Global Snapshot

Global Snapshot

- Record the global state of a system
  - Need to collect both node and channel states
- Applications
  - Checking "stable" properties
    - Termination, deadlock,…
  - Checkpoint & recovery for long running applications
- Issues
  - Whole system should not be stopped
  - No global clock
- But what is global state in distributed system anyway??

Some Notations

- LS_i : local state of process i
- send(m_{ij}) : send event of message m_{ij} from process i to process j
- rec(m_{ij}) : similar, receive instead of send
- time(x) : time at which state x was recorded
- time (send(m)) : time at which send(m) occurRed

- send(m_{ij}) ϵ LS_i iff
  time(send(m_{ij})) < time(LS_i)
- rec(m_{ij}) ϵ LS_j iff
  time(rec(m_{ij})) < time(LS_j)
- transit(LS_i,LS_j) = { m_{ij} | send(m_{ij}) ϵ LS_i and rec(m_{ij}) $\notin$ LS_j}
- inconsistent(LS_i, LS_j) = {m_{ij} | send(m_{ij}) $\notin$ LS_i and rec(m_{ij}) ϵ LS_j}

- Global state: collection of local states

$$GS = \{LS_1, LS_2, \ldots, LS_n\}$$

- GS is consistent iff

for all i, j, $1 \leq i, j \leq n$,

$$\text{inconsistent}(LS_i, LS_j) = \Phi$$

- GS is transitless iff

for all i, j, $1 \leq i, j \leq n$,

$$\text{transit}(LS_i, LS_j) = \Phi$$

- GS is strongly consistent if it is consistent and transitless.

- Note that channel state may be specified explicitly in a global state, or implicitly in node states using transit()

Chandy-Lamport's Algorithm

- Any one process acts as initiator and starts the global state recording
- Model
  - Asynchronous
  - Reliable, FIFO communication
  - Links are directed
  - Arbitrary topology, but strongly connected graph

The Algorithm

- Uses special marker messages
- Marker sending rule for any process P:
  - P records its state; then for each outgoing channel C from P on which a marker has not been sent already, P sends a marker along C before any further message is sent on C
- On startup
  - Initiator executes marker sending rule

- Process Q, on receiving a marker along a channel C:
  - If Q has not recorded its state then Q records the state of C as empty; Q then follows the marker sending rule
  - If Q has already recorded its state, it records the state of C as the sequence of messages received along C after Q's state was recorded and before Q received the marker along C

Collecting the Global State

- The algorithm ensures that all nodes record some parts of the global state
- What if the entire global state (the local states and the channel states recorded at different nodes) is to be collected at the initiator?
  - Can flood from each node (as graph is strongly connected), but too costly
  - For undirected graph (i.e., bidirectional links), can build a spanning tree during marker propagation, and use that to send the local states back to the root (initiator)

Some Observations

- Markers sent on a channel distinguish messages sent on the channel before the sender recorded its states and the messages sent after the sender recorded its state
- The state recorded may not be any state that actually happened in reality, rather a state that "could have" happened in another run of the system
- How can the initiator know if global snapshot is complete?
- Message complexity $O(|E|)$, where E = no. of links