Group Members:

Amarjit Kumar Singh UF ID# 01946116 Email : amarnitdgp@ufl.edu
Sharique Hussain   UF ID# 71064371 Email : sharique88@ufl.edu

Files:
project2.scala
project2-bonus.scala – please read Report-Bonus.pdf for explaination.
build.sbt
Report.odt – xls sheet of experimental data
Report-Bonus.pdf
Pushsum.scp – script file while data collection for pushsum algo
gossip.scp – script file while data collection for gossip algo

Details:
The assignment uses Scala and Actor model to implement Gossip and PushSum Algorithm. This project has a Boss and Node Actor class. When executed, Boss builds a graph of actors using adjacency list for a given topology and triggers Rumor/PushSum. The adjacency list stores information about neighbors for each node.

Rumour:

When a node receives a rumor, it select its neighbor randomly, and pass the message to them. Once a node reaches maximum count of 10 for received rumours it terminates and notifies the boss. The boss has a count for terminated nodes and can tell the percentage of convergence
i.e. terminatedNodes*100/TotalNodes.

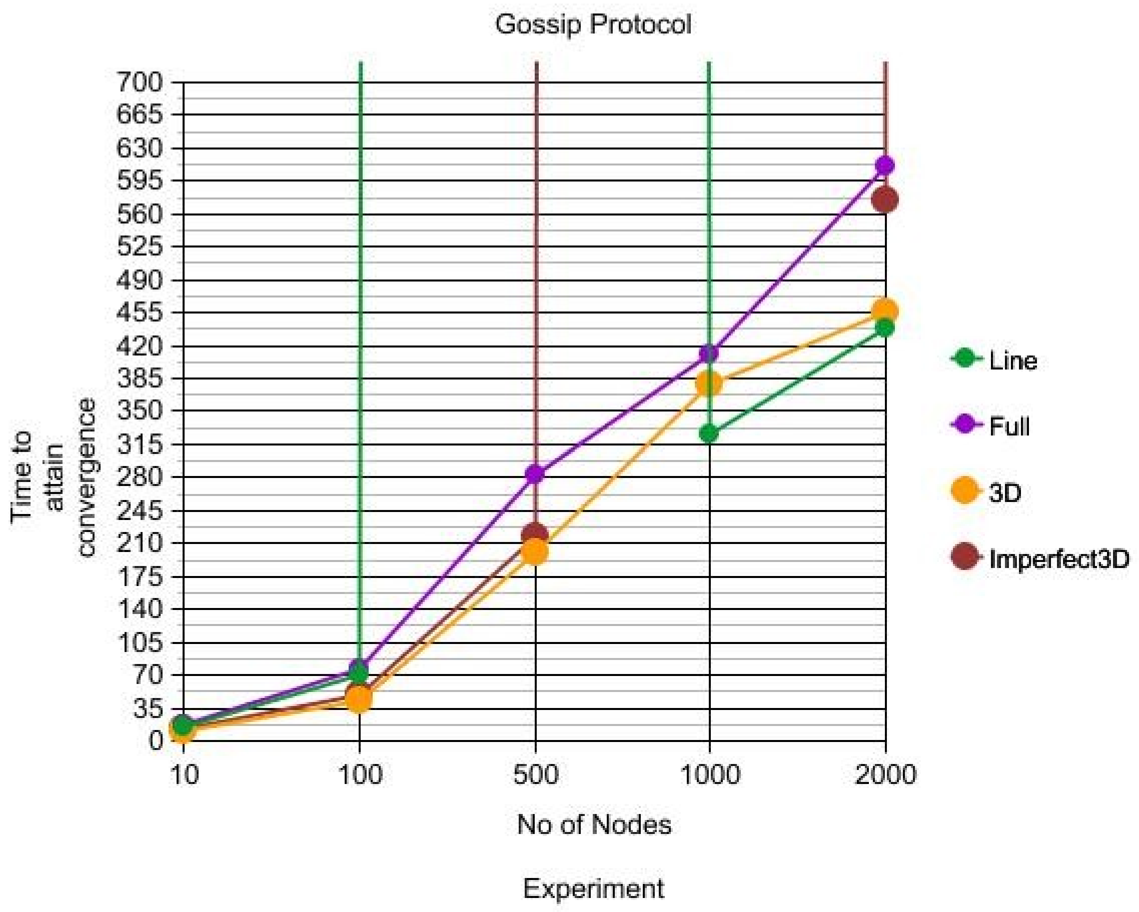
Findings:
<u>Line topology</u>
- Quite frequently it takes forever (it stucks) to converge to all nodes, especially for large network.
- Tried maximum with 500k nodes, and convergence reached to 62.96 % nodes and got stuck after that.

<u>Full topology</u>
- Convergence happened 100% up until 5000 network size, but OutofMemory errors started for larger networks due to high no. of messages.

<u>3d topology</u>
- This topology performed best and convergence happened 100% for even 500k network size in least amount of time (137.5 sec)

<u>Imperfect3d topology</u>
- Convergence happened for 99.98 % nodes most of the time (max checked 500k), so 3d topology was better.

Conclusion: ***Full is not as efficient due to OutOfMemory for large networks whereas line has infinite time limitation and take forever, 3d topology is fastest and best while imperfect3d topology can also get stuck but happens when few nodes are left to terminate.***



PushSum:

Boss starts by sending s,w → 0,0 (since there should be no contribution of sum & weight by Boss in the network) to a node in the middle. A node which receives message add's received sum and weight to its buffers, then divides by 2 and sends half to the other randomly selected node from its neighbor's list. When ratio stops changing much, as given in project2 pdf, it terminates.

Findings:

Line topology

- Again sometimes it stuck or takes forever, but even if it completes 100% the ratio wasn't uniform throughout all nodes. So this topology must not be followed.

Full topology

- Convergence happened 100% up until 5000 network size & the Ratio was uniform, but OutOfMemory errors started for larger networks due to high no. of messages .

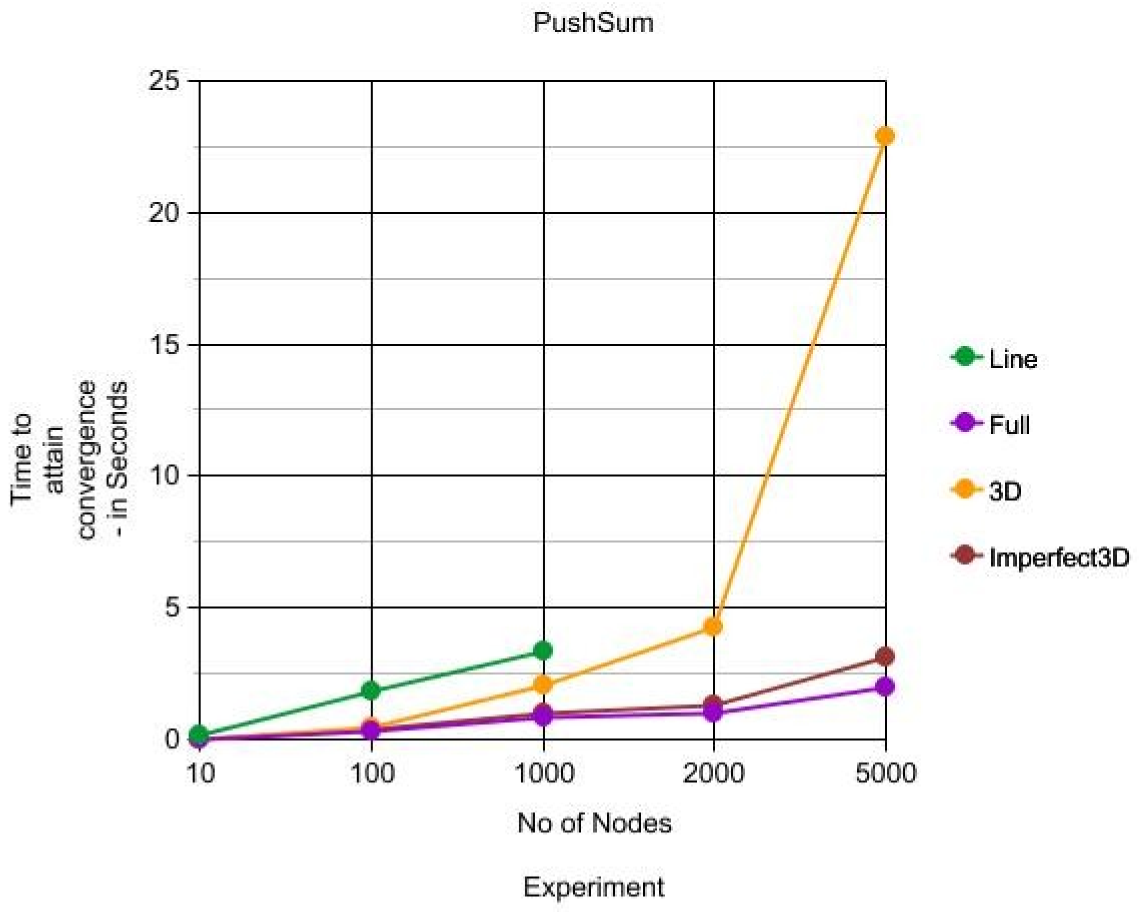
3d topology

- Convergence happened 100% and good thing is that the ratio was uniform in all nodes, but here 30k network size is the max we were able to run and got results for. And 100k was too large and it couldn't even start giving output for even one node.

Imperfect3d topology

- This topology performed best with uniform average throughout nodes and values were same as in 3d. This topology was very fast and program ran for even 500k network size.

Conclusion: ***Full is not useful due to OutOfMemory for large network, line is giving wrong result, imperfect3d is fastest and the best, 3d was very slow in comparison to imperfect3d.***



Imperfect3d algo performs best for large no. of nodes (checked with 500000). It took 525 seconds to complete.

Results were tested in Intel I3 machine having 4 cores.

Further work:
We could have done actor implemention on multiple machines, if we had more time. Also, we could checked for fault tolerance on comminucation failure scenarios over distributed systems.