Distributed Systems

Gossip Simulator

READ ME file for Distributed Operating Systems - Project 2

Date: October 7, 2017

Group members:

1. **Mohit Mewara**, UFID: 8413-2114, mohitmewara@ufl.edu
2. **Nakshatra Sharma**, UFID: 4935-2902, nakshatra2312@ufl.edu

Project Execution Instructions:

For Gossip:

mix escript.build

./project2 'numnodes' 'topology' gossip

- Where numnodes represents the number of nodes in the topology
- Where topology represents the type of topology
  - line
  - full
  - 2D
  - imp2D

For Push-Sm

./project2 'numnodes' 'topology' push-sum

- Where numnodes represents the number of nodes in the topology
- Where topology represents the type of topology
  - line
  - full
  - 2D
  - imp2D

Implementation Details:

For Gossip algorithm convergence, we have assumed that the topology will be converged if 95% of the nodes have heard the rumor at least once. Once the algorithm converges the program is terminated and convergence time is printed on the screen. The neighbors are being selected at random so there might be a possibility that the network is not converged fully because some nodes stop sending the message once

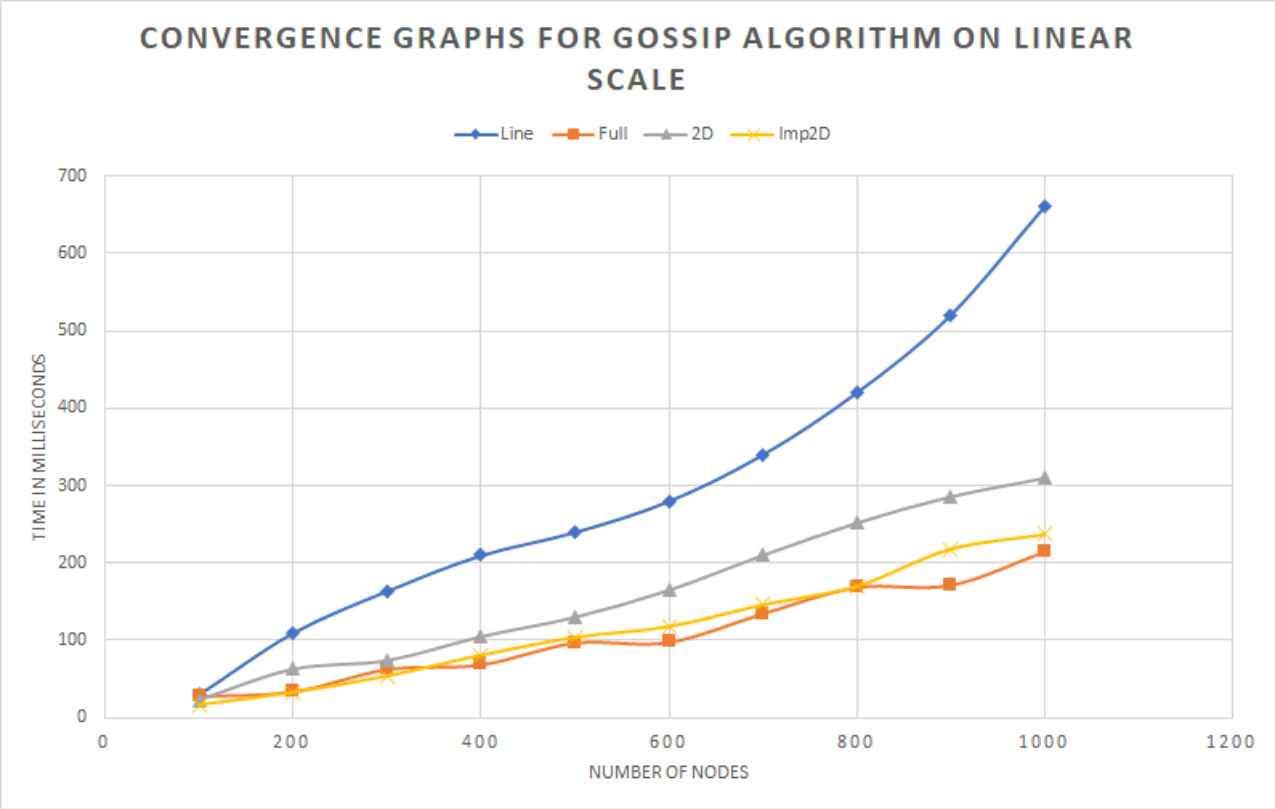
they have heard the rumor k times while their neighbors keep waiting for the message. In the current code we have used the value of k as 10, however we ran our code with different values of k to analyze how it would affect the convergence percentage.

For push-Sum algorithm, if the average estimate (s/w) of a node does not change more than 10^{-10} in three consecutive rounds, the node is assumed to be converged. We are making sure that the program terminates only when **all the nodes in the topology have been converged** so that each node maintains the same value of average estimate**.** If the program terminates after first node is converged the average estimate of the converged node will differ from the average estimate of the other nodes. So, we check that at least 95% of the nodes are converged to ensure that every node is maintain the same average. Our push-sum algorithm is spreading parallelly and not serially, meaning that every node will dissipate the sum in the node parallelly rather than serially, which dissipate only when it heard the message. In the parallel approach sum is dissipated much faster resulting in the fast convergence of the network.

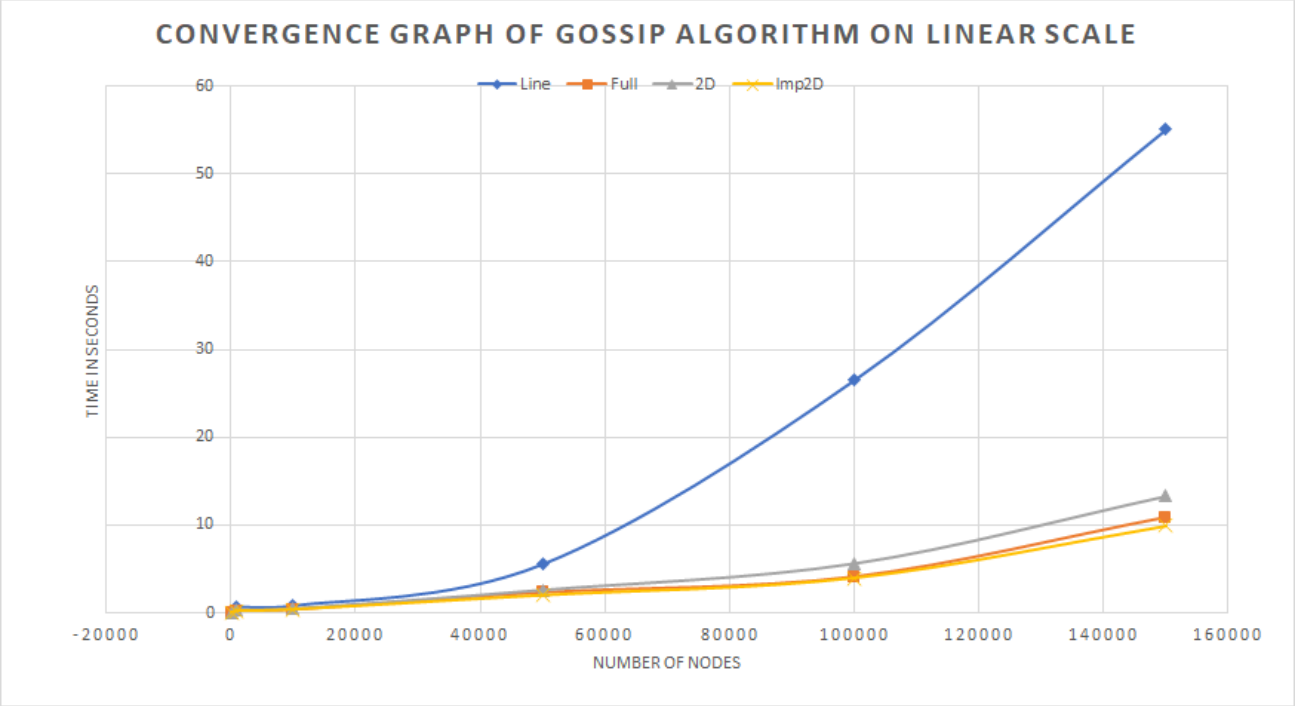
Graphs plotting convergence vs size of the network for different topologies and algorithms:

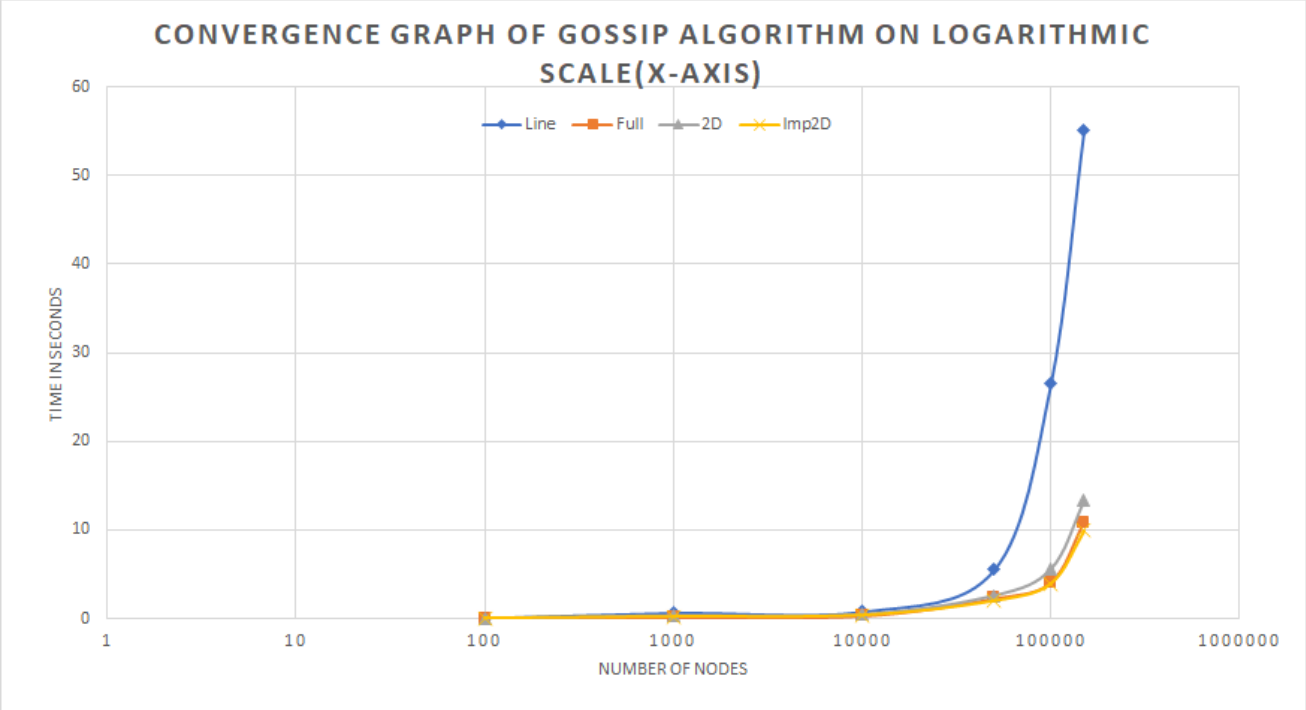
Gossip Algorithm:

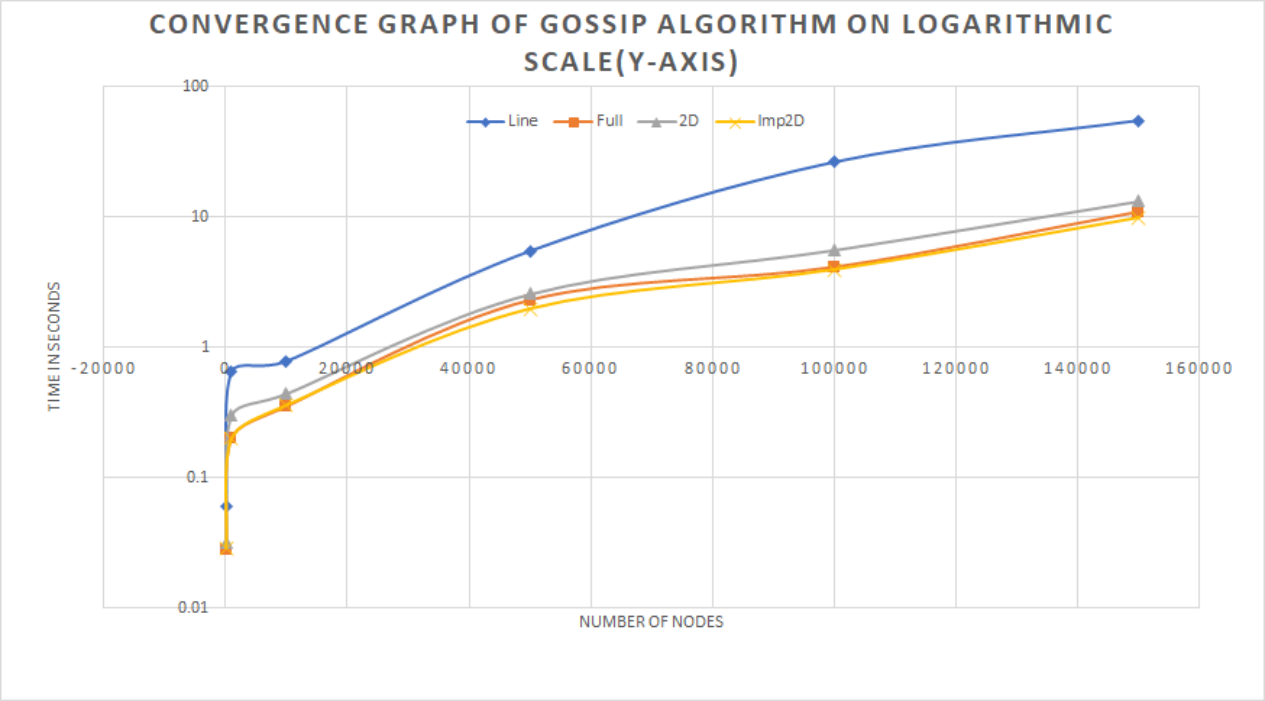
Small Network: We chose a small network (max 1000 nodes) to analyze the convergence time of gossip so that all four topologies gets converged easily in real time and can be analyzed on graph easily. Also, the time taken to converge varies every time so we have calculated average time to plot on the graph so that we get a better understanding of the convergence pattern.



Large Network: For large networks, all the network topologies get converged in real-time. However, line does not converge sometimes. Hence, we are taking the running average of all the topologies' convergence time.


CONVERGENCE GRAPH OF GOSSIP ALGORITHM ON LINEAR SCALE
Line
Full
2D
Imp2D
TIME IN SECONDS
NUMBER OF NODES
0
10
20
30
40
50
60
-20000
0
20000
40000
60000
80000
100000
120000
140000
160000


CONVERGENCE GRAPH OF GOSSIP ALGORITHM ON LOGARITHMIC SCALE(X-AXIS)
Line
Full
2D
Imp2D
TIME IN SECONDS
NUMBER OF NODES
0
10
20
30
40
50
60
1
10
100
1000
10000
100000
1000000



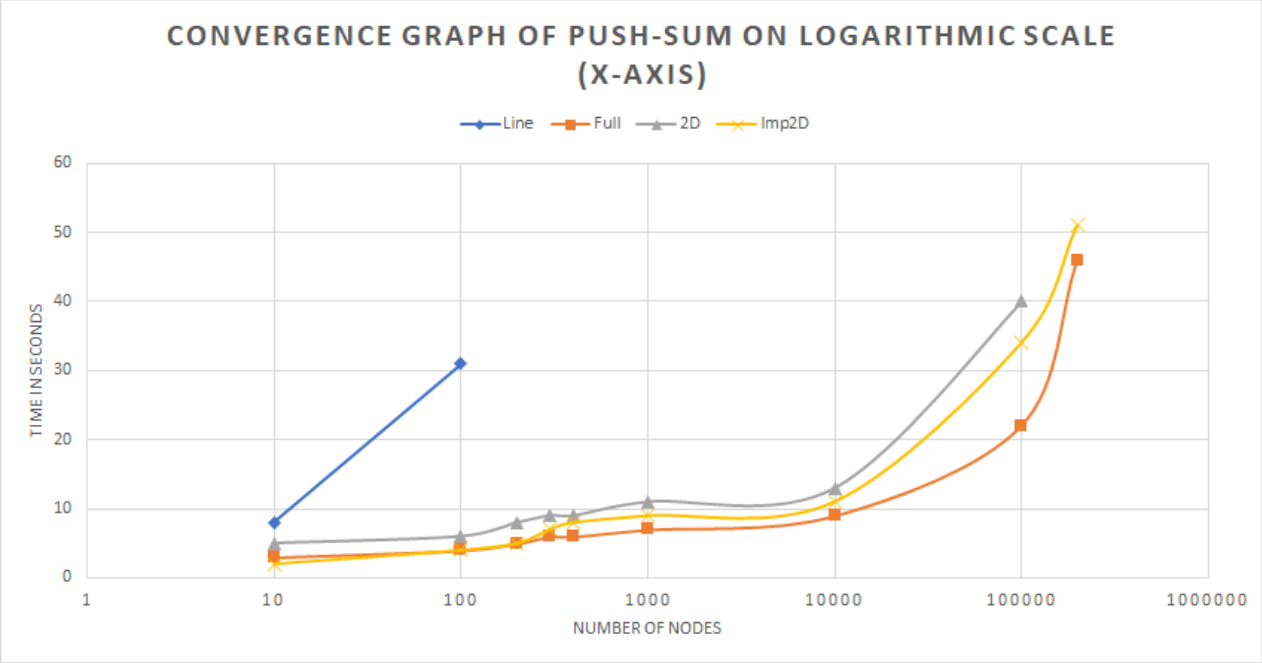
Push- Sum:

Push-sum takes a much more time in comparison with gossip. All the network topologies except line get converged in real-time. Most of the times, line converges but it takes a lot of time.
The graphs are plotted below:

The largest networks that we managed to converge at least once are as follows:

	Line	**Full**	**2D**	**Imp2D**
Gossip	100,000	250,000	150,000	200,000
Push-Sum	100	200,000	100,000	200,000

The maximum network topology that converge successfully was full of 250,000 nodes.



In push-sum the line takes a lot of time to converge for the node size of more than 100. So, we have limit the time for line topology to 100 nodes. Apart from line, all other nodes converged well with full taking the least time, followed by imp2D and then 2D topology.

Analysis:

- The converge time of gossip algorithm highly depends on the type of network topology and the number of nodes. For a smaller network (max 1000 nodes) the graph for line appears to be quadratic curve whereas the graphs for full, 2D and Imp2d appears to be logarithmic with full being the best followed by Imp2d and 2D.
- In large networks, the gossip algorithm follows the same trend with line being annoyingly slow and showing exponential curve. For large networks full and imp2d acts as similar in terms of convergence time followed by 2D.
- The push sum algorithm is an extension/application of gossip and hence follows the same trend as gossip in terms of convergence with topology. However, push-sum takes much longer to converge when compared to gossip since the sum must be reached from one end of the network to another end and each node must maintain the estimate average.

Interesting Observations:

- For line topology the value of k (number of times a node receives a message before stopping) greatly affects the convergence chances of the topology. When we chose k=10 for line, the network did not converge most of the times since some nodes after getting 10 messages stops

spreading the rumor to their neighbors. However, when we increase the value of k to 1000 we did not face such issue often.

- Ideally, the convergence time of full network should be lowest for full networks but IMP2d appeared to be as fast as full. This may because whenever a node selects a random far neighbor it creates a new front for gossip and hence the topology converges very fast.
- For push - sum if we assign the initial weight value of 1 to all the nodes, we will get the average value after the convergence. To receive the sum of all the nodes we initially assign the weight 0 to all the nodes and assigned the weight 1 to the node which is starting the push sum algorithm.
- For push-sum, we have observed that once a node converges fully the whole topology gets converged subsequently.

Failure Model / Bonus Implementation:

To implement the node failure model, we made use of Supervisor's "Supervise" functionality. The supervisor keeps track of all the children nodes and if any children node dies, it restarts the node instantly.

We tried to kill the nodes in between and they were restarted by the supervisor. Even if we don't include the Supervisor in our code and run the gossip and push sum algorithm we found that the nodes converge for full and imp2D. But, if we use supervisor then for full, 2D and Imp2D the convergence time is not greatly affected when a node dies.

However, for Line, if a node dies there were few instances when the topology stopped converging. The dying node could lead to a deadlock condition in the topology where the neighbors of the dead node are waiting for the first message to receive.