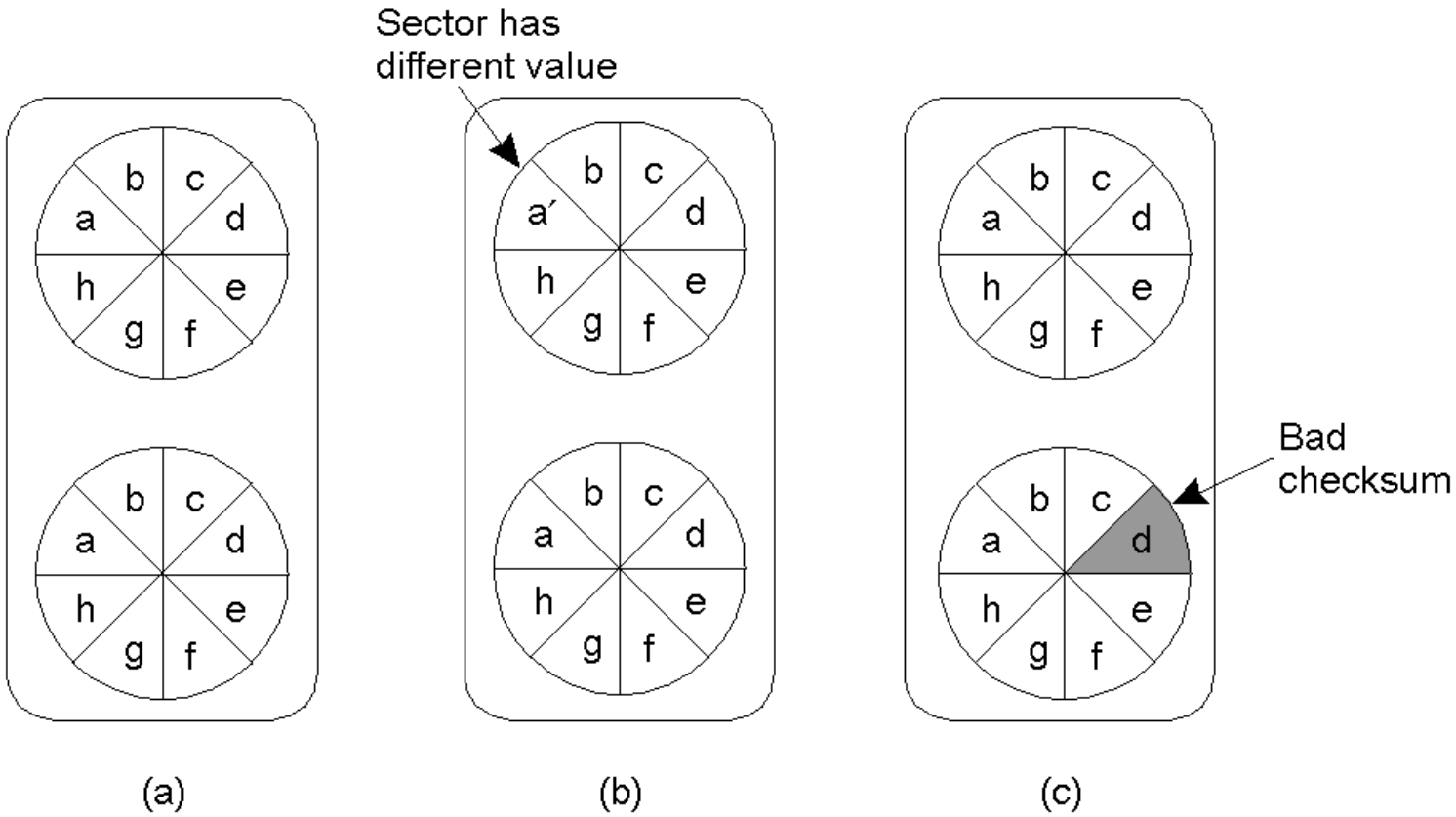
A participant crashes, disconnects, or is too slow

3PC – partitions are problematic

Coordinator crashes, disconnects, or is too slow – but not a participant

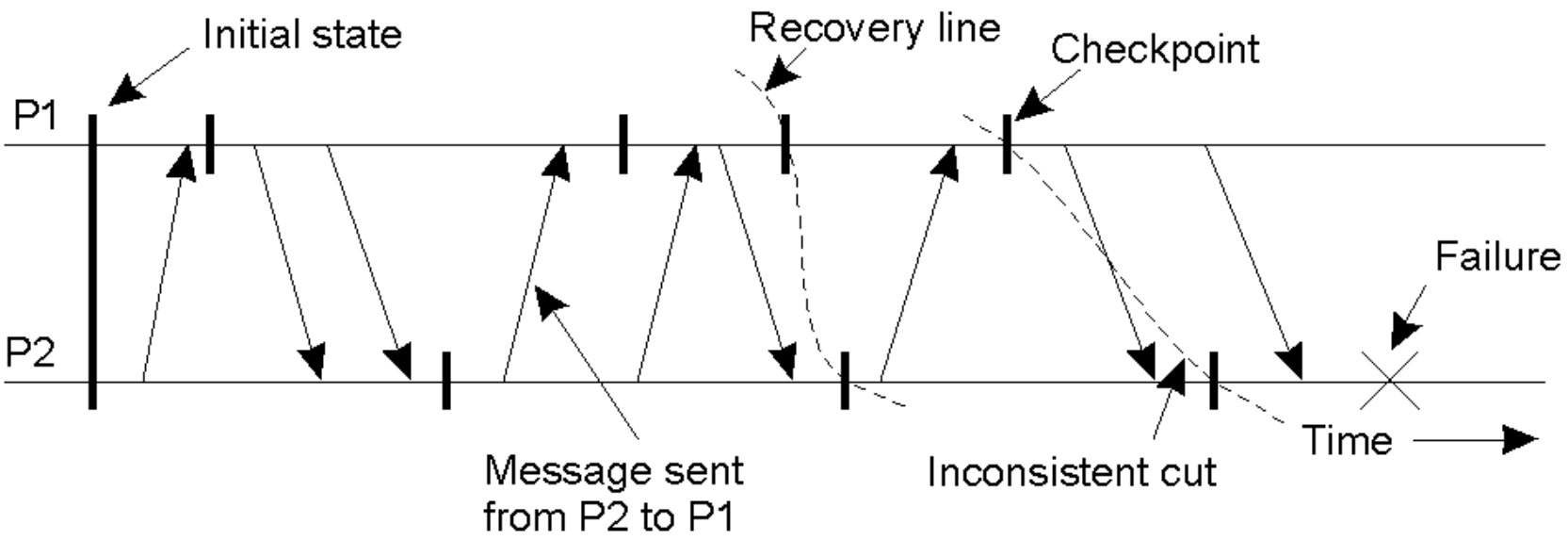


Recovery Stable Storage



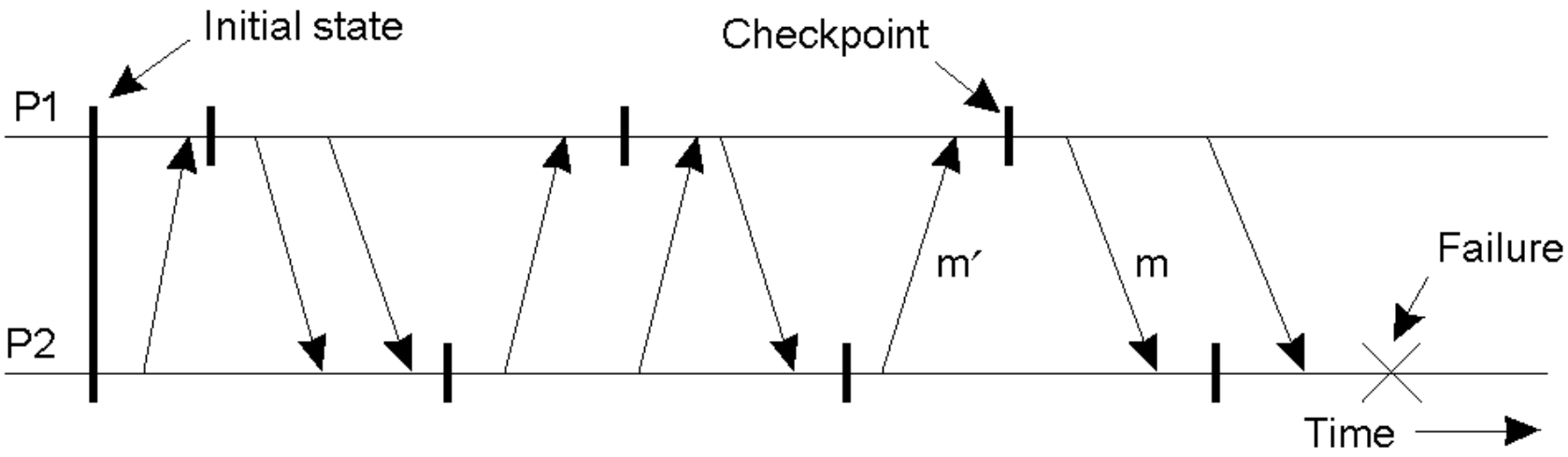
a) Stable Storage
b) Crash after drive 1 is updated
c) Bad spot

Checkpointing



A recovery line

Independent Checkpointing



The domino effect