

CPSC 416 – Distributed Systems

Lecture 05 Global state: The solution

UBC Vancouver

2026-01-20

Recap: The problem

From last time:

- Global state = process states + channel states
- A **consistent cut** respects causality
- An **inconsistent cut** shows effects without causes

The challenge: Compute a consistent cut without stopping the system, without a global clock, while processes keep executing.

Today: **Chandy–Lamport** (1985) shows it's possible.

Chandy–Lamport snapshot algorithm

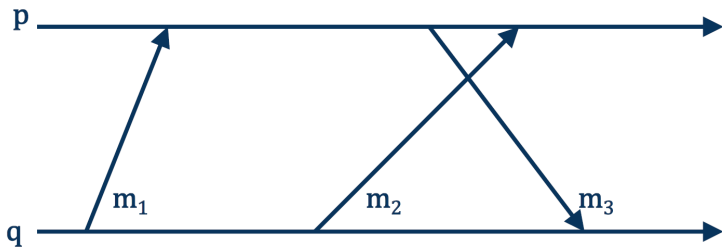
Key insight: Use special **marker** messages to delineate “before snapshot” from “after snapshot.”

Assumptions:

- Channels are FIFO (messages arrive in order sent)
- Channels are reliable (no loss, no duplication)
- Graph is strongly connected (can reach everyone)

No global clock. No freezing. Just markers.

The setup

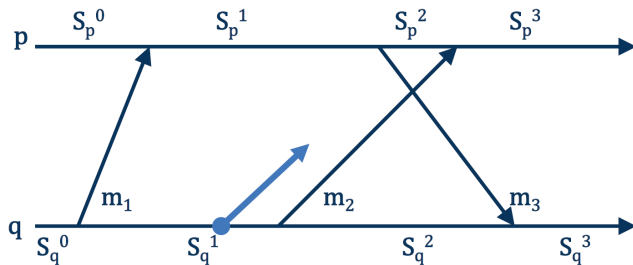


Two processes p and q , exchanging messages m_1, m_2, m_3 .

The algorithm: initiator

Any process p can initiate a snapshot:

- 1 Record p 's local state
- 2 Send a **marker** on every outgoing channel
- 3 Start recording incoming messages on all channels



The algorithm: on receiving a marker

When process q receives a marker on channel c :

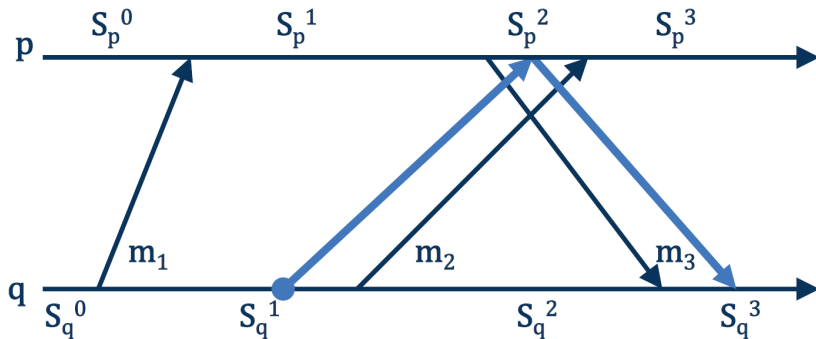
If first marker q has seen:

- 1 Record q 's local state
- 2 Record channel c 's state as *empty*
- 3 Send marker on all of q 's outgoing channels
- 4 Start recording on all other incoming channels

If q already recorded its state:

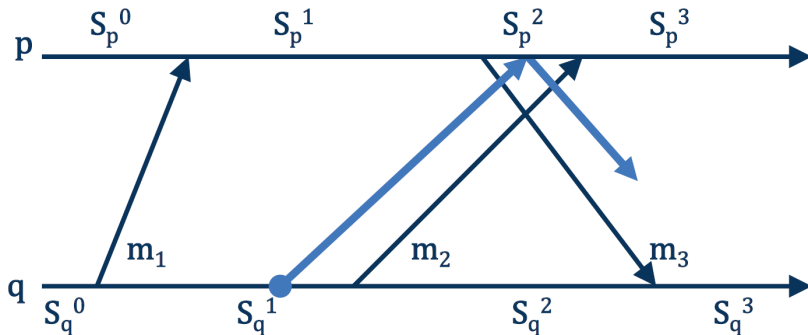
- 1 Channel c 's state = all messages received on c since q 's snapshot, before this marker

Walkthrough: marker propagation



Markers (light arrows) propagate; each process records state on first marker.

Walkthrough: snapshot complete



Snapshot complete when every process has received a marker on every incoming channel.

Why does this work?

FIFO channels + markers ensure:

- All “pre-snapshot” messages arrive before the marker
- All “post-snapshot” messages arrive after the marker
- No message “crosses” the cut backwards

Therefore: the recorded cut is **consistent**.

The marker acts like a “causal barrier” — separating before from after.

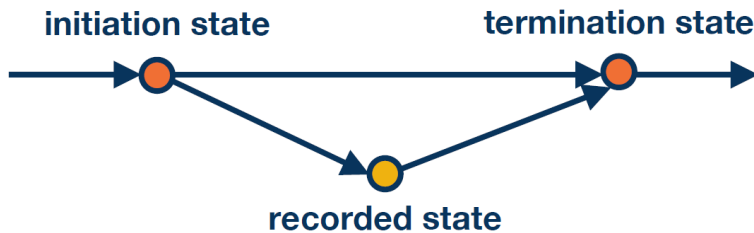
The key insight

The Chandy–Lamport algorithm does **not** capture a state that actually existed at any instant.

It captures a **consistent** state — one that *could have* existed.

This is enough! For deadlock detection, termination, GC — we need consistency, not actuality.

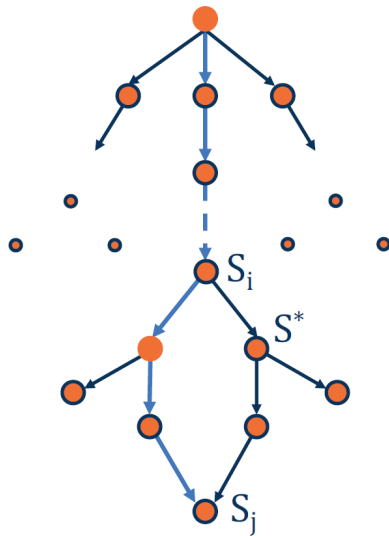
Reachability



The recorded state S^* satisfies:

- S^* is **reachable** from the initiation state
- The termination state is **reachable** from S^*

The state lattice



Stable properties

A property ϕ is **stable** if: once true, it stays true forever.

Examples:

- Deadlock (once deadlocked, stays deadlocked)
- Termination (once all processes idle, stays idle)
- “Object X is unreachable” (for GC)
- “All crew are assigned” (Southwest!)

The stable property theorem

Theorem: If ϕ is stable and ϕ holds in the recorded snapshot S^* , then ϕ holds in the actual current state.

Proof intuition:

- The current state is reachable from S^*
- ϕ is true in S^*
- ϕ is stable (can't become false)
- Therefore ϕ is true now

This is why “consistent but not actual” is **good enough**.

Limitations

Chandy–Lamport requires:

- FIFO channels — what if channels reorder?
- Reliable delivery — what if messages are lost?
- Strongly connected — what if network partitions?

Extensions exist for weaker assumptions, but add complexity.

Connection to Lecture 03: The marker effectively carries a “logical timestamp” that partitions before/after.

Practical applications

- **Checkpointing:** Fault-tolerant systems save consistent snapshots
- **Debugging:** Record global state for post-mortem analysis
- **Termination detection:** Dijkstra-Scholten, weight-throwing
- **Garbage collection:** Distributed reference counting

The idea of “consistent but not actual” appears throughout distributed systems.

Beyond snapshots: Lattice theory

The set of consistent cuts forms a **lattice**.

- Partially ordered by $\subseteq$
- Has meet ($\sqcap$) and join ($\sqcup$) operations
- Rich mathematical structure

This same structure underlies:

- **CRDTs** (Conflict-free Replicated Data Types)
- **Version vectors** in distributed databases
- **Causal consistency** models

We'll see these later in the course.

Summary

- **Chandy–Lamport:** Use markers to compute consistent snapshots without stopping the world
- **Key insight:** The snapshot is **consistent**, not necessarily **actual**
- **Reachability:** Snapshot lies on some valid execution path
- **Stable properties:** If true in snapshot, true now
- **Lattice structure:** Foundation for CRDTs and causal consistency

Next: We'll build on this for replication and consensus.

Logistics

- Readings + quiz (Canvas → Readings page)
- Self-assessment opens at 13:00, due next class 11:30 (Pacific)
- Upcoming DP/capstone milestones (Canvas → Assignments)

Recommended reading:

- Chandy & Lamport, “Distributed Snapshots” (1985)
- Mattern, “Virtual Time and Global States” (1989)