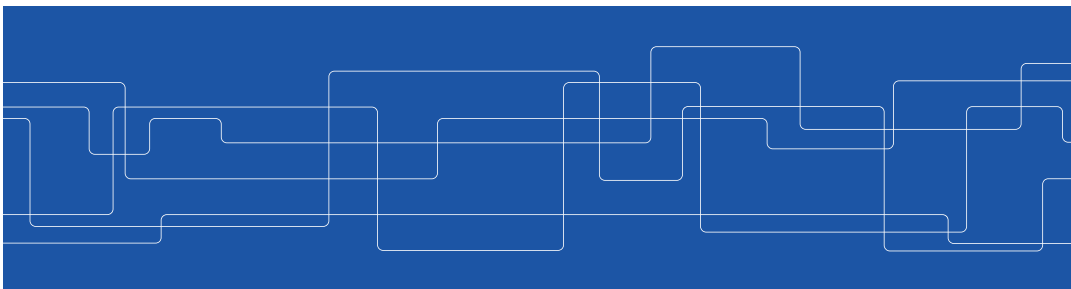


Global state

Johan Montelius and Vladimir Vlassov



Global state

Time is very much related to the notion of *global state*.

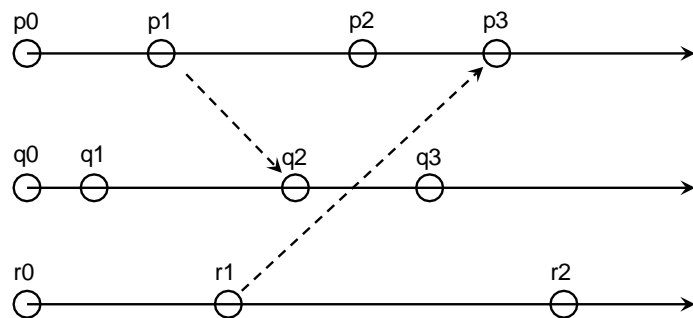
If we cannot agree on a time, how should we agree on a global state?

Global state is important:

- Garbage collection
- Dead-lock detection
- Termination
- Debugging

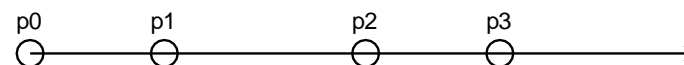
Global state

Given a partial order of events, can we say anything about the state of the system?



History and state

The *history* of a process is a sequence of events: $\langle p_0, p_1, \dots, p_n \rangle$



The *state* of a process is a description of the process after (before) an event.

Global state

Is the state of a process the history of events?

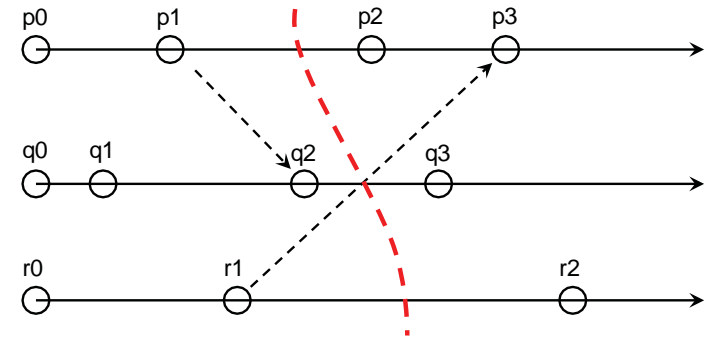
What is the **global state** of a distributed system?

The union of histories of all processes?

Do all unions make sense?

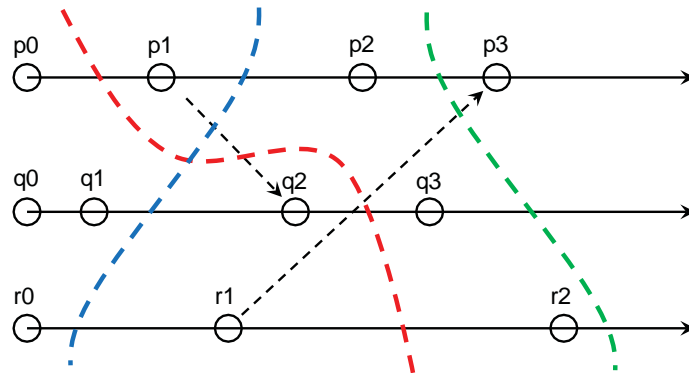
Global history and cut

A **cut** is a subset in the global history up to a specific event in each history.

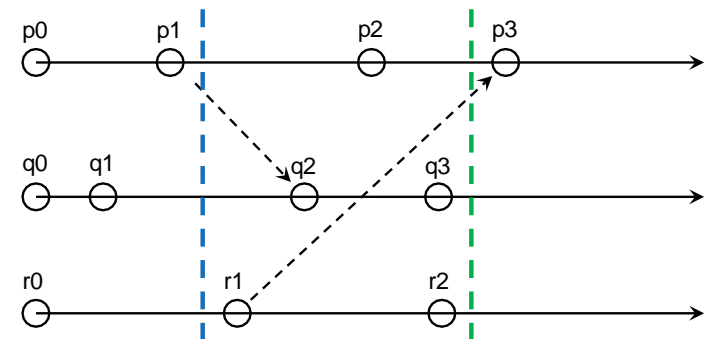


An event is in the **cut** if it belongs to the events of a history up to the specific event.

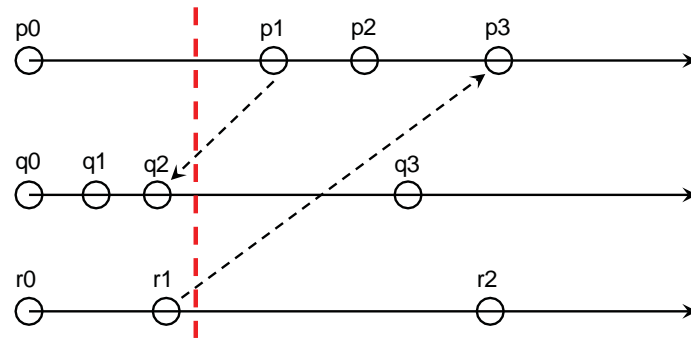
All cuts are equal, but ...



..some are more equal ..



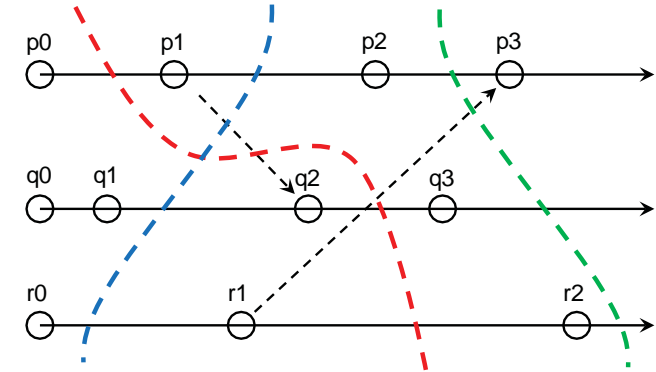
.. than others



Consistent cuts

For each event e in the cut:

- if f happened before e then
- f is also in the cut.

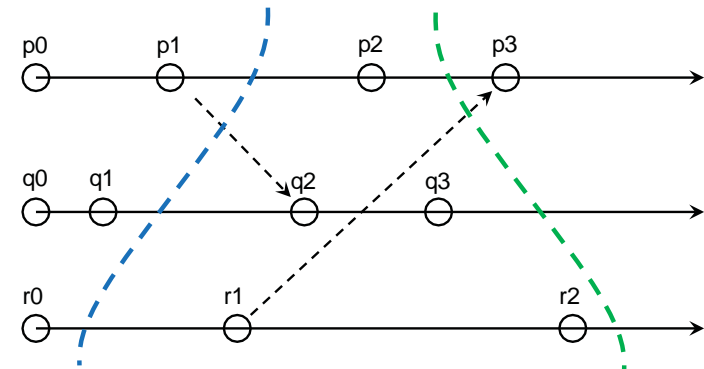


Consistent global state

A **consistent cut** corresponds to a **consistent global state**.

- it is a possible state without contradictions
- the actual execution might not have passed through the state

Consistent, but not actual states

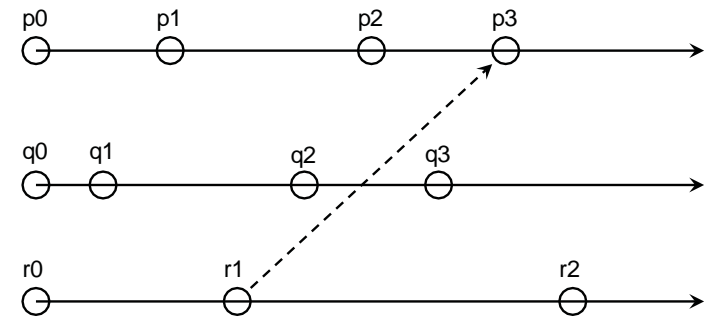


All *real time cuts* are *consistent*, but who knows the real time?

Linearization

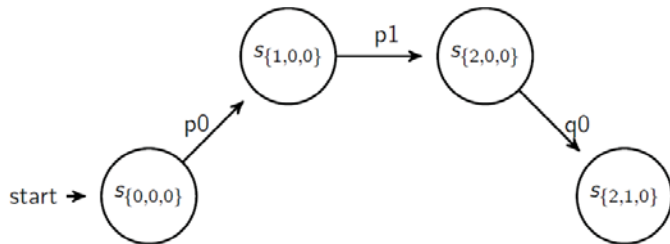
- A **run** is a total ordering of all events in a global history that is consistent with each local history.
- A **linearization** or **consistent run** is a run that describes transitions between consistent global states.
- A state S' is reachable from state S if there is a linearization from S to S' .

Linearization



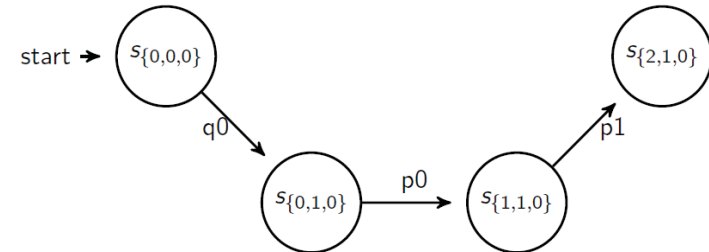
Possible state transitions

[p0, p1, q0, r0, q1, r1, p2, p3, q2, r2, q3]

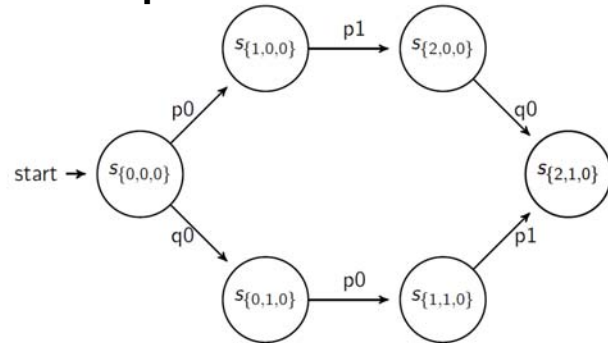


Possible state transitions

[q0, p0, p1, r0, q1, r1, p2, p3, q2, r2, q3]



Possible paths



Each path is a consistent run, a linearization, one of which the execution actually took.

Why is this important?

- If we can collect all events and know the happened before order, then we can construct all possible linearizations.
- We know that the actual execution took one of these paths.
- Can we say something about the execution even though we do not know which path that was taken?

Global state predicate

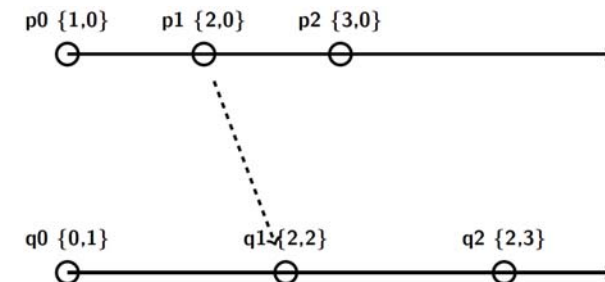
A global state predicate is a property that is true or false for a global state.

- **Safety** - a predicate is never (or always) true in any state.
- **Liveness** - a predicate that eventually evaluates to true.

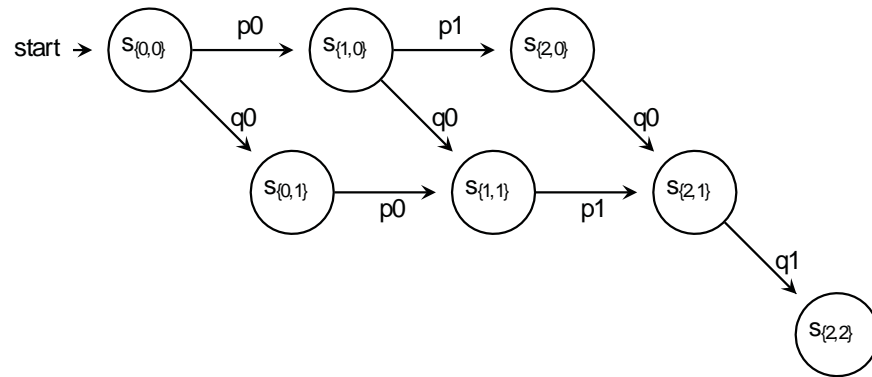
How do we determine if a property holds in an execution?

Let's capture all linearizations

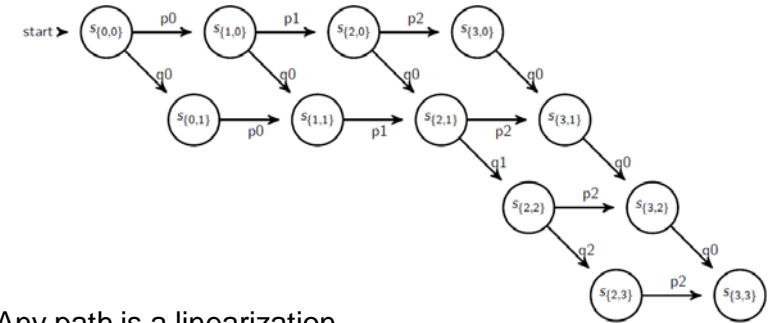
Idea - use **vector clocks**, collect all events of the execution.



Construct all linearizations



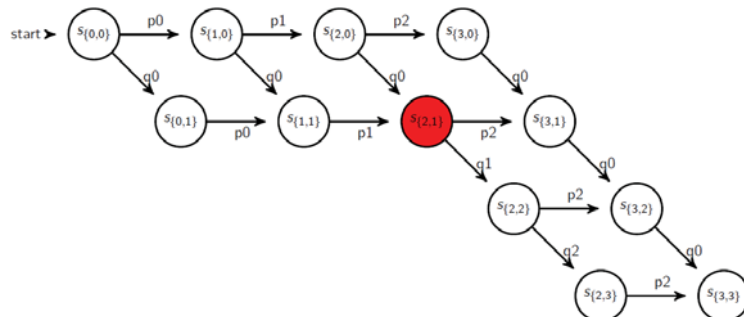
An execution lattice



Any path is a linearization.

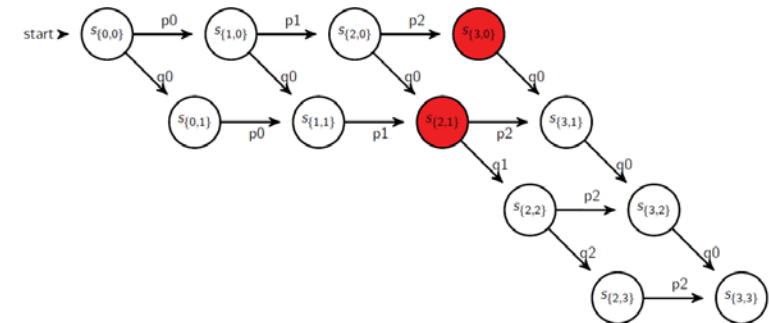
The actual execution took one path.

Possibly true



If a predicate is true in a consistent global state of the lattice, then it is *possibly true* in the execution.

Definitely true



If we cannot find a path from the initial state to the final state without reaching a state for which a predicate is true then the predicate is *definitely true* during the execution.

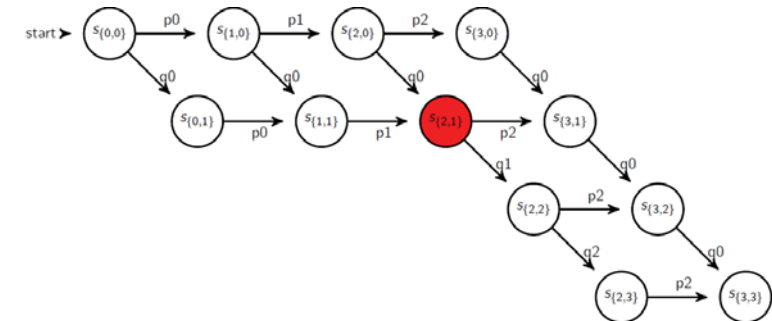
Stable and non-stable

We differentiate between:

- **Stable**: if a predicate is true it remains true for all reachable states
- **Non-stable**: if a predicate can become true and then later become false

Stable is good

What do I know if a stable predicate is true for state $S_{\{2,1\}}$?



Let's capture a possible state

Idea: capture a consistent global state that was possibly true in the execution.

If a stable predicate is true for this state - then it is true in the actual execution.

How do we capture a state?

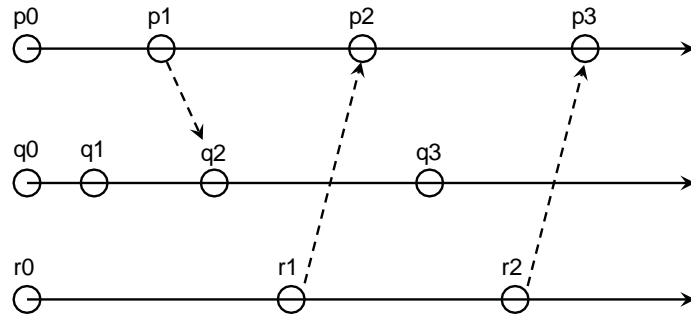
Snapshot - Chandy and Lamport

A node initiates a snapshot when it receives a **marker**.

- Record the local state and
- send a **marker** on all out going channels.
- Record all incoming messages on each channel, ..
- until you receive a marker.
- When the last channel is closed you have a local and a set of messages.

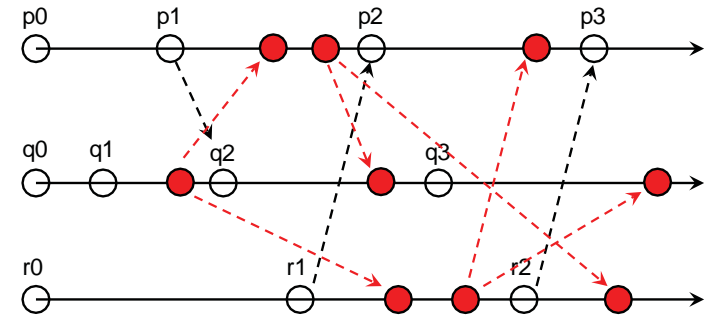
Ask one node to initiate the snapshot, collect all local states and messages and construct a global state.

Snapshot markers



What messages are collected by which node?

Snapshot markers



What messages are collected by which node?

Snapshot

- Allows us to collect a global state during execution.
- Only allows us to determine stable predicates.

Summary

The happened before order gives us *consistent cuts or consistent global states*.

Using vector clocks we can time stamp states, *construct all possible linearizations* and evaluate if predicates hold true in the execution.

A snapshot can record a consistent state that can be used to evaluate *stable predicates*.