COP5615 - Project 2

Gossip and Push-sum Simulation

1. Group Members

Prajwal Dondiganahalli Prakash (UFID 04464906)
Rishabh Khanna (UFID 13399251)

2. Steps to run:

1. Unzip khanna_dondiaganahalliprakash.zip

2. Change directory to the project folder in the terminal.

3. You can run the script as:

```
$ mix escript.build
$ ./my_program
Usage: mix run gossip_simulator.exs num_nodes topology algorithm

Available topologies:
- full
- line
- rand2D
- 3Dtorus
- honeycomb
- randhoneycomb

Available algorithms:
- gossip
- push-sum
```

or

```
$ mix run gossip_simulator.exs
Usage: mix run gossip_simulator.exs num_nodes topology algorithm

Available topologies:
- full
- line
- rand2D
- 3Dtorus
- honeycomb
- randhoneycomb
```

```
Available algorithms:
- gossip
- push-sum
```

3. Description

The project is divided into two parts. The first part is the implementation of two algorithms - gossip and push-sum. The second part deals with topologies used to connect different workers/nodes. The six topologies - Line, Full, Random 2D, 3D Torus, Honeycomb, Random Honeycomb, have been implemented.

Time Measurement - Start time is marked using System.system_time function and is used before the algorithm starts executing (after the network is set up). Once the algorithm finishes, the end time is marked using System.system_time function. The difference of these times is used as the time elapsed for the algorithm.

3.1 Gossip

- A random node is selected among all the nodes. A message is sent to the selected node.
- When a node receives a message, it transmits the message to a random neighbour every 20ms. This ensures that there is more than one instance of the message flowing around in the distributed system.
- A node stops transmitting messages to its neighbour once it has received the message 10 times. This is achieved by keeping a counter for each node (in the GenServer state).
- Once 90% of the nodes have received the message at least once, the convergence condition is achieved and the program terminates.

3.2 Push-Sum

- S values for all the nodes are set during initialization. A random node is selected among all the nodes. Push-sum starts with the selected node (using its s and w values).
- The node then sends half of its s and w values to a random neighbour.
- A node is marked as terminated once 3 consecutive s/w ratios don't change greater than $1/10^{-3}$.
- Once 90% of the nodes have been marked as terminated the algorithm stops.

Note: The number of nodes taken as input is scaled up to the nearest perfect square for the *Honeycomb* and *Random Honeycomb* topologies and to the nearest perfect cube for *3D Torus topology*.

4. Results

The following results were observed:

# Nodes	Line		Full		Random 2D	
	Gossip (ms)	Push-sum (ms)	Gossip (ms)	Push-sum (ms)	Gossip (ms)	Push-sum (ms)
50	188	1766	0	46	converges till 4%	converges till 10%
100	469	12312	0	219	converges till 11%	17000
200	500	104047	47	735	converges till 31%	13078
300	844	321281	16469	1938	62	8344
400	1375	780954	67813	3437	47	9922
500	1500	1421171	177328	5140	31	22984
1000	1843	converges till 70%	1375000	19469	125	152422
3000	8875	timeout	timeout	172235	1250123	410750
5000	346672	timeout	timeout	52837	timeout	timeout

# Nodes	3D Torus	
	Gossip (ms)	Push-sum (ms)
50 scaled up to 64	16	110
100 scaled up to 125	31	125
200 scaled up to 216	62	609
300 scaled up to 343	62	1485
400 scaled up to 512	31	2594
500 scaled up to 512	47	2453
1000	47	7782
3000 scaled up to 3375	235250	56735
5000 scaled up to 5832	466188	134391

# Nodes	Honeycomb		Random Honeycomb	
	Gossip (ms)	Push-sum (ms)	Gossip (ms)	Push-sum (ms)
50 scaled up to 64	63	250	31	63
100	31	719	31	94
200 scaled up to 225	63	3266	31	266
300 scaled up to 324	63	7188	31	328
400	93	10578	31	422
500 scaled up to 529	94	17516	31	750
1000 scaled up to 1024	94	613660	47	1687
3000 scaled up to 3025	234	529750	78	6110
5000 scaled up to 5041	18109	1267561	219157	10719

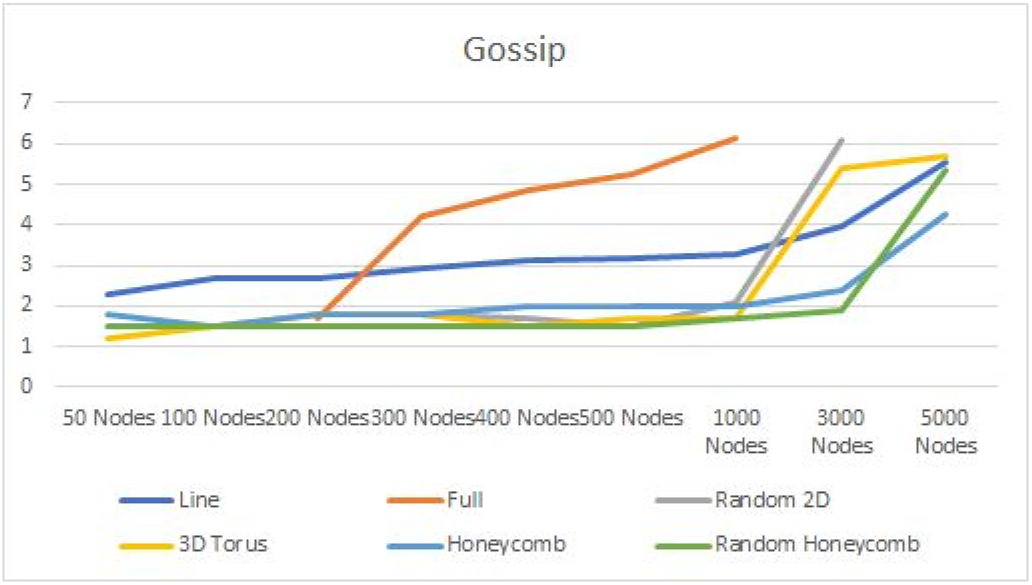
5. Observations

5.1 Gossip

- **Line -** There is an increasing trend, however, we observe that there is a sudden increase in time between 3000 and 5000 nodes.
- **Full** - For less number of nodes, *full* takes less time but there is a sudden increase from 200 to 300 nodes. The time taken by *full* is a lot more than *line* topology. However, for 3000 and 5000 nodes, the algorithm times out.
- **Random 2D** - For less than 300 nodes, the algorithm has a hard time achieving 90% convergence rate. As observed with increase in nodes the convergence increases. The timing of *random 2D* is better than *line* and *full* topology. But for higher number of nodes like 3000, the time consumed is very high. It times out for 5000 nodes.
- **3D Torus -** For below 3000 nodes, the algorithm shows very promising results. For 3000 and above nodes, the time increases but still shows better performance than *full* and *random 2D* topology. *Line* topology shows better performance for this number of nodes.
- **Honeycomb -** This topology shows results which are better than all the other topology But, for 3000 and 5000 nodes, the time increases suddenly. But the time taken by the other topologies is much greater than the time taken by *Honeycomb*.

- **Random Honeycomb** - This is an improvement to the *honeycomb* as the results show (though the time taken by 5000 nodes is more than the time taken by honeycomb with 5000 nodes).

As the following graph (of number of nodes against log(time)) suggests, the best topology for gossip algorithm can be random honeycomb. *Honeycomb* and *3D Torus* can also be considered as they give similar results for less nodes.



5.2 Push Sum

- **Line** - Push-sum takes a lot of time to complete the algorithm. For nodes more than 1000, the algorithm struggles to converge and times out.

- **Full** - This topology has better results than *line* topology for the push-sum algorithm. But the results for nodes 3000 and 5000 are unexplainable (for 5000 nodes, the time decreases as compared to 3000 which was not the trend with starting data points).

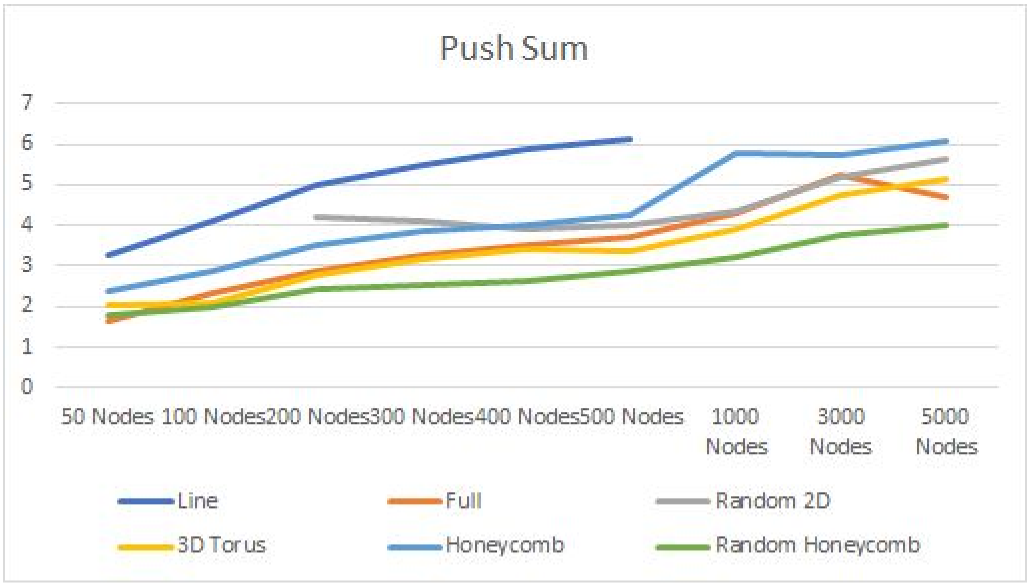
- **Random 2D** - Showing random behaviour, runs for less number of nodes have a hard time converging. With 50 nodes, the algorithm doesn’t terminate because of randomness resulting in nodes being very far from each other. As the number of nodes increases, the algorithm starts converging but the time does not show an increasing or decreasing trend. Infact, it best performs with 300 nodes.

- **3D Torus** - Showing similar performance as in gossip, *3D torus* shows promising results for push-sum algorithm also. With an increasing trend, the algorithm converges for all the nodes and takes less time for larger number of nodes.

- **Honeycomb** - The timing of this topology are not as promising as *3D Torus* but it converges for all the runs in good time.

- **Random Honeycomb** - This topology showcases the best convergence times among all the topologies and is an improvement over honeycomb.

By plotting the following chart between number of nodes and log(time), it can be observed that for push-sum, the best topology is *Random Honeycomb* as this topology gives the best results throughout. For lower number of nodes *Honeycomb*, *3D Torus* and *Random Honeycomb* all behave similarly.


Push Sum
50 Nodes
100 Nodes
200 Nodes
300 Nodes
400 Nodes
500 Nodes
1000 Nodes
3000 Nodes
5000 Nodes
Line
Full
Random 2D
3D Torus
Honeycomb
Random Honeycomb