The Model

Node properties:
- No shared memory
- No global clock

Channel properties:
- FIFO
- Loss free
- Non-duplicating

The Problem

$500 \rightarrow \text{C1:empty} \rightarrow \text{C2:empty} \rightarrow \text{empty} \rightarrow \text{empty} \rightarrow \text{empty} \rightarrow \text{empty}$
Distributed Snapshot
(Global State Recording)

Problems:
• recording a “consistent” state of the global computation
  • checkpointing for fault tolerance (rollback, recovery)
  • testing and debugging
  • monitoring and auditing
• detecting stable properties in a distributed system via snapshots.
  A property is “stable” if, once it holds in a state, it holds in all subsequent states.
  • termination
  • deadlock
  • garbage collection

Definitions

Local State and Actions:
- local state: \( L_{S_i} \)
- message send: \( \text{send}(m_{ij}) \)
- message receive: \( \text{rec}(m_{ij}) \)
- time: \( \text{time}(x) \)

\[ \text{send}(m_{ij}) \in L_{S_i} \iff \text{time}(\text{send}(m_{ij})) < \text{time}(L_{S_i}) \]

\[ \text{rec}(m_{ij}) \in L_{S_j} \iff \text{time}(\text{rec}(m_{ij})) < \text{time}(L_{S_j}) \]

Predicates:
- \( \text{transit}(L_{S_i}, L_{S_j}) = \{ m_{ij} \mid \text{send}(m_{ij}) \in L_{S_i} \land \neg \text{rec}(m_{ij}) \in L_{S_j} \} \)
- \( \text{inconsistent}(L_{S_i}, L_{S_j}) = \{ m_{ij} \mid \neg \text{send}(m_{ij}) \in L_{S_i} \land \text{rec}(m_{ij}) \in L_{S_j} \} \)

Consistent Global State:
\[ \forall i, \forall j : 1 \leq i, j \leq n \ni \text{inconsistent}(L_{S_i}, L_{S_j}) = \varnothing \]
Global-State-Detection Algorithm

**Marker-Sending Rule for a Process p:**
For each channel c, incident on, and directed away from p: p sends one marker along c after p records its state and before p sends further messages along c.

**Marker-Receiving Rule for a Process q:**
if (q has not recorded its state) then
  begin q records its state;
    q records the state of c as the empty sequence;
  end
else q records the state of c as the sequence of message received along c after q's state was recorded and before q received the marker along c.

**Detecting a Stable Property**

begin
  record a global snapshot, S*;
  test for the stable property in S*;
end;

---

**Global-State-Detection Algorithm**

**Marker-Sending Rule for a Process p:**
For each channel c, incident on, and directed away from p: p sends one marker along c after p records its state and before p sends further messages along c.

**Marker-Receiving Rule for a Process q:**
if (q has not recorded its state) then
  begin q records its state;
    q records the state of c as the empty sequence;
  end
else q records the state of c as the sequence of message received along c after q's state was recorded and before q received the marker along c.

**Detecting a Stable Property**

begin
  record a global snapshot, S*;
  test for the stable property in S*;
end;
Snapshot/State Recording Example

\[ \begin{array}{c}
\node (p) at (0,0) {p};
\node (c1) at (1,1) {c1};
\node (c2) at (2,1) {c2};
\node (r) at (2,-2) {r};
\node (q) at (1,1) {q};
\node (c4) at (1,-2) {c4};
\node (c3) at (2,-2) {c3};
\end{array} \]

\[ M \] = Marker

\begin{tabular}{|c|c|c|c|c|}
\hline
Node & Recorded state \\
\hline
p & {} & {} & {} \\
q & {} & {} & {} \\
r & {} & {} & {} \\
\hline
\end{tabular}

Snapshot/State Recording Example (Step 1)

\[ \begin{array}{c}
\node (p) at (0,0) {p};
\node (c1) at (1,1) {c1};
\node (c2) at (2,1) {c2};
\node (r) at (2,-2) {r};
\node (q) at (1,1) {q};
\node (c4) at (1,-2) {c4};
\node (c3) at (2,-2) {c3};
\end{array} \]

\[ M \] = Marker

\begin{tabular}{|c|c|c|c|c|}
\hline
Node & Recorded state \\
\hline
state & c1 & c2 & c3 & c4 \\
\hline
p & 490 & {} & {} & {} \\
q & {} & {} & {} & {} \\
r & {} & {} & {} & {} \\
\hline
\end{tabular}
Snapshot/State Recording Example (Step 2)

Node | Recorded state
--- | ---
state | c1 | c2 | c3 | c4
p | 490 | {} | {} |
q | 480 | {} | {} |
r | 485 | {} | {} |

Snapshot/State Recording Example (Step 3)

Node | Recorded state
--- | ---
state | c1 | c2 | c3 | c4
p | 490 | {} | {} |
q | 480 | {} | {} |
r | 485 | {} | {} |
Snapshot/State Recording Example (Step 4)

<table>
<thead>
<tr>
<th>Node</th>
<th>Recorded state</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>c1</td>
</tr>
<tr>
<td>p</td>
<td>490</td>
</tr>
<tr>
<td>q</td>
<td>480</td>
</tr>
<tr>
<td>r</td>
<td>485</td>
</tr>
</tbody>
</table>

Snapshot/State Recording Example (Step 5)

<table>
<thead>
<tr>
<th>Node</th>
<th>Recorded state</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>c1</td>
</tr>
<tr>
<td>p</td>
<td>490</td>
</tr>
<tr>
<td>q</td>
<td>480</td>
</tr>
<tr>
<td>r</td>
<td>485</td>
</tr>
</tbody>
</table>