Report

Simulate message delivery guarantees such as Causal and Arbitrary , and their impact on some Termination Detection distributed algorithm.

Team 10 ⇒ Darshan Kansagara - 2018201033
Dhawal Jain - 2018201065
Yaswanth Koravi - 2018202011

Implemented in 3 parts

1) Casual Channel
2) Arbitrary channel with node failure
3) Termination Detection algorithm using weight throwing

There is one folder for each implementation part

The final video is too long and exceeds the submission size(20MB) so we made 3 videos, one for each above sub-topic DEMO.

NOTE: For Video Please used headphones and preferably used VLC for better voice quality.

Casual Channel [2 min]
https://youtu.be/kF2zOFYMXLs

Arbitrary Channel with Node_failure [1.5 min]
https://youtu.be/Chq2cotqh50

Termination Detection [4.5 min] --> It is also attached in submission
https://youtu.be/iRqvfqFT4Sk

Merged video of all above 3 is here
https://drive.google.com/open?id=1bJnrxu25xqPoKYHKeCgB6WmChLZBBqVI

There is a r**eadme.txt** file to provide a guideline on how to run the code.

Casual Channel Implementation Algorithms:

BSS - Birman Schiper Stephenson

- It broadcast-based algorithm
- Even when process P1 wants to send msg to P2 only, It will broadcast the msg to all other processes in the system.
- Casual Ordering is maintained by Vector Clock.

Arbitrary Channel With Node failure

- we have implemented arbitrary channel that handles node failure
- If msg was sent from the sender, It guaranteed to be received at receiver when the receiver will UP again.
- There is NO loss of any messages.
- When P1 sending msg to P2, P2 fails, P1 maintains Queue of msg which are pending, When P2 will wakeup again, Then P1 will send all these pending msg to P2.

Termination Detection Algorithm

- Implemented huangs-termination-detection-algorithm
- One of the co-operating processes which monitor the computation is called the **controlling agent**.
- The initial weight of the **controlling agent** is 1
- All other **NORMAL** processes are initially idle and have weight 0.
- The computation starts when the controlling agent sends a computation message to one of the processes.
- The process becomes active in receiving a computation message.
- Computation message can be sent only by controlling agent or an active process.
- The control message is sent to the controlling agent by an active process when they are becoming idle.
- The algorithm assigns a weight **W** (such that 0 < **W** < 1) to every active process and every in-transit message.

Notations used in the algorithm:

- **B(DW):** Computation message with weight **DW**

        - When Msg sends from one process to another Normal Process.
    - **C(DW)**: Control the message with weight **DW**
        - When Msg sends from Normal process to Controlling Agent.

Impact of the channel (Casual and Arbitrary Channel with node failure on Termination Detection Algorithm)

- Termination Detection ALgorithm **works well with the Casual and Arbitrary channel.**
- Even in case of node failure in arbitrary channel termination detection algorithm work properly except few cases as mentioned below.
- If any process fails, **Termination Detection will halt for some time**, until this process wakes up again.
- When process comes up again it will receive pending msg from other processes and send its whole weight to a controlling agent to become idle.
- **If the Controlling Agent process fails then there is a problem**. So we assume that the Controlling agent will never fail, Only the Normal process fails.