CHANDY–LAMPORT’S GLOBAL STATE RECORDING ALGORITHM

Each distributed system has a number of processes running on a number of different physical servers. These processes communicate with each other via communication channels using text messaging. These processes neither have a shared memory nor a common physical clock, this makes the process of determining the instantaneous global state difficult.

A process could record it own local state at a given time but the messages that are in transit (on its way to be delivered) would not be included in the recorded state and hence the actual state of the system would be incorrect after the time in transit message is delivered.

Chandy and **Lamport** were the first to propose a algorithm to capture consistent global state of a distributed system. The main idea behind proposed algorithm is that if we know that all message that hat have been sent by one process have been received by another then we can record the global state of the system.

Any process in the distributed system can initiate this global state recording algorithm using a special message called **MARKER**. This marker traverse the distributed system across all communication channel and cause each process to record its own state. In the end, the state of entire system (Global state) is recorded. This algorithm does not interfere with normal execution of processes.

Assumptions of the algorithm:

- There are finite number of processes in the distributed system and they do not share memory and clocks.
- There are finite number of communication channels and they are unidirectional and FIFO ordered.
- There exists a communication path between any two processes in the system
- On a channel, messages are received in the same order as they are sent.

Algorithm:

- **Marker sending rule for a process *P* :**
    - Process **p** records its own local state
    - For each outgoing channel **C** from process **P**, **P** sends marker along **C** before sending any other messages along **C**. (**Note:** Process Q will receive this marker on his incoming channel C1.)
- **Marker receiving rule for a process *Q* :**
    - If process **Q** has not yet recorded its own local state then
        - Record the state of incoming channel C1 as an empty sequence or null.
        - After recording the state of incoming channel C1, process **Q** Follows the marker sending rule
    - If process **Q** has already recorded its state

    - Record the state of incoming channel **C1** as the sequence of messages received along channel C1 after the state of **Q** was recorded and before **Q** received the marker along C1 from process P.

Need of taking snapshot or recording global state of the system:

- **Checkpointing:** It helps in creating checkpoint. If somehow application fails, this checkpoint can be used re
- **Garbage collection:** It can be used to remove objects that do not have any references.
- It can be used in deadlock and termination detection.
- It is also helpful in other debugging.