Distributed Operating Systems- Project-2
Implementation of Gossip and Pushsum Algorithms

Group info:

Name: Namratha Reddy Tippireddy **UFID: 9151-9979**
Name: Megha Nagarmunoli **UFID: 6768-1778**

Aim:
The aim of this project is to implement the Gossip and Push-Sum Algorithms

Topologies Observed:

- **line**
- **full**
- **rand2D**
- **3Dtorus**
- **honeycomb**
- **randhoneycomb**

Implementation:
Gossip Implementation: First the main randomly selects an actor and sends it the rumor, then each actor selects another random actor and sends it the rumor. An actor becomes inactive once it receives a message more than 10 times. Every actor notifies it's neighbors that it is no longer and to remove it from their neighbour list. Every actor also notifies the Watcher actor that increments the termination count. Once a selected percentage of actors are inactive, the main is intimated by the Watcher and the algorithm terminates.

All the actors are implemented as GenServers.

Pushsum Implementation: In the pushsum algorithm, each actor is initialized with a set of s and w values, we are initializing S_i = x_i = i and w=1. Every time an actor receives a message, it adds the received pair to its own values and while sending it keeps half of s, w with itself and sends the other half to a randomly selected neighbor. If the difference of s/w does not change more than 10^{-10} in 3 consecutive rounds the actor terminates.

Similar to gossip, in pushsum, the actors are GenServer and they communicate with each other using call and cast functions. Once the given percentage of actors terminate, or they no longer have a node to communicate with, the algorithm converges.

Execution:

- Extract Tippireddy-Nagarmunoli.zip
- Enter proj2
- Commands:

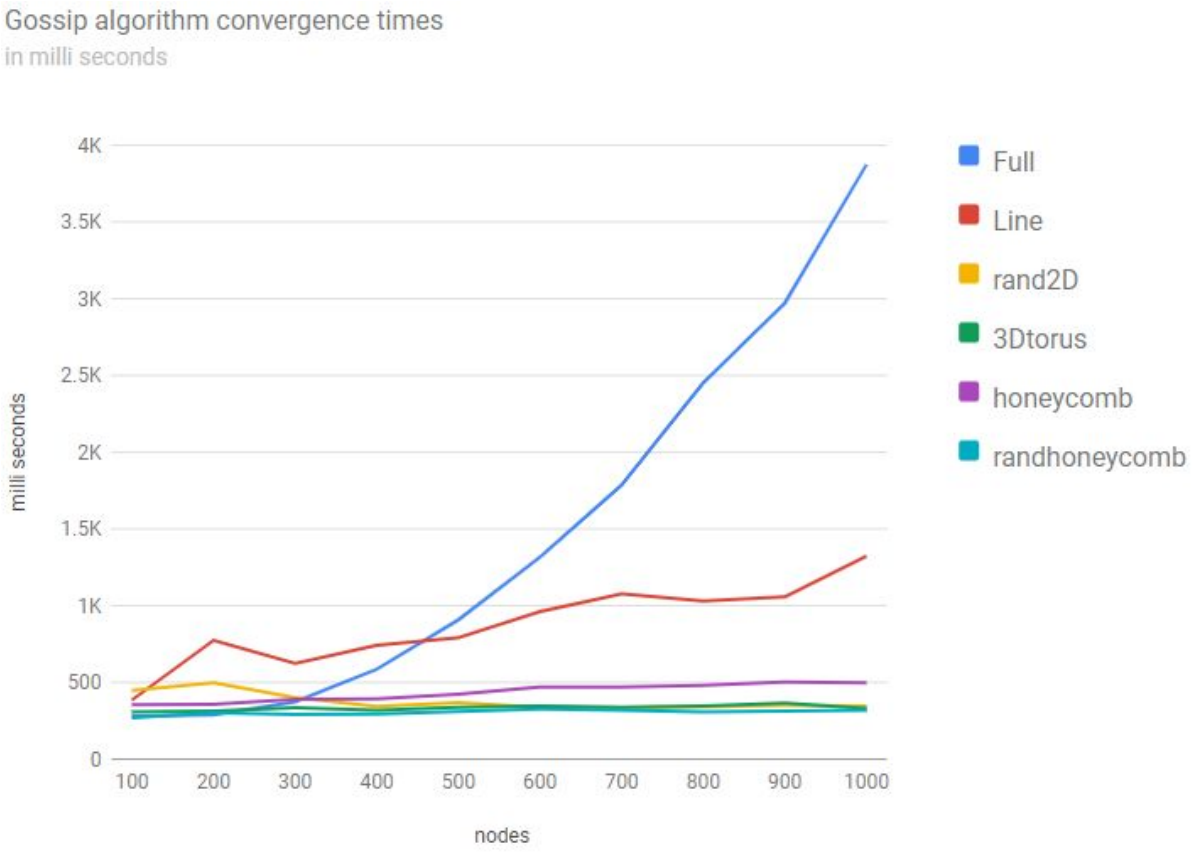
  > mix escript.build

  > escript proj2 <nodes> <topology> <algorithm>

  Example: escript proj2 1000 honeycomb gossip

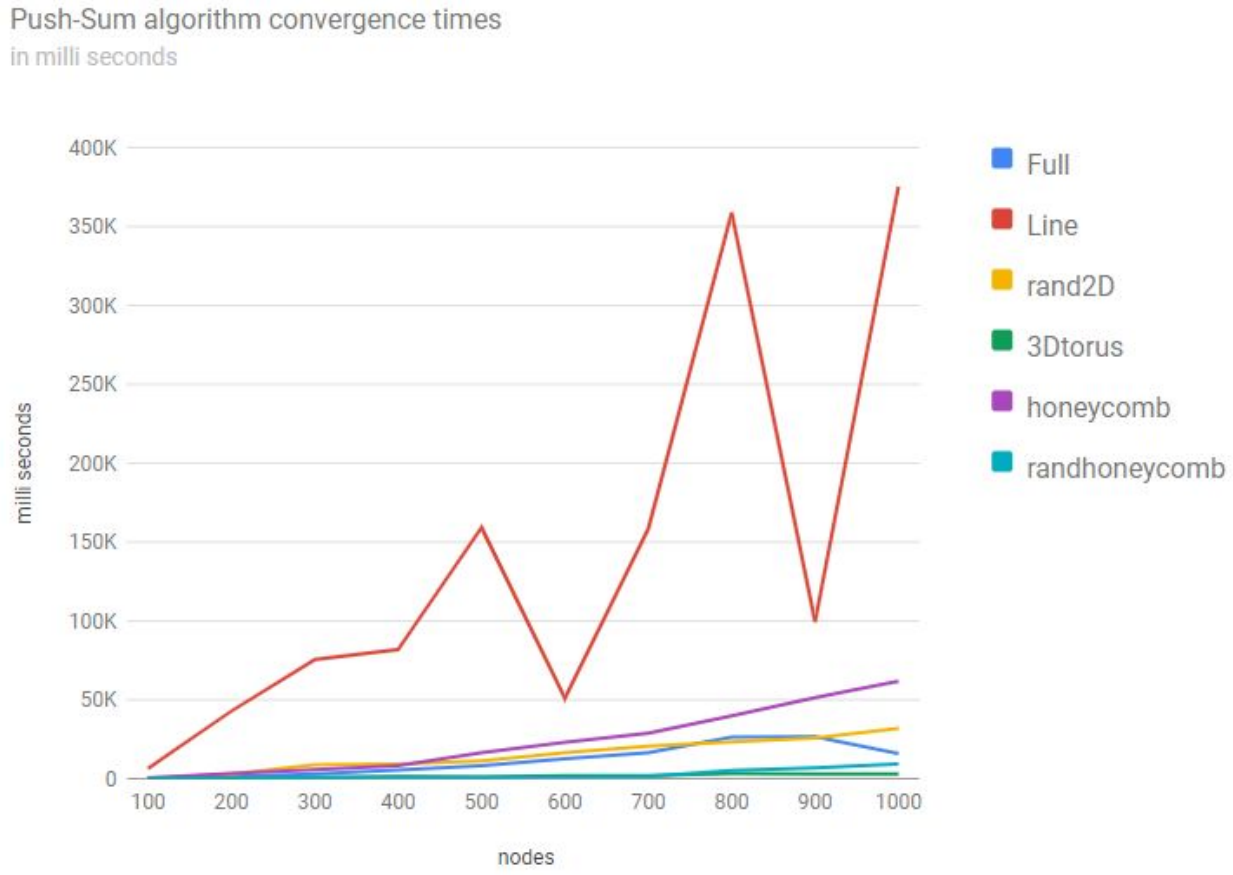
  escript proj2 1000 full pushsum

Observations:



In the above graph for Gossip Protocol, it may be observed that full and line topologies convergence time increase as the number of nodes increase. The other topologies,

rand2D, 3Dtorus, honeycomb, randhoneycomb, owing to their constant degree of connections, and topology scale better than full and line topologies.



In the case of Push-Sum, it may be observed that the line topology is erratic owing to the fact that once a node terminates/ becomes inactive, the connection is broken. Hence, the line is inefficient compared to other topologies.

Other topologies are performing better compared to the line topology.

Maximum working nodes for Gossip Algorithm:

Topology	Maximum nodes	Time for Convergence
Full network	5000	121124ms
Line	5000	2346ms
rand2D	5000	1253ms
honeycomb	5000	370ms
3Dtorus	5000	276ms
randhoneycomb	5000	237ms

:

Maximum working nodes for Pushsum Algorithm:

Topology	Maximum nodes	Time for Convergence
Line	5000	2754ms
3Dtorus	5000	35001ms
randhoneycomb	5000	12574ms