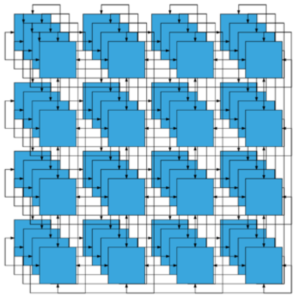
Project Report: Gossip Simulator

Group Members

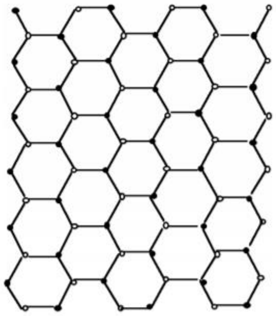
- Kalpak Seal – 8241 – 7219
- Sagnik Ghosh – 3363 – 6044

Topologies

- Line: All the actors are placed in a linear order forming a line, where an actor at position 'i' has neighbors at positions i+1 and i-1. Maximum neighbors possible are 2 and minimum is 1.
- Full: Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.
- Random 2D: Actors are randomly position at (x, y) coordinates on a [0 - 1.0] x [0 - 1.0] square. Two actors are connected if they are within 0.1 distance to other actors. This is an interesting case which have lot of degree of randomness here with unsure maximum and minimum neighbors.
- Torus 3D Grid: Actors form a 3D grid. The actors can only talk to the grid neighbors. And, the actors on outer surface are connected to other actors on opposite side, such that degree of each actor is 6. Communication can take place in 6 directions, +x, −x, +y, −y, +z, −z. Each edge of 3D Torus consist of n nodes. The total nodes of 3D Torus is n^3



- Honeycomb: Actors are arranged in form of hexagons. Two actors are connected if they are connected to each other. Each actor has maximum degree 3.



- Honeycomb with a random neighbor: Actors are arranged in form of hexagons (Similar to the Honeycomb topology). The only difference is that every node has one extra connection to a random node in the entire network.

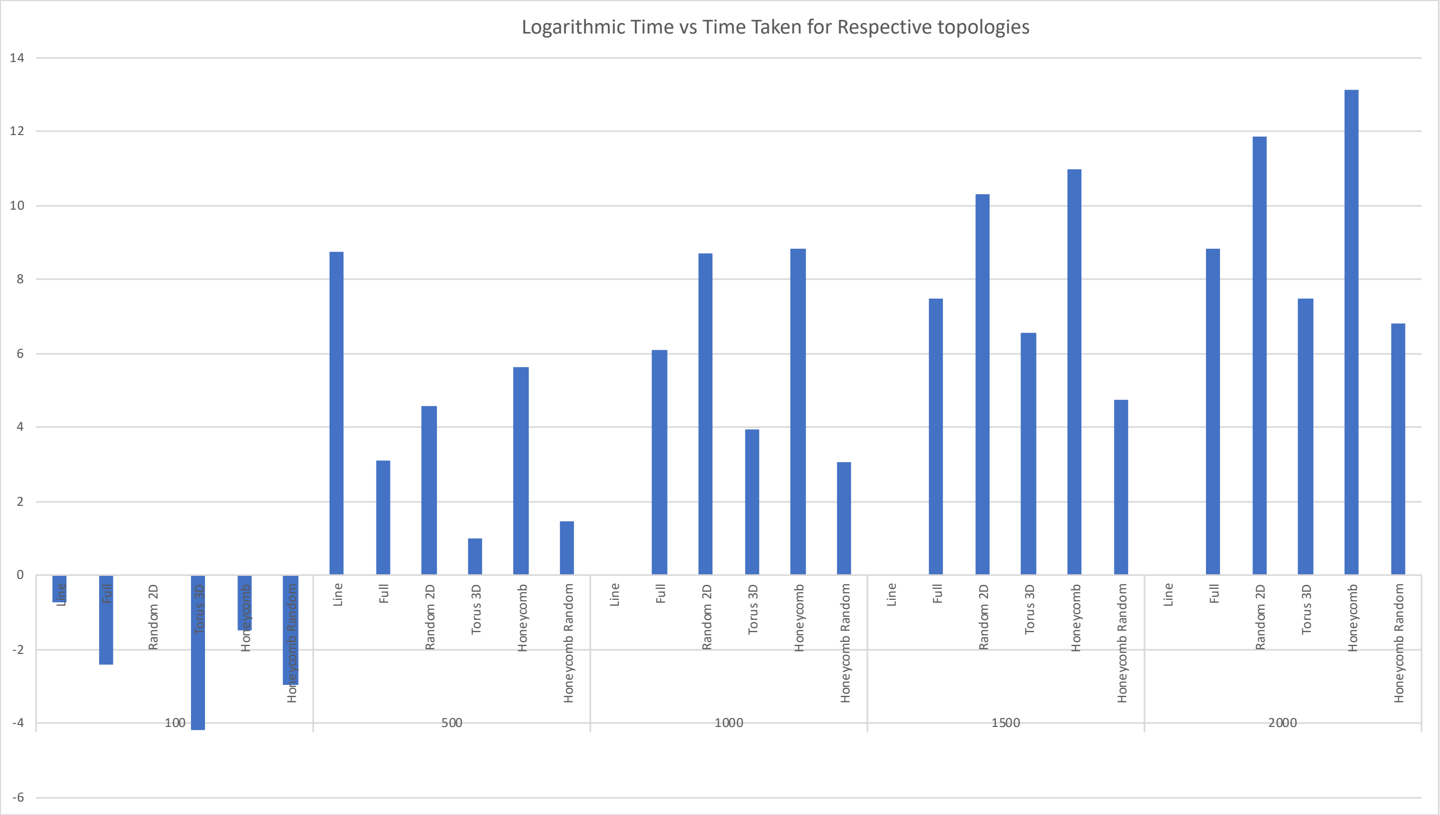
Gossip Protocol

Implementation:

- Starter module invokes the gossip protocol.
- On receiving the control, a random node sends all of its neighbors the message, and this way, the message keeps spreading.
- Each node keeps track of the number of times it has received the message. If this count goes beyond 10:
    - It stops propagating this message to its neighbors.
    - It tells the Master module to add to the list of **converged nodes**.
- Once the Master module finds out that at least 90% of nodes has converged, it prints the total time it took and terminates the application.

Node Count	Topology	Time seconds	Log Time
100	Line	0.599	-0.739372092
	Full	0.187	-2.418889825
	Random 2D	NIL	
	Torus 3D	0.055	-4.184424571
	Honeycomb	0.35	-1.514573173
	Honeycomb Random	0.126	-2.988504361
500	Line	431.302	8.752554596
	Full	8.537	3.093729179
	Random 2D	23.821	4.574162073
	Torus 3D	2.016	1.011495639
	Honeycomb	48.606	5.603062508
	Honeycomb Random	2.739	1.453649266
1000	Line	NIL	
	Full	67.249	6.07144091
	Random 2D	416.009	8.70047093
	Torus 3D	15.515	3.955591792
	Honeycomb	453.062	8.823564681
	Honeycomb Random	8.34	3.060047384
1500	Line	NIL	
	Full	179.884	7.49092306
	Random 2D	1243.539	10.28023604
	Torus 3D	93.049	6.539918741
	Honeycomb	2009.11	10.97234084
	Honeycomb Random	26.538	4.729987743
2000	Line	NIL	
	Full	453.626	8.825359522
	Random 2D	3675.867	11.84386885
	Torus 3D	179.736	7.48973559
	Honeycomb	8879.361	13.11624014
	Honeycomb Random	112.736	6.816804475

The 'NIL' values denotes that we could not make the topologies converge and get an output. The reason being Gossip, as a protocol, performs best for the Full topology where all the nodes are interconnected to each other.



We can see that time taken to converge increases with the increase in the number of node counts and the 3Dtorus topology performs best out of all the topologies. This is expected as the topology as the highest degree of freedom, i.e. it gossips with 6 other nodes in the network topology.

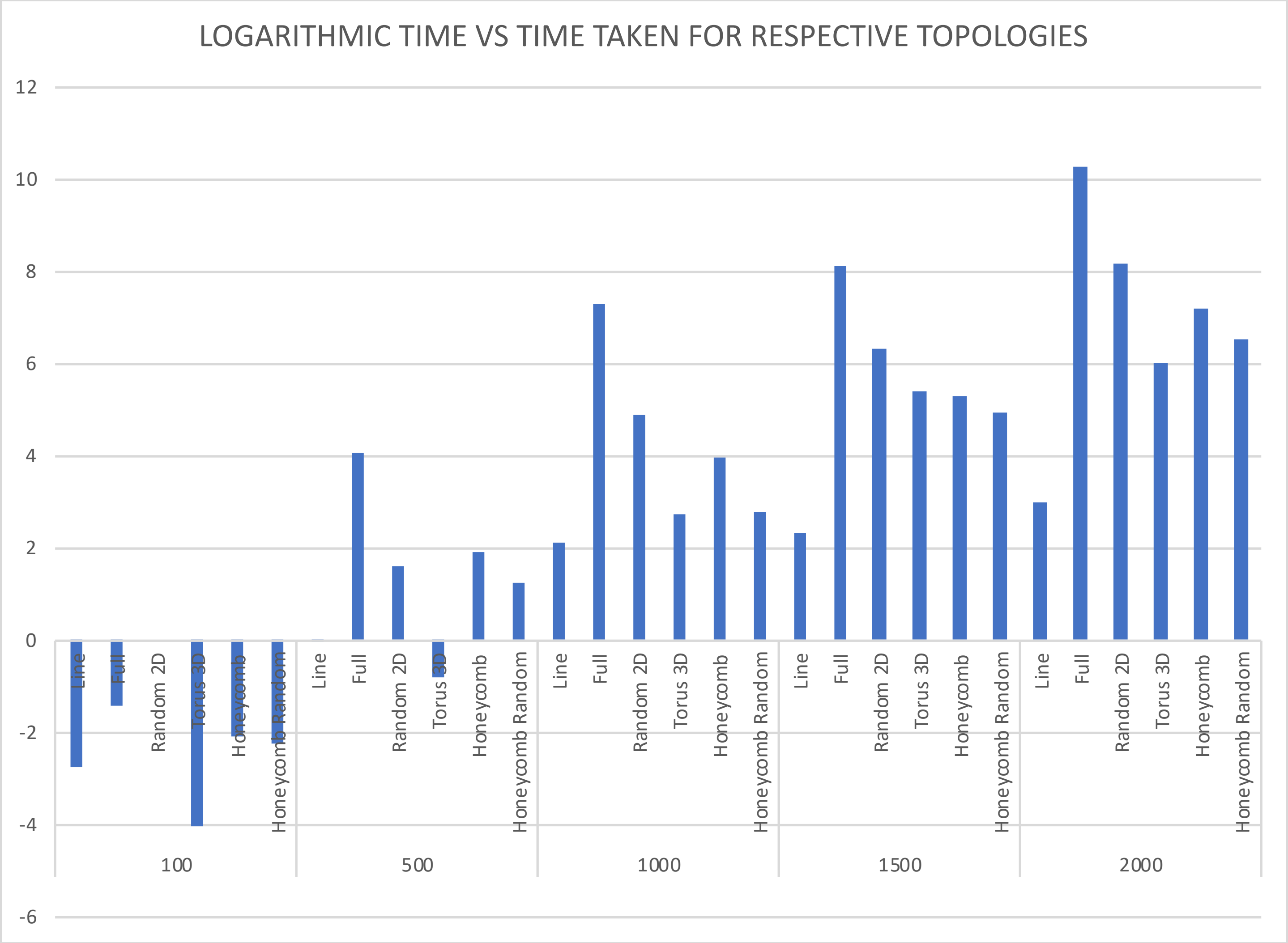
PushSum Protocol

Implementation:

- Each node has three values in a tuple as its state. These values are s, w and old_count. ‘s’ and ‘w’ are same as defined in the problem statement and old_count is the number of times the ratio ‘s’ and ‘w’ has not changed by more than pow (10, -10).
- Starter module initiates by selecting random node and tells it a message.
- On receiving a message of the form {s_half, w_half}, a node adds this tuple to its state values ‘s’ and ‘w’ and sends half of the new state values to a randomly selected neighbour.
- Before sending, a node updates its ratio and changes the ‘old_count’, if necessary.
- Once the ‘old_count’ has reached the value 3, it tells the Master module that it’s ratio (sum estimate) has converged. This node shall not transmit any longer.
- Once every node has converged, the driver function prints the total time it took to converge and terminates.

Node Count	Topology	Time seconds	Log Time
100	Line	0.145	-2.785875195
	Full	0.369	-1.438307279
	Random 2D	1	0
	Torus 3D	0.06	-4.058893689
	Honeycomb	0.232	-2.10780329
	Honeycomb Random	0.213	-2.231074664
500	Line	0.965	-0.051399153
	Full	16.402	4.035799837
	Random 2D	3.063	1.614945367
	Torus 3D	0.563	-0.828793173
	Honeycomb	3.717	1.894138688
	Honeycomb Random	2.387	1.255198566
1000	Line	4.225	2.078951341
	Full	153.806	7.264967974
	Random 2D	29.648	4.889862881
	Torus 3D	6.574	2.716771456
	Honeycomb	15.377	3.942702161
	Honeycomb Random	6.947	2.796390097
1500	Line	4.967	2.312374747
	Full	276.445	8.110848668
	Random 2D	79.761	6.317611592
	Torus 3D	42.579	5.412070162
	Honeycomb	38.938	5.283106879
	Honeycomb Random	30.533	4.932297442
2000	Line	7.952	2.991317757
	Full	1235.089	10.27039929
	Random 2D	285.025	8.154944656
	Torus 3D	64.664	6.014890848
	Honeycomb	146.412	7.193889992
	Honeycomb Random	90.122	6.493807425

The ‘NIL’ values denote that we could not make the topologies converge and get an output. When there are 100 nodes in the randon2D topology, there may be some nodes who won’t get neighbors assigned and therefore the nodes may not get messages so that they converge.



Torus3D and line has the best performance, time taken fairly constant with increment of nodes. Therefore, it converges faster than the other algorithms, the other algorithm which performs fairly good is Honeycomb with a random neighbor. These two are the best topologies for push-sum.

Observation

Doing comparative study of the two algorithms, we can see for Gossip algorithm, full and 3D torus has best performance. This is intuitive from the algorithm, as more the degree of nodes, the more gossip it can do. From this, we can conclude that 3D Torus and Honeycomb with a random neighbor is more stable topologies as the performance is fair for both the gossip and push-sum protocol.