COP5615 – Project2

Neel Rami, UFID: 7712-3151

Ma Haodi, UFID:7719-2198

Email: nrami@ufl.edu , ma.haodi@ufl.edu

Oct 1, 2018

1. Introduction

The goal is to determine the convergence of such algorithms through a simulator based on actors written in Elixir.

In this project we manage to use two algorithm, Gossip and Push-Sum algorithm, on 6 different topologies: Full Network, 3D Grid, Random 2D Grid, Torus, Line, Imperfect Line.

- **Full Network:**
  Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.
- **3D Grid:**
  Actors form a 3D grid. The actors can only talk to the grid neighbor. Since the node number of 3D grid must be a perfect cube, we implement a function to turn the input node number into the nearest perfect cube number that is larger than the origin. For example, if the input node number is 20, our program will turn it into 27.
- **Random 2D Grid:**
  Actors are randomly position at x,y coordinates on a [0-1.0]*[0-1.0] square. Two actors are connected if they are within .1 distance to other actors. In this topology we implement a function to make sure that different nodes won't have same coordinates.
- **Torus:**
  Actors are arranged in a torus. That is, each actor has 4 neighbors (similar to the 2D grid) but both directions are closed to form circles. We implement to make the node number to the nearest perfect square number. E.g. if the input node number is 5, then the program will increase it to 9 to form a torus.
- **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).
- **Imperfect Line:**
  Line arrangement but one random other neighbor is selected from the list of all actors.

2. Assumptions made:

- For torus, we assume that the torus is 2D torus.
- For 3D Grid, we assume that the number of nodes in the grid should be a perfect cube number.
- For Random 2D grid topology, we generate numbers up to 3 decimal digits, so only 1000 nodes will have distinct co-ordinates. If more than 1000 nodes are created, then topology won't be formed. Also, random number generation and calculating distance takes $0(n^2)$ time, so the largest network for Random 2D grid topology is 1000.

3. Implementation details

a) File explanation:

a. topology.ex

In this file we implement the six topologies we mentioned in the previous part.

b. proj2.ex

This file serves as entry point of the project. Here in this file we initiate Gossip and Push Sum Protocols.

c. server1.ex

GenServer callback methods for Gossip Protocol are implemented here.

d. pushSumGS.ex

GenServer callback methods for Push Sum Protocol are implemented here.

b) Instructions for running the code:

a. For Ubuntu based systems:

1. Go the project directory
2. Type the command in the terminal: mix escript.build (Optional)
3. Type the command in the terminal: ./proj2 100 line gossip
4. Here the first command line argument is the number of nodes
5. Here the second command line argument is the type of topology you choose
6. Here the third command line argument is the name of the algorithm you choose
7. General command: ./proj2 <node-num> <topology> <algorithm>

b. For Windows:

1. Go the project directory
2. Type the command in the cmd: mix escript.build (Optional)
3. Type the command in the cmd: escript .\proj2 100 line gossip
4. Here the first command line argument is the number of nodes

5. Here the second command line argument is the type of topology you choose
6. Here the third command line argument is the name of the algorithm you choose
7. General command: .\proj2 <node-num> <topology> <algorithm>

4. Result for Gossip:

Important Point to Note:

The convergence time for all topologies and both the algorithms is the Expected Convergence time and not the actual convergence time. We calculate the expected convergence time by running the program for 5 times and then find the average.

How to we calculate convergence time:

Just before the start of the gossip algorithm, we use the time and store it as startTime and just after the algorithm terminates, we again calculate the current time and subtract startTime from it.

a) Full Network:

Nodes number	Convergence Time(ms)
50	2546
100	2830
200	3079
500	3207
1000	3629

b) 3D Grid:

Nodes number	Convergence Time(ms)
50	2567
100	2610
200	2960
500	3028
1000	3271

c) Random 2D Grid:

Nodes number	Convergence Time(ms)
50	*
100	*
200	*
500	3432
1000	3464

***: For small number of nodes, many nodes have 0 neighbors, so gossip will not converge as there are many nodes with 0 neighbors.**

d) Torus:

Nodes number	Convergence Time(ms)
50	2667
100	2789
200	2959
500	3463
1000	3903

e) Line:

Nodes number	Convergence Time(ms)
50	5333
100	9394
200	19962
500	43739
1000	80741

f) Imperfect Line:

Nodes number	Convergence Time(ms)
50	2566
100	2830
200	2876
500	3146
1000	3257

g) Largest network size for every topology:

Topology	Largest Node Number
Full Network	1000**
3D Grid	50000
Random 2D Grid	1000***
Torus	50000
Line	10000
Imperfect Line	100000

****: In our implementation, we have stored PIDs of processes in a list. Then we calculate neighbors based on indexes and then we map those indexes to PIDs. So, during the mapping stage, finding a PID takes O(n) for each neighbor index. Therefore, a lot of time is consumed in this mapping stage. So, we can test the Full Network Topology only for numbers in order of 10^3.**

*****: In Random 2D Grid, random number generation and then calculating distance takes $O(n^2)$ time. So, we can run the program for only numbers in order of 10^3 for Random 2D Grid.**

One improvement that can be done is to use a data structure like Hash Table which can find elements in constant time. Registry which is provided by Elixir can be used.

The below are graphs for topologies convergence time with Gossip algorithm for different node number.

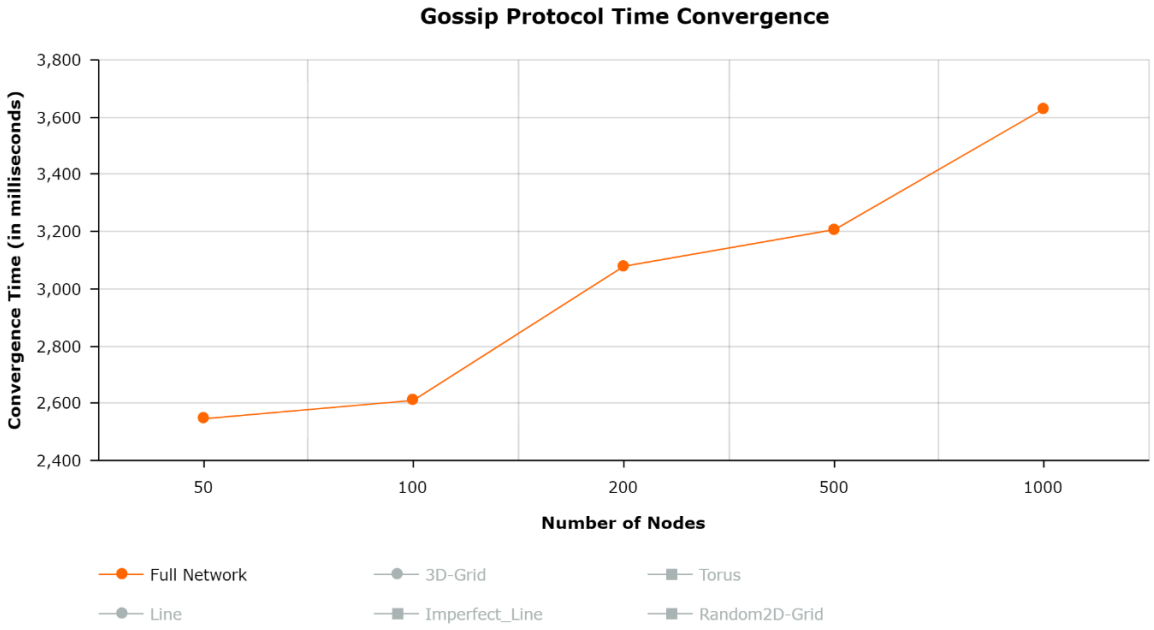

Figure 1:Full Network for Gossip

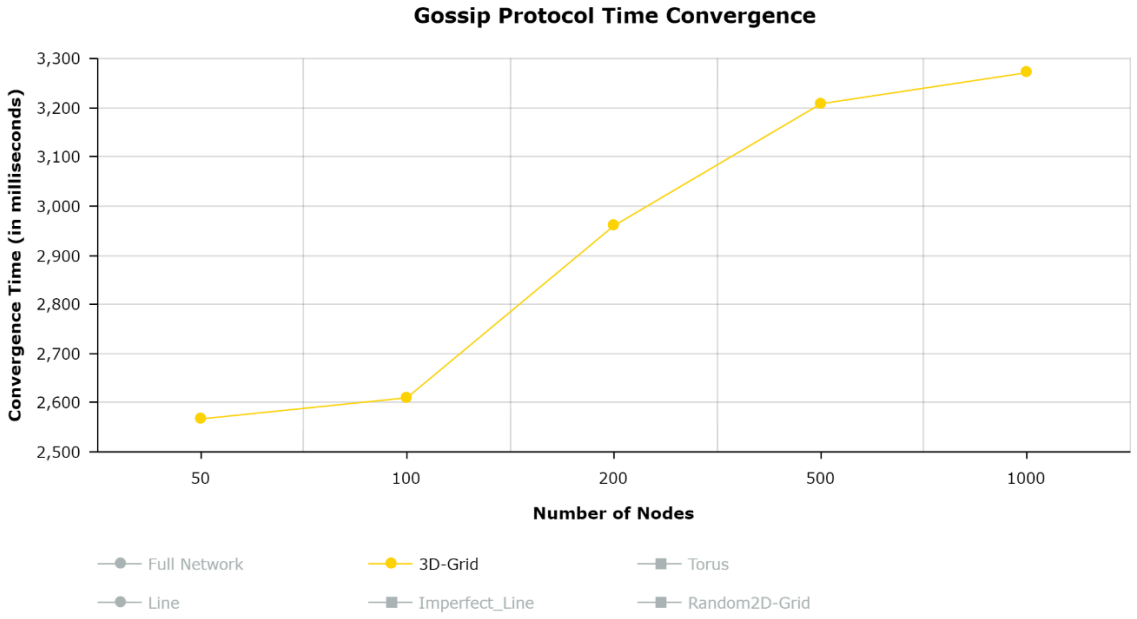

Figure 2:3D-Grid for Gossip

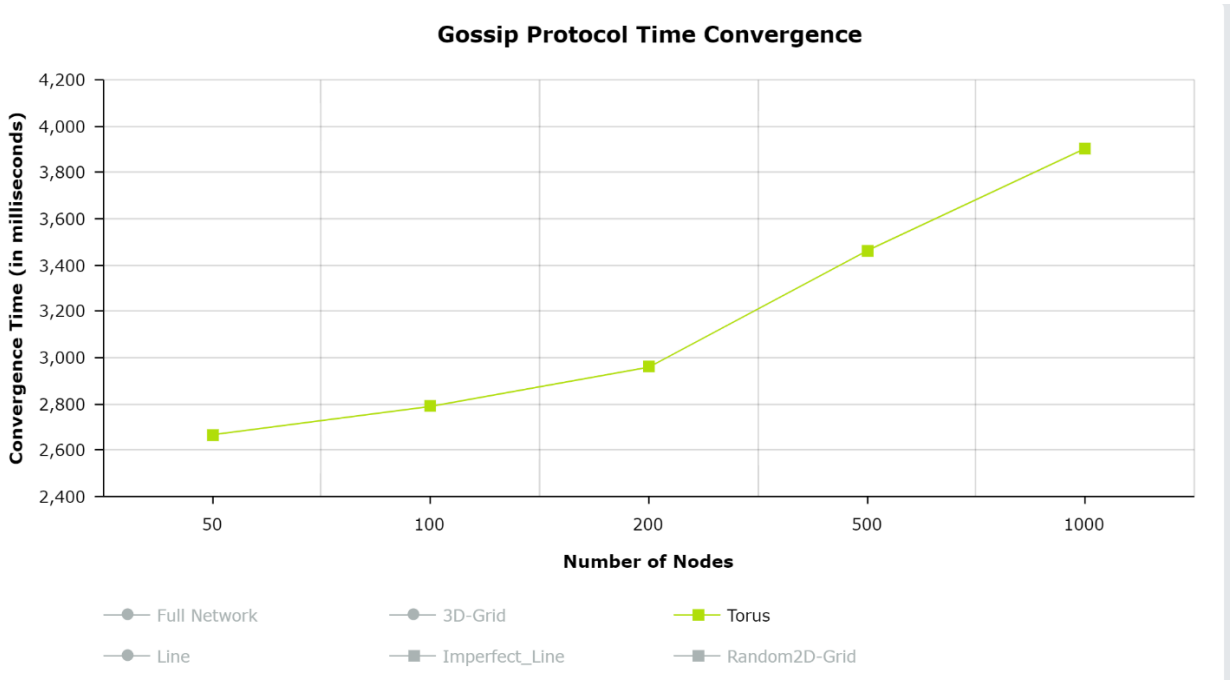

Figure 3:Torus for Gossip

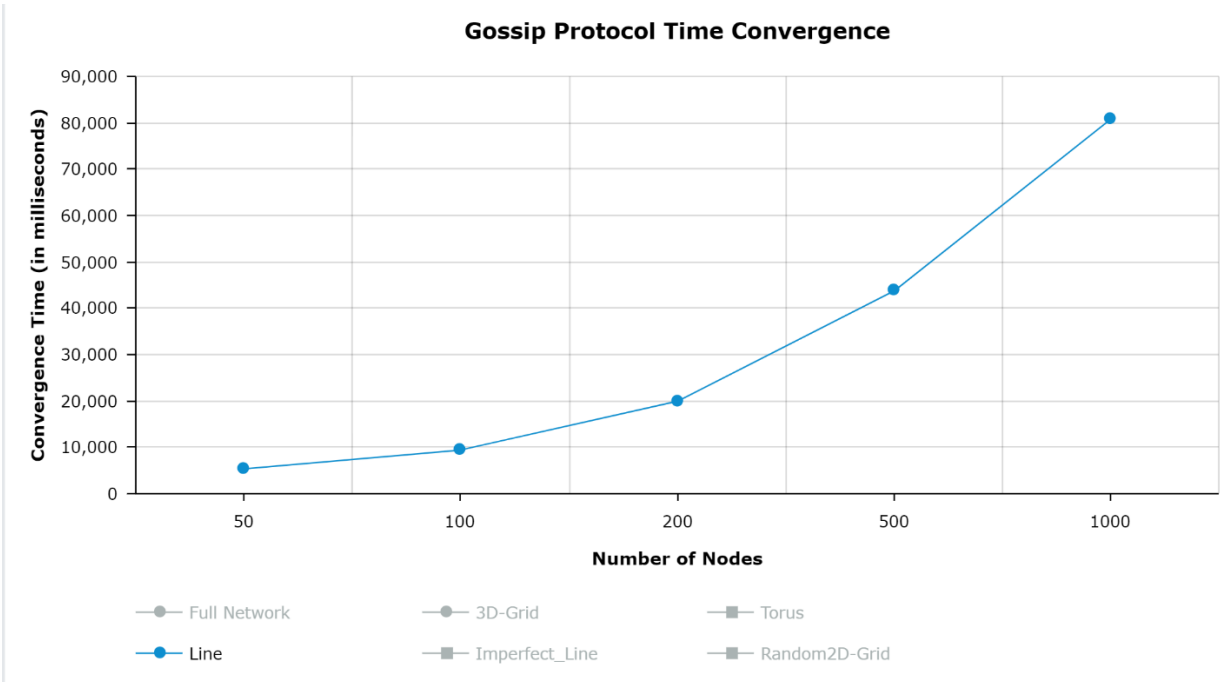

Figure 4:Line for Gossip

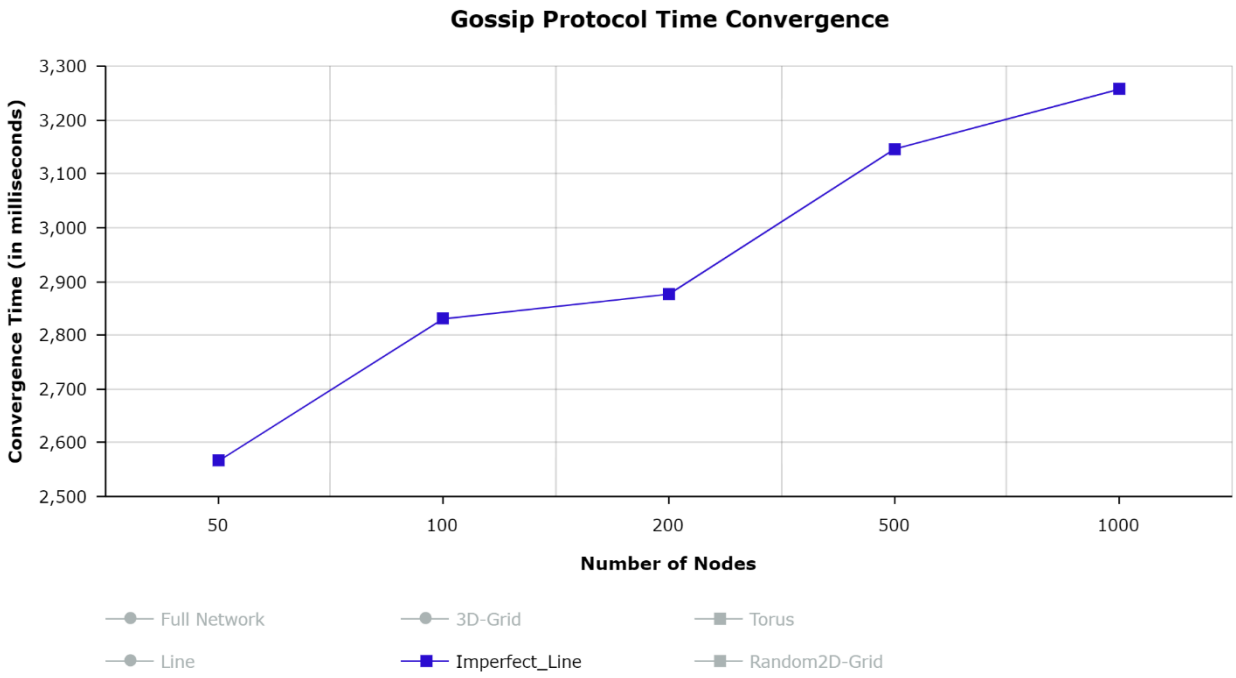

Figure 5:Imperfect Line for Gos

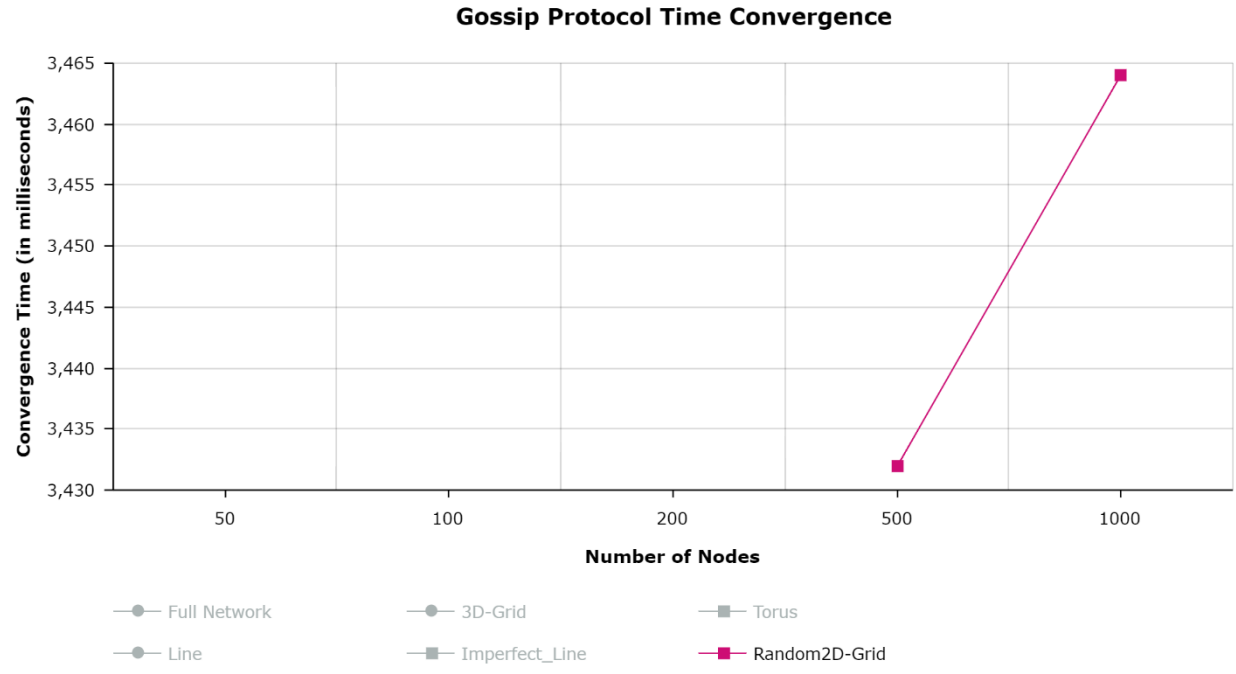

Figure 6:Random-2D-Grid for Gossip

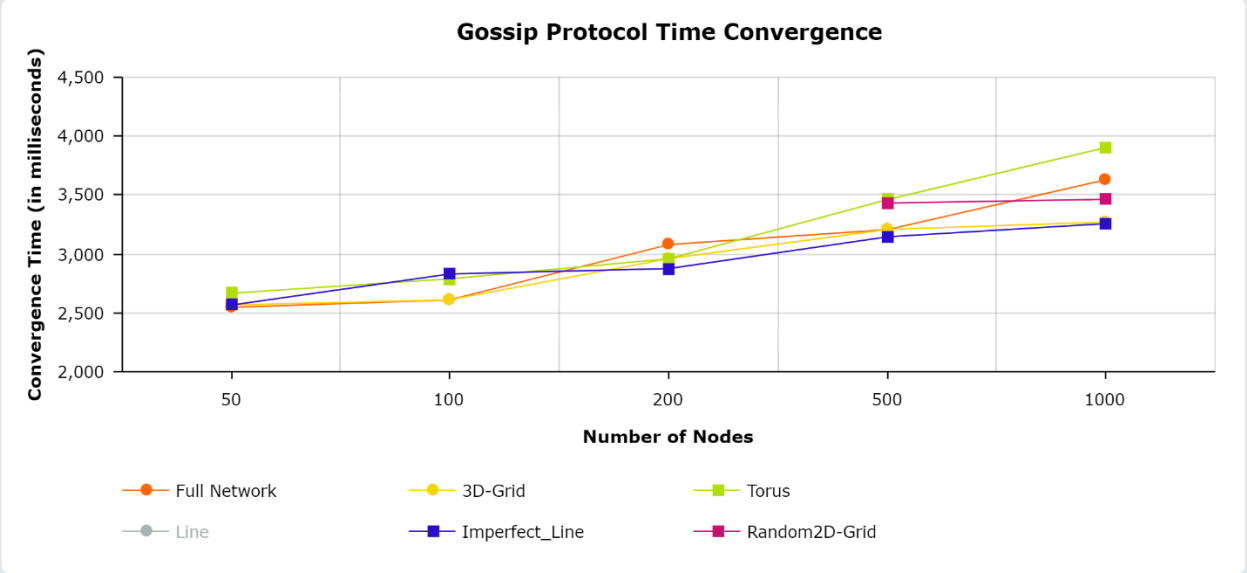

Figure 7: 5 topologies except Line comparison for Gossip

5. Interesting Findings for Gossip Protocol

1. Line Topology takes $O(n^2)$ time to converge when n is the number of nodes.
2. Imperfect Line and 3D Grid in our case perform the best and almost take the same amount of time
3. Full Topology performs better than Torus topology.
4. We couldn't draw any conclusion for Random 2D grid as our largest network for Random 2D grid consists of only 1000 nodes, but we can make an vague estimation that Random 2D grid might beat all other topologies as it has the smallest slope. (This is only an estimation so don't consider it to be always true).
5. Suppose T(topology) is the convergence time taken by a topology.
6. Then T(line) > T (any topology)
7. T(A) < T(torus) < T(line) where A is Imperfect Line, 3D or Full.

6. Result for Push-Sum

a) Full Network:

Nodes number	Convergence Time(ms)
50	10
100	25
200	59
500	299
1000	997

b) 3D Grid:

Nodes number	Convergence Time(ms)
50	39
100	109
200	251
500	1005
1000	3052

c) Random 2D Grid:

Nodes number	Convergence Time(ms)
50	*
100	*
200	*
500	2413
1000	4614

***: For small number of nodes, many nodes have 0 neighbors, so gossip will not converge as there are many nodes with 0 neighbors.**

d) Torus:

Nodes number	Convergence Time(ms)
50	30
100	68
200	261
500	1228
1000	4768

e) Line:

Nodes number	Convergence Time(ms)
50	327
100	2184

200	3449
500	15441
301000	68691

f) Imperfect Line:

Nodes number	Convergence Time(ms)
50	48
100	87
200	200
500	1430
1000	2690

g) Largest network size for every topology:

Topology	Largest Node Number
Full Network	1000**
3D Grid	50000
Random 2D Grid	1000***
Torus	50000
Line	10000
Imperfect Line	100000

****: In our implementation, we have stored PIDs of processes in a list. Then we calculate neighbors based on indexes and then we map those indexes to PIDs. So, during the mapping stage, finding a PID takes O(n) for each neighbor index. Therefore, a lot of time is consumed in this mapping stage. So we can test the Full Network Topology only for numbers in order of 10^3.**

*****: In Random 2D Grid, random number generation and then calculating distance takes $O(n^2)$ time. So, we can run the program for only numbers in order of 10^3 for Random 2D Grid.**

One improvement that can be done is to use a data structure like Hash Table which can find elements in constant time. Registry which is provided by Elixir can be used.

The below are graphs for topologies' convergence time with Push-Sum algorithm for different node number.

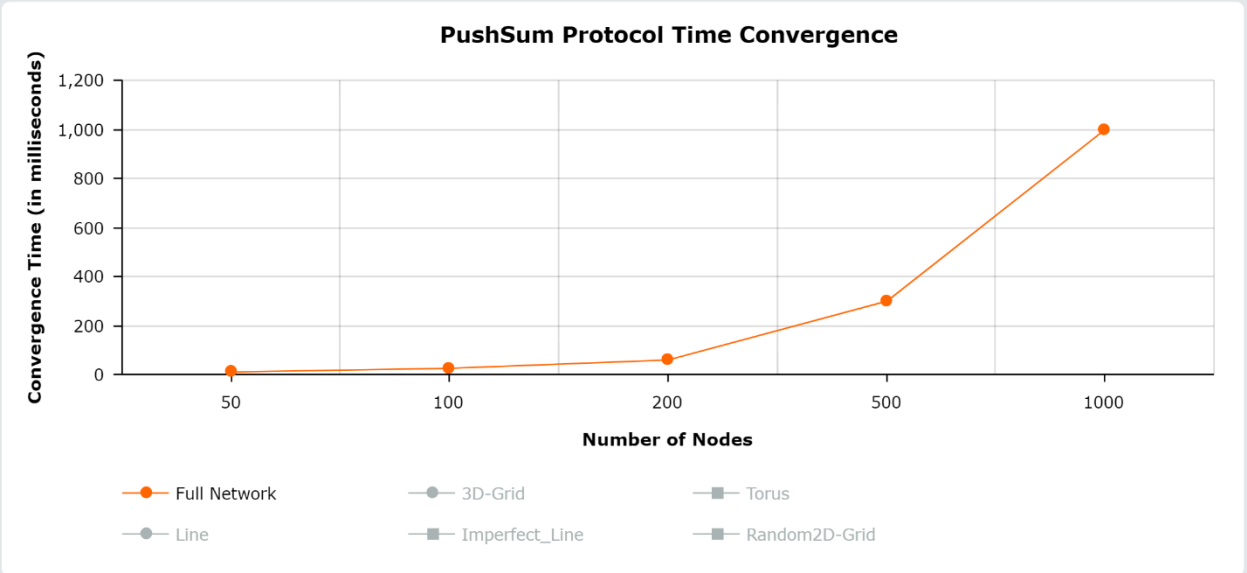

Figure 8: Full Network for Push-Sum

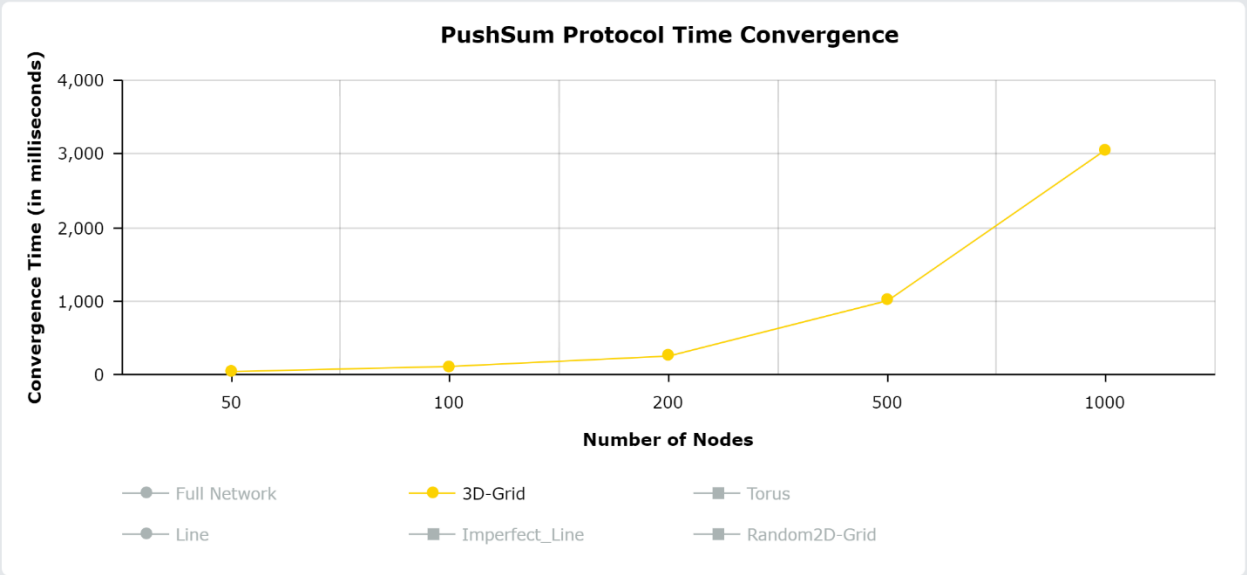

Figure 9: 3D-Grid for Push-Sum

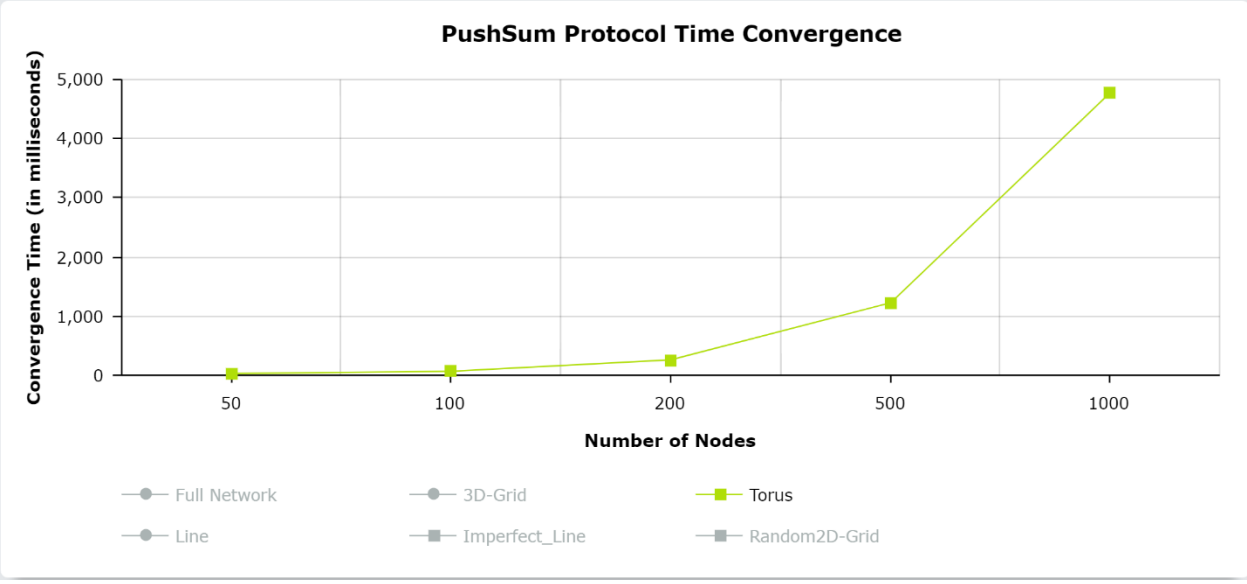

Figure 10: Torus for Push-Sum

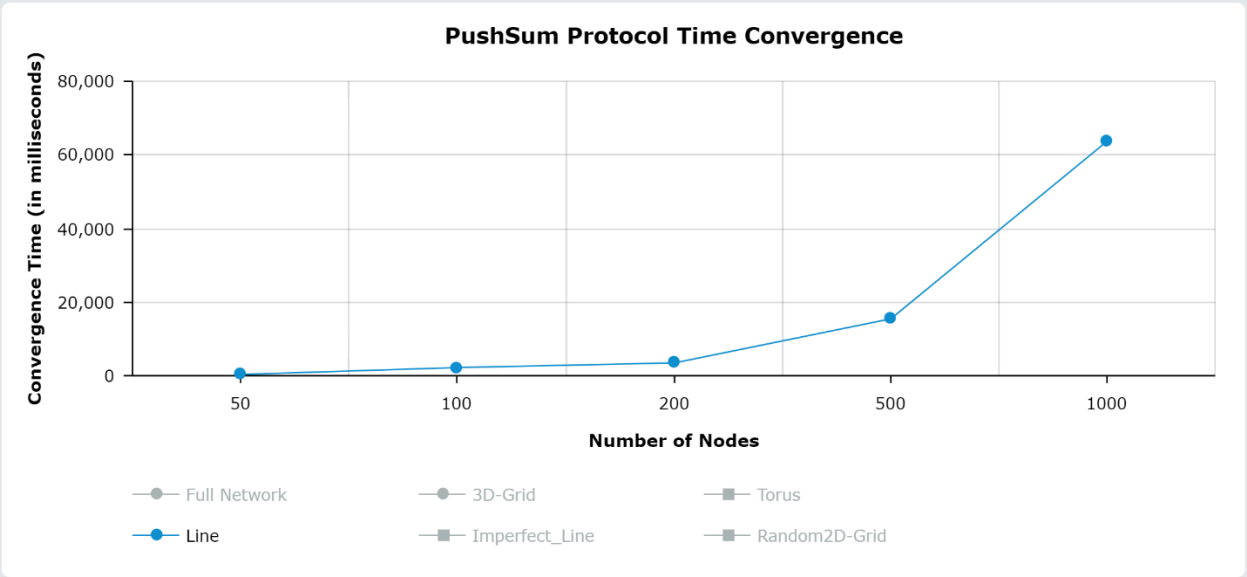

Figure 11: Line for Push-Sum

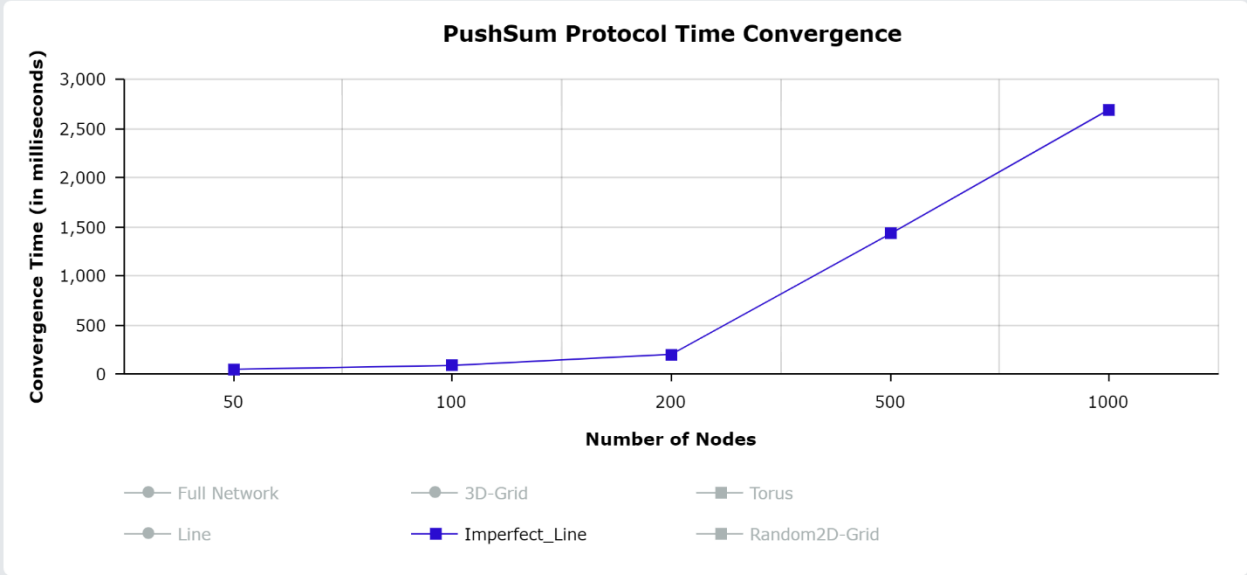

Figure 12: Imperfect Line for Push-Sum

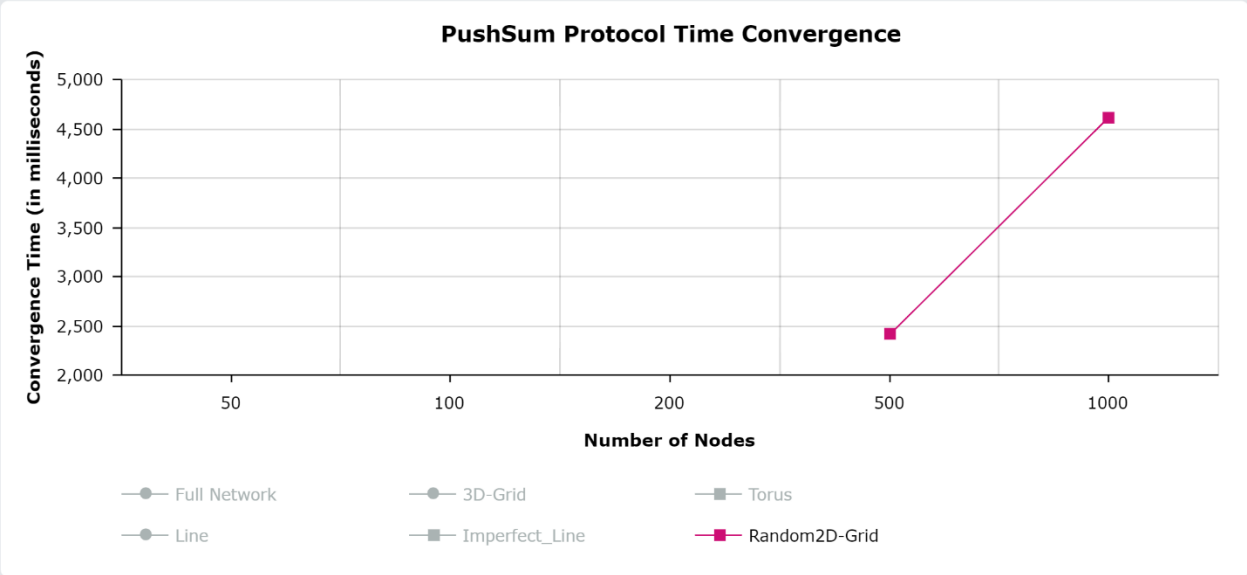

Figure 13:Random 2D-Grid for Push-Sum

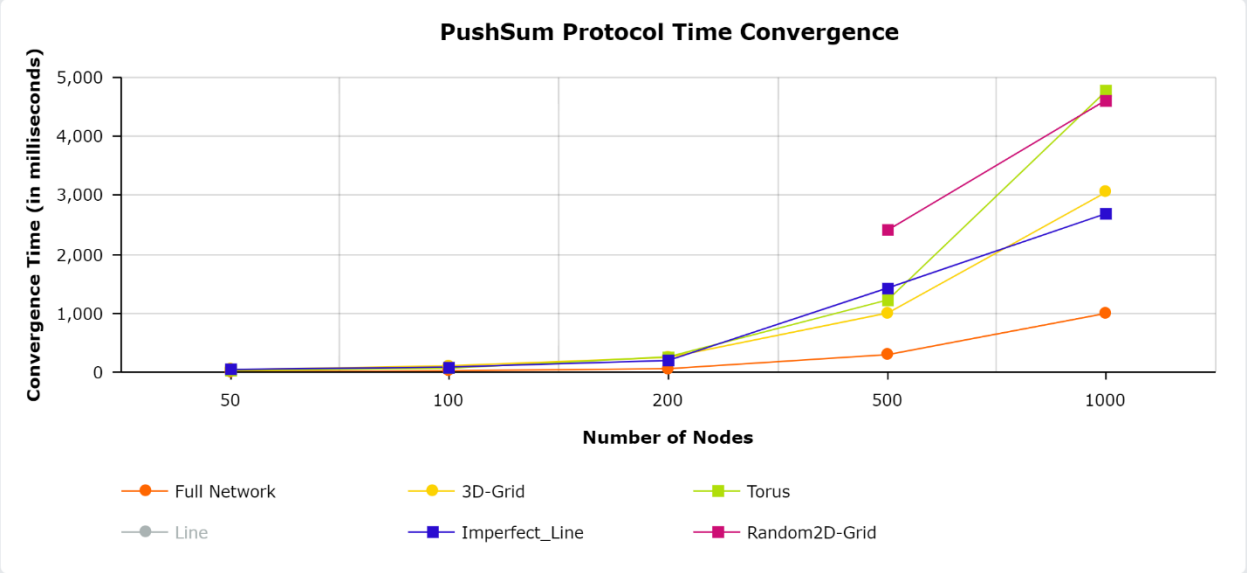

Figure 14: 5 Topologies except Line comparison for Push-Sum

7. Interesting Findings for Push Sum Protocol

1. As usual Line Topology took the largest time i.e. $O(n^2)$ time to converge when n is the number of nodes.
2. Surprisingly, Full Topology took the least time to converge.
3. Suppose T(topology) is the convergence time taken by a topology.
4. Then T(line) > T (any topology)
5. T(A) < T(torus) < T(line) where A is Imperfect Line, 3D or Full.
6. We couldn't draw any conclusion for Random 2D grid as our largest network for Random 2D grid consists of only 1000 nodes, but we can make a vague estimation that Random 2D grid might beat all other topologies as it has the smallest slope. (This is only an estimation so don't consider it to be always true).
7. T(Full) < T(any topology)