SNAPSHOT ALGORITHMS FOR FIFO CHANNELS

Each distributed application has number of processes running on different physical servers. These processes communicate with each other through messaging channels.

A snapshot captures the local states of each process along with the state of each communication channel.

Snapshots are required to:

- Checkpointing
- Collecting garbage
- Detecting deadlocks
- Debugging

Chandy–Lamport algorithm

- The algorithm will record a global snapshot for each process channel.
- The Chandy-Lamport algorithm uses a control message, called a marker.
- Aftera site has recorded its snapshot, it sends a marker along all of its outgoingchannels before sending out any more messages.
- Since channels are FIFO, amarker separates the messages in the channel into those to be included in the snapshot from those not to be recorded inthe snapshot.
- This addresses issue I1. The role of markers in a FIFO systemis to act as delimiters for the messages in the channels so that the channelstate recorded by the process at the receiving end of the channel satisfies thecondition C2.

Marker sending rule for process p_i

(1) Process p_i records its state.
(2) For each outgoing channel C on which a marker has not been sent, p_i sends a marker along C before p_i sends further messages along C.

Marker receiving rule for process p_j

On receiving a marker along channel C:
  if p_j has not recorded its state **then**
    Record the state of C as the empty set
    Execute the "marker sending rule"
  else
    Record the state of C as the set of messages received along C after p_j's state was recorded and before p_j received the marker along C

Fig : Chandy–Lamport algorithm

Initiating a snapshot

- Process P_i initiates the snapshot
- P_i records its own state and prepares a special marker message.
- Send the marker message to all other processes.
- Start recording all incoming messages from channels C_{ij} for j not equal to i.

Propagating a snapshot

- For all processes P_j consider a message on channel C_{kj}.
- If marker message is seen for the first time:
  - P_j records own sate and marks C_{kj} as empty
  - Send the marker message to all other processes.
  - Record all incoming messages from channels C_{lj} for 1 not equal to j or k.
  - Else add all messages from inbound channels.

Terminating a snapshot

- All processes have received a marker.
- All process have received a marker on all the N-1 incoming channels.
- A central server can gather the partial state to build a global snapshot.

Correctness of the algorithm

- Since a process records its snapshot when itreceives the first marker on any incoming channel, no messages that followmarkers on the channels incoming to it are recorded in the process's snapshot.
- A process stops recording the state of an incoming channel whena marker is received on that channel.
- Due to FIFO property of channels, itfollows that no message sent after the marker on that channel is recorded inthe channel state. Thus, condition C2 is satisfied.
- When a process p_j receives message m_{ij} that precedes the marker on channel C_{ij}, it acts as follows: if process p_j has not taken its snapshot yet, then it includes m_{ij} in its recorded snapshot. Otherwise, it records m_{ij} in the state of the channel C_{ij}. Thus, condition C1 is satisfied.

Complexity

The recording part of a single instance of the algorithm requires O(e) messages and O(d) time, where e is the number of edges in the network and d is the diameter of the network.

Properties of the recorded global state

The recorded global state may not correspond to any of the global states that occurred during the computation.

This happens because a process can change its state asynchronously before the markers it sent are received by other sites and the other sites record their states.

But the system could have passed through the recorded global states in some equivalent executions.

The recorded global state is a valid state in an equivalent execution and if a stable property (i.e., a property that persists) holds in the system before the snapshot algorithm begins, it holds in the recorded global snapshot.

Therefore, a recorded global state is useful in detecting stable properties.

GLOBAL STATE AND SNAPSHOT RECORDING ALGORITHMS

- A distributed computing system consists of processes that do not share a common memory and communicate asynchronously with eachother by message passing.
- Each component ofhas a local state. The state of the process is the local memory and a history of its activity.
- The state of achannel is characterized by the set of messages sent along the channel lessthe messages received along the channel. The global state of a distributedsystem is a collection of the local states of its components.
- If shared memory were available, an up-to-date state of the entire systemwould be available to the processes sharing the memory.
- The absence ofshared memory necessitates ways of getting a coherent and complete view ofthe system based on the local states of individual processes.
- A meaningfulglobal snapshot can be obtained if the components of the distributed systemrecord their local states at the same time.
- This would be possible if thelocal clocks at processes were perfectly synchronized or if there were aglobal system clock that could be instantaneously read by the processes.
- If processes read time froma single common clock, various indeterminatetransmission delays during the read operation will cause the processes toidentify various physical instants as the same time.

System Model

- The system consists of a collection of n processes, p_1, p_2,…,p_n that are connected by channels.
- Let C_{ij} denote the channel from process p_i to process p_j.
- Processes and channels have states associated with them.
- The state of a process at any time is defined by the contents of processor registers, stacks, local memory, etc., and may be highly dependent on the local context of the distributed application.
- The state of channel C_{ij}, denoted by SC_{ij}, is given by the set of messages in transit in the channel.
- The events that may happen are: internal event, send (send (m_{ij})) and receive (rec(m_{ij})) events.
- The occurrences of events cause changes in the process state.

- A **channel** is a distributed entity and its state depends on the local states of the processes on which it is incident.

  Transit: $transit(LS_i, LS_j) = \{m_{ij} \mid send(m_{ij}) \in LS_i \bigwedge rec(m_{ij}) \notin LS_j\}$

- The transit function records the state of the channel C_{ij}.
- In the FIFO model, each channel acts as a first-in first-out message queue and, thus, message ordering is preserved by a channel.
- In the non-FIFO model, achannel acts like a set in which the sender process adds messages and thereceiver process removes messages from it in a random order.

A consistent global state

The global state of a distributed system is a collection of the local states ofthe processes and the channels. The global state is given by:

$GS = \{\bigcup_i LS_i, \bigcup_{i,j} SC_{ij}\}$.

The two conditions for global state are:

C1: $send(m_{ij}) \in LS_i \Rightarrow m_{ij} \in SC_{ij} \oplus rec(m_{ij}) \in LS_j$
C2: $send(m_{ij}) \notin LS_i \Rightarrow m_{ij} \notin SC_{ij} \wedge rec(m_{ij}) \notin LS_j$.

Condition 1 preserves **law of conservation of messages.**Condition C2 states that in the collected global state, for everyeffect, its cause must be present.

Law of conservation of messages: Every messagem_{ij}that is recorded as sent in the local state of a process p_i must be capturedin the state of the channel C_{ij} or in the collected local state of the receiver process p_j.

- In a consistent global state, every message that is recorded as received isalso recorded as sent. Such a global state captures the notion of causalitythat a message cannot be received if it was not sent.
- Consistent global statesare meaningful global states and inconsistent global states are not meaningful in the sense that a distributed system can never be in an inconsistentstate.

Interpretation of cuts

Cuts in a space–time diagram provide a powerful graphical aid in representingand reasoning about the global states of a computation. A cut is a line joiningan arbitrary point on each process line that slices the space–time diagraminto a PAST and a FUTURE.

- A consistent global state corresponds to a cut in which every messagereceived in the PAST of the cut has been sent in the PAST of that cut. Sucha cut is known as a consistent cut.
- In a consistent snapshot, all the recorded local states of processes are concurrent; that is, the recorded local state of no process casuallyaffects the recorded local state of any other process.

Issues in recording global state

The non-availability of global clock in distributed system, raises the following issues:

Issue 1:

How to distinguish between the messages to be recorded in the snapshot from those not to be recorded?

Answer:

- Any message that is sent by a process before recording its snapshot,must be recorded in the global snapshot (from C1).
- Any message that is sent by a process after recording its snapshot, mustnot be recorded in the global snapshot (from C2).

Issue 2:

How to determine the instant when a process takes its snapshot?

The answer

Answer:

A process p_j must record its snapshot before processing a message m_{ij} that was sent by process p_i after recording its snapshot.