Replicated State Machine

Are we done now that we have logical clocks?

Failures!
RSMs with Logical Clocks

- Any replica can execute an update only after confirming clock is higher on all other replicas.

- Implication: If any one replica is down, all other replicas cannot progress.
Types of failures

- **Crash failures**
  - Can resume with saved state

- **Fail stop**
  - All state is lost upon failure
Recovering from failures

- Checkpoint process state periodically
  - Where to store checkpoint?
  - Persistent storage vs. volatile memory
  - Local machine vs. remote machine

- Where you store checkpoint depends on
  - Types of failures you want to tolerate
  - Performance overhead you are willing to bear

- Resume from last checkpoint after restart
Challenge in Checkpointing

P1 \( \rightarrow \) m_1 \( \rightarrow \) m_3 \( \rightarrow \) m_4 \( \rightarrow \) m_6 \( \rightarrow \) m_7 \( \rightarrow \) P2

P2 \( \rightarrow \) m_2 \( \rightarrow \) m_3 \( \rightarrow \) m_4 \( \rightarrow \) m_5 \( \rightarrow \) m_8 \( \rightarrow \) P3

X
Challenge in Checkpointing
Challenge in Checkpointing
Challenge in Checkpointing
Challenge in Checkpointing
Challenge in Checkpointing
Challenge in Checkpointing

P1

P2

P3

m_1

m_2

m_3
Challenge in Checkpointing
Challenge in Checkpointing
Challenge in Checkpointing

P1

P2

P3
Checkpoint-based Recovery

- **Problem:** Domino effect
  - **Cause:** Uncoordinated checkpointing

- **Coordinated checkpointing**
  - **Example:** Chandy-Lamport snapshot
    - Global snapshot of a distributed system
Example
Chandy-Lamport algorithm

- How to take a snapshot of a distributed system?

- Example use cases:
  - Deadlock detection
  - Garbage collection
  - Evaluation of any stable property
Example: Token ring

P1

m1

m2

m3

P2

P3
Example snapshot 1

P1

P2

P3

m_1

m_2

m_3
Example snapshot 2
Global snapshot

- Captured state must satisfy “happens before”

- If event $b$ in snapshot and $a \Rightarrow b$, then event $a$ must be in snapshot
Example desired snapshot
Global snapshot

- **Goals of capturing global snapshot**
  - Capture instantaneous state of every process
  - Capture relevant messages in transit

- **What comprises a distributed system’s state?**
  - State of individual processes
  - State of every communication channel
Chandy-Lamport Snapshot

- **N** processes in the system
  - Processes don’t fail while taking snapshot
  - Any process may initiate collection of snapshot

- One unidirectional channel in either direction between each pair of processes
  - All channels ensure FIFO delivery
  - Lossless and no duplication
Chandy-Lamport Snapshot
Enable snapshot by broadcasting marker message
- Distinct from the application’s messages

Marker messages serve two purposes:
1. Enable processes to discover need for snapshot
2. Serve as a barrier on every channel
   - Record all messages received before marker
Initiator process does the following:
- Records its local state
- Sends out *marker* messages on every outgoing channel
- Starts recording messages on every incoming channel

Two cases for how $P_i$ should handle receipt of marker message from $P_j$
Chandy-Lamport Snapshot

- If the state has not been recorded by $P_i$ upon the receipt of a marker along a channel $c$
  - Record local state
  - Record state of channel from $P_j$ to $P_i$ as empty
  - Send marker messages on all outgoing channels

- If state was previously recorded
  - Stop recording channel from $P_j$ to $P_i$
  - Record state of channel as all messages received since marker

- Snapshot complete when every process has received marker on every incoming channel
Chandy-Lamport in action
Coordinated checkpointing

- When to initiate global snapshot?
- External environment does not checkpoint
  - Cannot roll back output
  - Cannot re-issue inputs

- Implication: Need coordinated checkpoint upon input or output
  - Too slow!
Challenge in Checkpointing

P1

P2

P3

m_1

m_2

m_3

m_4

m_5

m_6

m_7

m_8

X
Challenge in Checkpointing
Challenge in Checkpointing
Message logging

- Between checkpoints, log all events that lead to non-determinism

- For example, log every message received

- Log everything necessary to ensure same messages are sent as in original execution