COP5615 – Distributed Operating Systems

Project Report – 2

Team Members

1. Akshay Ganapathy (UFID - 3684-6922)
2. Kamal Sai Raj Kuncha (UFID - 4854-8114)

Input

The input to the program is the number of nodes (which is the number of actors involved), the topology to be used (“line” or “full” or “3D” or “imp3D”) and the algorithm to be used (“gossip” or “part-sum”)

Output

The output is the amount of time it took to achieve convergence of the algorithm.

Zip File Contents

The zip file consists of the readme.md file, Project_Report.pdf file, and the project2.fsx file which contains the code to be run.

Topologies

The following topologies are used:

Full: Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.

Line: Actors are arranged in a line. Each actor has only 2 neighbors (one left and one right unless you are the first or last actor).

3D Grid: Actors form a 3D grid. The actors can only talk to the grid neighbors which ranges from 3 to 6 depending on the position.

Imperfect 3D Grid: It has the same grid arrangement as 3D Grid, but one extra random neighbor is selected from the list of all actors.

How To Run

Run the project2.fsx file using the command:
“dotnet fsi project2.fsx <numNodes> <topology> <algorithm>”

where ‘numNodes’ is the number of actors used,
‘topology’ is either of “line” or “full” or “3D” or “imp3D”, and
‘algorithm’ is either of “gossip” or “push-sum”

Platforms used for running the code

Visual Studio Code
.NET version 5.0
NuGET Akka.NET v1.4.25

The following are the observations:

Gossip Protocol

The largest network we were able to run was **10000** nodes for each topology. The Gossip protocol execution time order obtained for different topologies is as follows:

Line Topology > Full Topology > 3D Topology > Imperfect 3D Topology

1. **Line Topology**: When the gossip algorithm is run with line topology, as the number of nodes increases, it does not converge. It converges the slowest among full topology, 3D topology and Imperfect 3D topology.

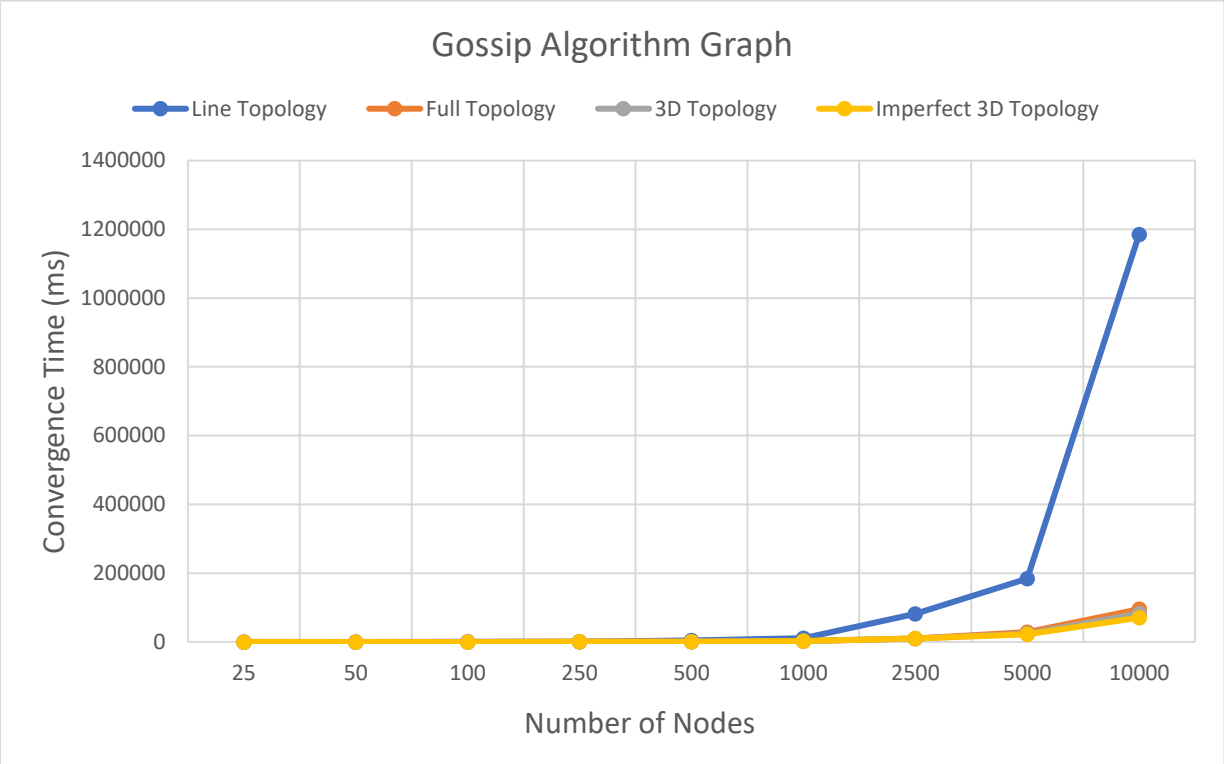
2. **Full Topology**: Full topology converges in most scenarios, even for high number of nodes because every node is connected to every other node. Full topology converges faster than line topology.

3. **3D Topology**: In 3D topology, each node has neighbors ranging from 3-6 nodes, and converges in most cases because each node can transmit the message to one of the 3-6 possible neighbors , unlike line topology where a node has only two neighbors. 3D topology performs better than full topology in terms of time convergence since in 3D topology nodes are arranged more symmetrically and neighbors of nodes are arranged around the node. Also, flooding of messages happens in full topology which introduces a delay, which does not happen in 3D Topology.

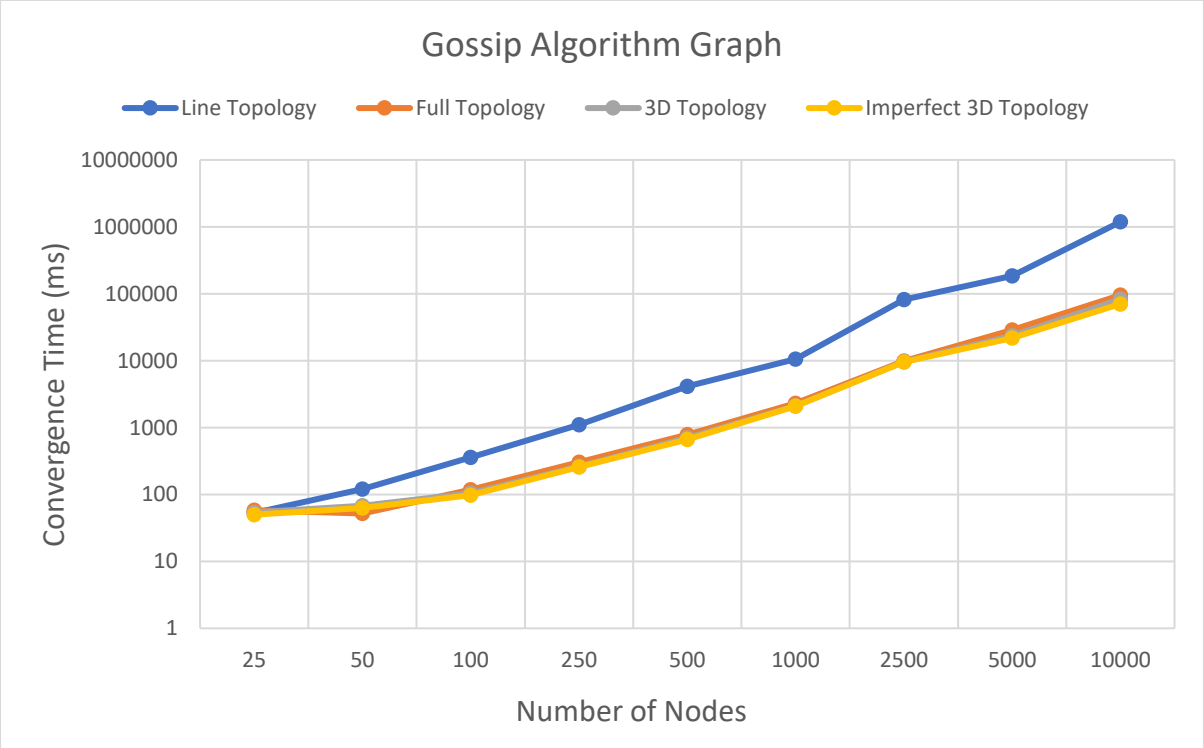
4. **Imperfect 3D Topology**: Imperfect 3D topology functions in the same manner as 3D topology. Since imperfect 3D topology has got an extra neighbor, it can transmit the message faster compared to 3D topology. It is observed that imperfect 3D topology is the fastest in terms of convergence among line topology, full topology, and 3D topology.

Interesting Observations: Since the message is started by a random node and on each reception of the message the node selects another node at random from its neighbors, there is a high probability that if the randomly chosen neighbor does not change until the node is transmitting messages, then the other neighbor does not receive the message and in cluster if this happens with many nodes, it results into blind spots, which have to be removed in order to achieve convergence in line topology. Hence, it is necessary that the nodes keep transmitting messages to the neighboring nodes so that each node has received the message at least once.

Linear Scale:



Log Scale:



Execution table:

	Line Topology	Full Topology	3D Topology	Imperfect 3D Topology
Number of Nodes	**Convergence Time (ms)**	**Convergence Time (ms)**	**Convergence Time (ms)**	**Convergence Time (ms)**
25	53	58	54	50
50	121	52	68	63
100	360	118	104	98
250	1102	304	263	257
500	4138	784	703	665
1000	10557	2316	2101	2093
2500	81986	9828	9647	9601
5000	184390	28799	23510	21897
10000	1185368	95270	82728	70458

Push Sum Algorithm

The largest network we were able to run was **500** nodes for each topology. The Push-Sum algorithm execution time order obtained for different topologies is as follows:

3D Topology > Imperfect 3D Topology > Line Topology > Full Topology

1. **Line Topology**: When the push-sum algorithm is run with line topology, it converges faster than 3D and Imperfect 3D topologies but is a bit slower than full topology. This has been considered with modifications made to the original form.

2. **Full Topology**: Full topology converges in most scenarios, even for high number of nodes because every node is connected to every other node. Full topology converges faster than other topologies.

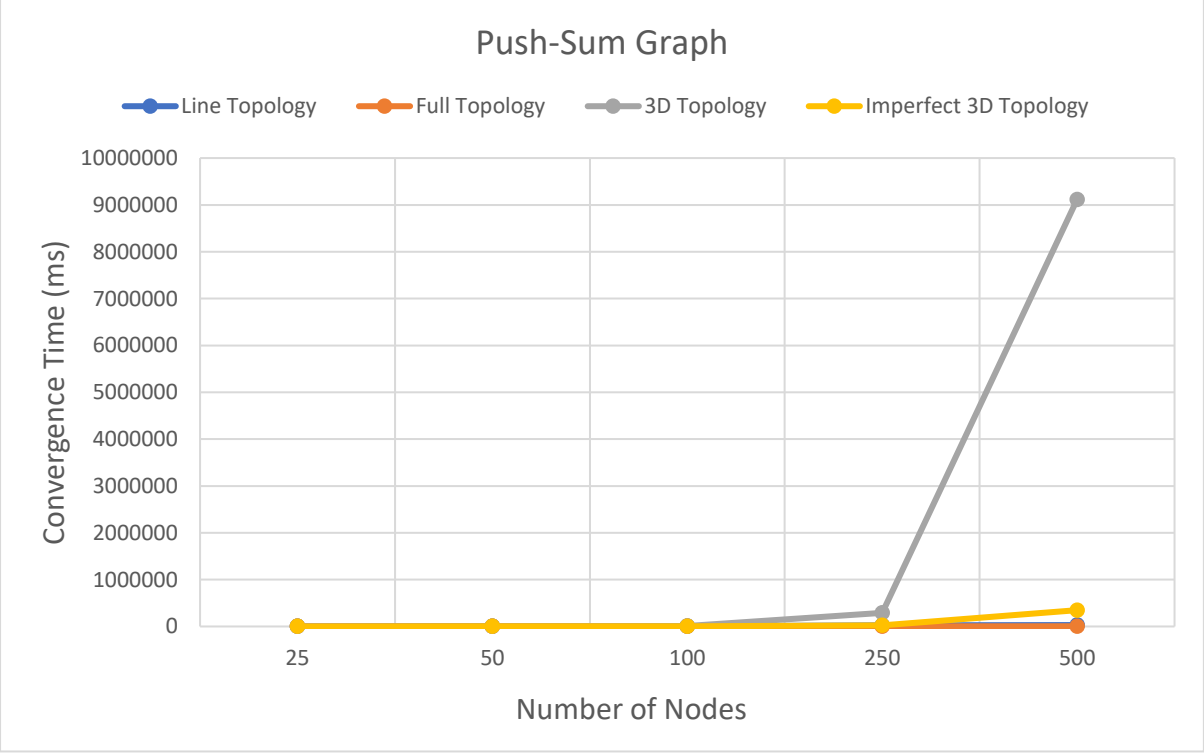
3. **3D Topology**: In 3D topology, each node has neighbors ranging from 3-6 nodes, and converges in most cases because each node can transmit the message to one of the 3-6 possible neighbors. This node arrangement helps the cluster to spread (s, w) value pair message in all directions of grid among all the nodes. In 3D topology, each node's neighbors are not consecutive. Therefore, the value of a node index differs among the neighboring nodes, which in turn affects the value of s. This disparity causes more delay in converging the s/w ratios for the given precision. Hence, 3D topology is the slowest.

4. **Imperfect 3D Topology**: Imperfect 3D topology functions in the same manner as 3D topology. In this case, the s/w ratio among the nodes converges faster than 3D topology but is slower than line and full topology.

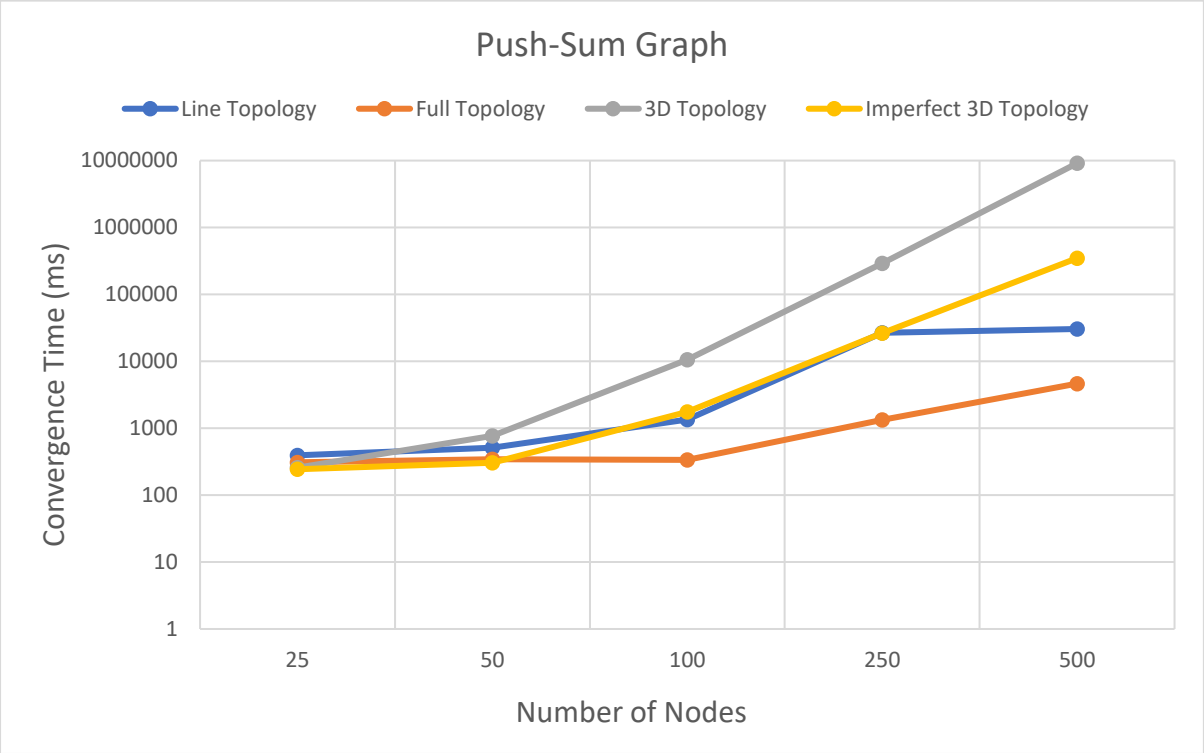
Interesting Observations: The same blind spot issue which occurred which we observed while running the gossip algorithm persists here as well. Hence, it is necessary that the nodes keep transmitting messages to the neighboring nodes so that each node has received the message at least once.

Given the condition for convergence is so minimal that when the topologies are run for even small clusters like (number of nodes = 50, 100), the program execution was running for hours. Hence, we chose different distribution of s value(s, w pair), so to achieve convergence for topologies in reasonable time, value of s to be node_id / 1000000000.

Linear Scale:



Log Scale:



Execution table:

	Line Topology	Full Topology	3D Topology	Imperfect 3D Topology
Number of Nodes	**Convergence Time (ms)**	**Convergence Time (ms)**	**Convergence Time (ms)**	**Convergence Time (ms)**
25	394	310	258	244
50	510	345	769	305
100	1350	336	10585	1757
250	26797	1336	291004	26502
500	30414	4645	9115386	348063