**PROJECT 2 – GOSSIP SIMULATOR**

**Group Members**

|  |  |  |
| --- | --- | --- |
| **Name** | **UF-ID** | **E-mail** |
| Aditya Bharadwaj | 9890-1939 | aditya.bharadwaj@ufl.edu |
| Shiridi Sai Prasad | 8073-3499 | shiridisaiprasad@ufl.edu |

**Problem Definition**

To implement the gossip and push-sum algorithm for the following topologies, and observe their behaviors.

* Full network
* Line network
* Random 2D network
* 3D Torus network
* Honeycomb network
* Random Honeycomb network

**Steps to Execute Code**

1. Open terminal at project location.
2. Run the command “**mix escript.build**”.
3. Run the command “**./topology NumberOfNodes Topology Algorithm**”

**Implementation Details**

**Gossip Algorithm**

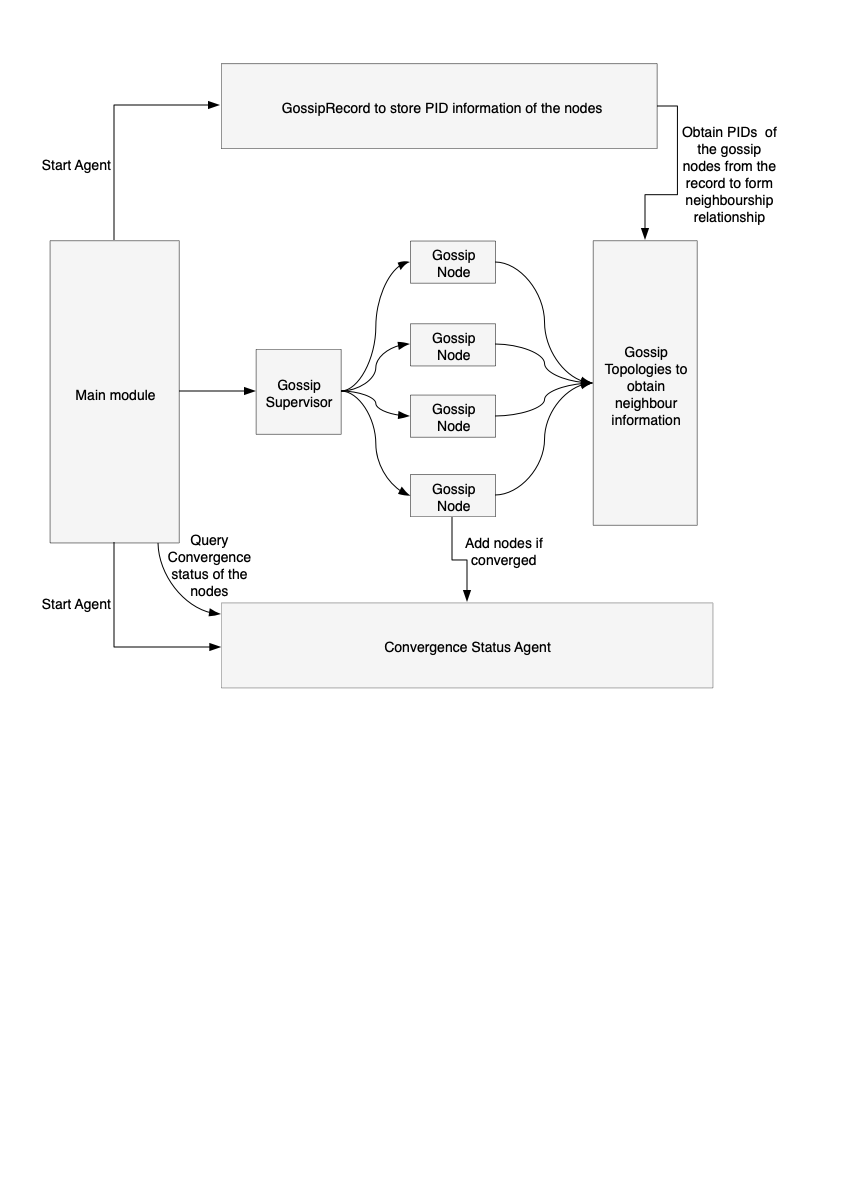
We create as many nodes as specified in the command line, each node being a Genserver actor in Elixir.

A rumor is sent manually to any one random node, and that rumor is spread across all nodes in the network.

The manner in which the information is propagated depends on the topology chosen by the user.

A node “converges” if it receives the rumor 10 times.

With respect to the code design, please find the flowchart below:



**Push-Sum Algorithm**

State: Each actor Ai maintains two quantities: s and w. Initially, s = x = I and w = 1.

Starting: Ask one of the actors to start from the main process.

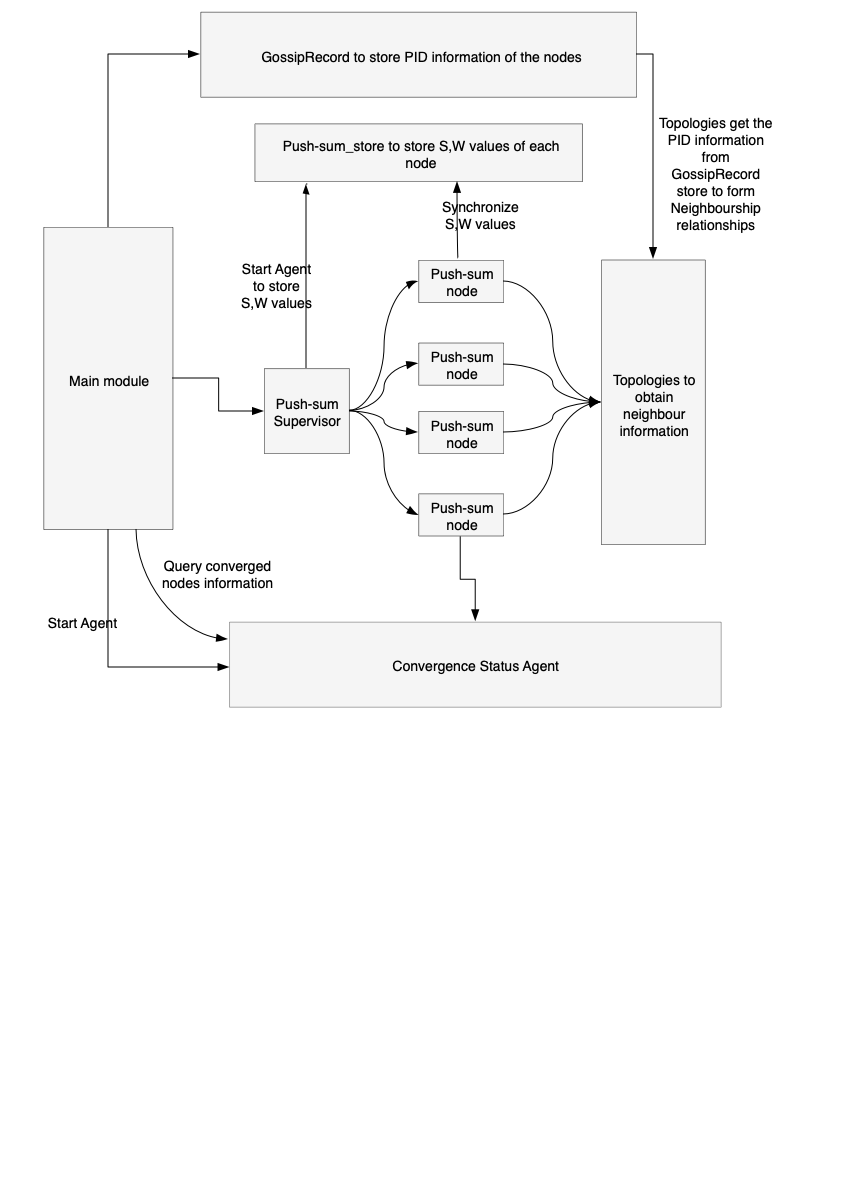
Receive: Messages sent and received are pairs of the form (s, w). Upon receive, an actor should add received pair to its own corresponding values. Upon receive, each actor selects a random neighbor and sends it a message.

Send: When sending a message to another actor, half of s and w is kept by the sending actor and half is placed in the message.

Sum estimate: At any given moment of time, the sum estimate is s/w where s and w are the current values of an actor.

Termination: If an actor ratio s/w did not change more than 10-10 in 3 consecutive rounds the actor terminates.

With respect to the code design, please find the below flowchart



**Termination Condition**

**Gossip Algorithm:**

The termination condition is reached when all the nodes have converged, i.e when each node has received the message 10 times.

When a node has run out of neighbors, it is terminated.

**Push-sum Algorithm:**

The termination condition is reached when for a node, the s/w ratio does not change by 0.0000000001 for 3 consecutive times. When a node runs out of neighbors, it is killed.

**Calculation of execution time**

We start the timer after all the nodes have been created and just before the first message is sent manually to a random node.

The timer ends when convergence or end condition is obtained.

**Data and Execution Screenshots**

While experimenting with the time taken to converge for the different topologies, we documented our findings as shown in the excel file attached.

**Inferences:**

1. We observed that the time taken for convergence by line topology is the worst compared to the other topologies over the range of 100- 2000 node networks.
2. We have observed that for rand2D network topology, the convergence rate is better when the number of nodes increase. In case of small number nodes (for e.g, 10) there are more isolated nodes resulting in less convergence.
3. We observed that in case of structured topologies, (honeycomb, 3Dtorus, randhoneycomb, rand2D) the rate of change in the convergence time is significantly low compared to the other topologies.
4. In the case of random 2D topology, 100% of the nodes start converging with highest probability when the number of nodes is 257 and 294 for the gossip and push-sum algorithms respectively. This is because the probability of the network being fully connected, without any clusters of disconnected networks increases when more number of nodes are added to the network.
5. In case of asynchronous push-sum algorithm, where a node asynchronously selects a random node, and sends the [s/2 , w/2] values after receiving the first message, we have observed that introducing delays in the message propagation results in better convergence rate.
6. We have observed that the performance of the line topology is the worst by far for both gossip and push-sum.
7. The structured topologies mentioned in point 3 have better performance the full network.