COP5615 – FALL 2019

PROJECT 2 – GOSSIP SIMULATOR

(September 10, 2019 – October 01, 2019)

GROUP MEMBERS

- Arpan Banerjee (UFID 9359-9083)
- Krutantak Patil (UFID 5615-6343)

TOPOLOGIES

- Full Network (full) – Process has all the other processes as its neighbors. With **n** processes in the network, all processes will have **n-1** neighbor processes.
- Line (line) – All processes are placed in a linear order forming a line, where a process at **i**$^{\text{th}}$ position has neighbors at positions **i-1** and **i+1**. 1st and the last process will have only **1** adjacent neighbor, whereas, all the other process will have **2** neighbor processes.
- Random 2D Gird (rand2D) – All processes are randomly positioned at x, y coordinates on a [0-1.0] x [0-1.0] square. Two processes are connected if they are within **.1** distance to other process. Processes in this topology will have random number of neighbor process depending upon.
- 3D Torus Grid (3Dtorus) – Processes are placed in a 3-dimensional cube and each process will have **6** neighbor processes. Processes on the outer surface are connected to other actors on opposite side.
- Honeycomb (honeycomb) – Processes are arranged in form of hexagons and two processes are connected if they are connected to each other. Each process will have maximum of **3** neighbor processes.
- Honeycomb with a random neighbor (randhoneycomb) – Process are arranged in form of hexagons. Neighbor processes are derived using Honeycomb Topology and an extra neighbor process is randomly chosen from the process in the network. Each process will have maximum of 4 neighbor processes.

GOSSIP PROTOCOL

- Implementation –
  - Each node has the following values in a tuple as its state –
    - **Rumor** – Message passed between processes in the topology

        - **Rumor Count** – Maximum count for each process is set to **20**. Once the process reaches maximum rumor count, it stops transmitting
        - **Stats PID** – Process id of a separate timekeeping process for maintaining start and end timer
        - **Failure Probability** – Required for failure model
    - Initially the main process will select a gossip process at random and transmit a rumor **"psst"** to the process.
    - Once a process receives a rumor, it will increment its rumor count by 1, i.e. rumorCount = rumorCount + 1.
    - Each process, after receiving a rumor for the first time, will start transmission loop which sends a message to its neighbor process, selected at random, every 10 milli seconds.
    - A process stops transmitting rumors to its neighbor processes, once the rumor count reaches maximum limit.
    - Timekeeping process reports back the total convergence time to the main process, once all the processes have reached maximum rumor count or pre-defined number of processes have reached maximum rumor count.
- Observations –
    - Xyz

PUSH-SUM PROTOCOL

- Implementation –
    - Each node has following values in a tuple as its state –
        - **Neighbors** – List of process ids of its neighbors decided as per the topology
        - **Sum** – It is initialized to index value of the process
        - **Weight** – It is initialized to 1
        - **Ratio** – It is the sum estimate which is (sum / weight)
        - **Round** – To keep track of number of rounds
        - **Stats PID** – Process id of a separate timekeeping process for maintaining start and end timer
        - **Running** – Flag to determine whether a process can transmit a message or has reached termination condition and hence cannot transmit or receive messages over the network
        - **Failure Probability** – Required for failure model.

  - Initially the main process will select a push-sum process at random and transmit a message with sum=0 and weight=0 to that process.
  - Once a process receives a message with sum=s and weight=w, it will update its sum and weight i.e. sum=state.sum + s and weight=state.weight + w.
  - A process will select a random process from its neighbor list and then transmit message, with sum=new_state.sum/2 and weight = new_state.weight/2, if the selected process is running.
  - A process terminates, i.e. state.running is updated to false, if its ratio i.e. sum estimate does not change more than 10^{-10} in 3 rounds
  - A process with no neighbors in its list of neighbors terminates and notifies the timekeeping process to converge all running processes in the network topology.
  - Timekeeping process will report the converge time for the network once every node in the network has converged.
- Observations –
  - Xyz