COP5615 – Project2

**Neel Rami, UFID: 7712-3151**

**Ma Haodi, UFID:7719-2198**

Email: [nrami@ufl.edu](mailto:nrami@ufl.edu) , [ma.haodi@ufl.edu](mailto: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 along with failure models.

In this project we manage to use two algorithm, Gossip 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.

1. **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(n2) time, so the largest network for Random 2D grid topology is 1000.
* Here when we say a node fails, in practical scenario the outgoing connections fail.

1. **Failure Models Implemented:**
2. 10% Failure Model
3. 20% Failure Model
4. 90% Failure Model
5. **Implement details:**
   1. **File explanation:**
      1. **topology.ex**

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

* + 1. **proj2.ex**

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

* + 1. **server1.ex**

GenServer callback methods for Gossip Protocol are implemented here.

* + 1. **server2.ex**

GenServer callback methods for Push Sum Protocol are implemented here.

* 1. **Instructions for running the code:**
     1. For Ubuntu based systems:

Go the project directory

Type the command in the terminal: mix escript.build (Optional)

Type the command in the terminal: ./proj2bonus 100 line gossip 10

Here the first command line argument is the number of nodes

Here the second command line argument is the type of topology you choose

Here the third command line argument is the name of the algorithm you choose

Here the forth command line argument is the number of nodes that are going to fail in the network.

General command: ./proj2bonus <node-num> <topology> <algorithm><fail-node-num>

* + 1. For Windows:

Go the project directory

Type the command in the cmd: mix escript.build (Optional)

Type the command in the cmd: escript .\proj2bonus 100 line gossip 10

Here the first command line argument is the number of nodes

Here the second command line argument is the type of topology you choose

Here the third command line argument is the name of the algorithm you choose

Here the forth command line argument is the number of nodes that are going to fail in the network.

General command: .\proj2bonus <node-num> <topology> <algorithm> <fail-node-num>

1. **Result:**
   1. **Full Network:**

|  |  |
| --- | --- |
| Kill Nodes number | Convergence Time(ms) |
| 10% | 3281 |
| 20% | 4156 |
| 90% | 5985 |

* 1. **3D Grid:**

|  |  |
| --- | --- |
| Kill Nodes number | Convergence Time(ms) |
| 10% | 5328 |
| 20% | 6313 |
| 90% | \* |

* 1. **Random 2D Grid:**

|  |  |
| --- | --- |
| Kill Nodes number | Convergence Time(ms) |
| 10 | 5469 |
| 20 | 5657 |
| 90 | \* |

* 1. **Torus:**

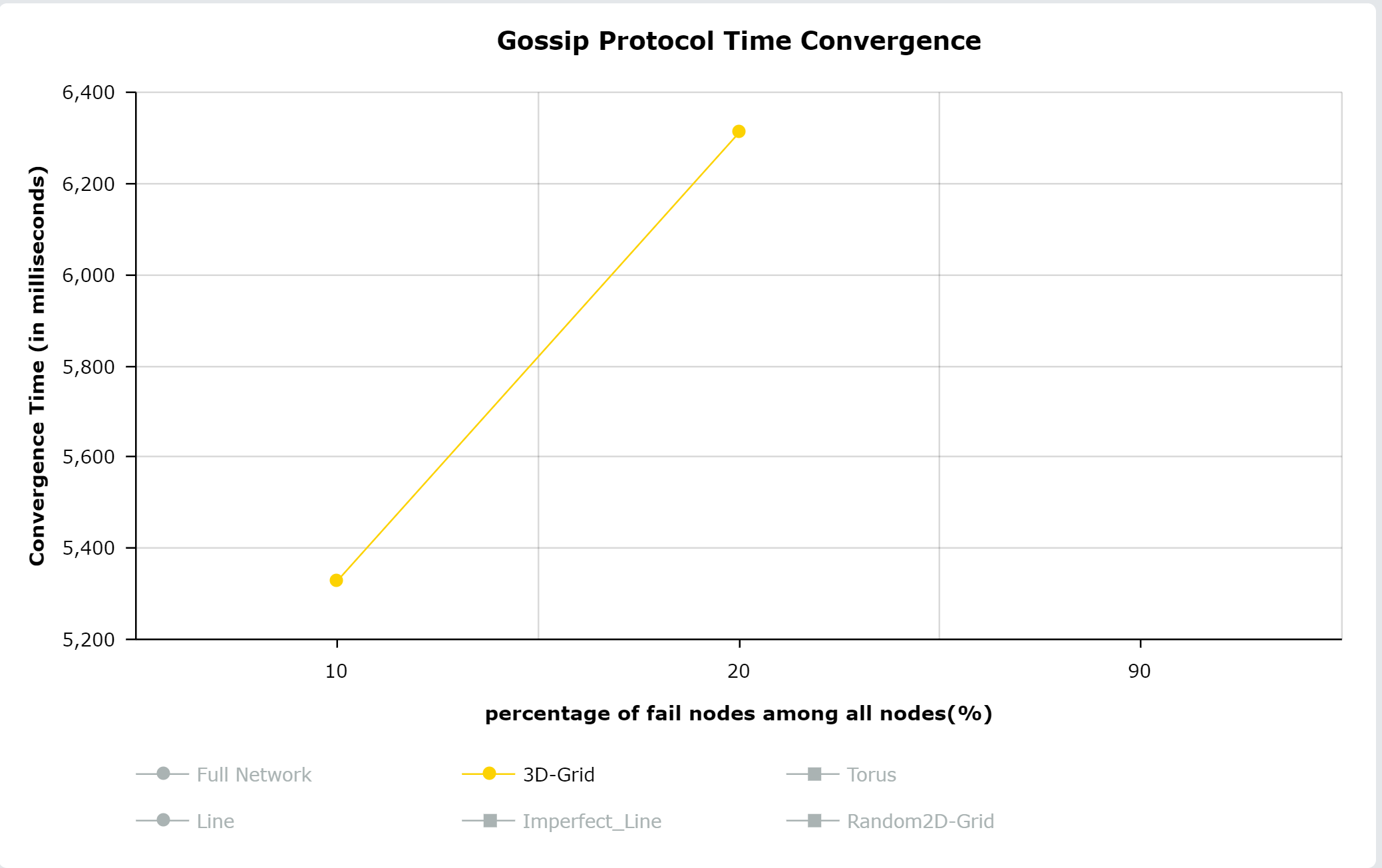
|  |  |
| --- | --- |
| Kill Nodes number | Convergence Time(ms) |
| 10 | 5360 |
| 20 | 5781 |
| 90 | 7734 |

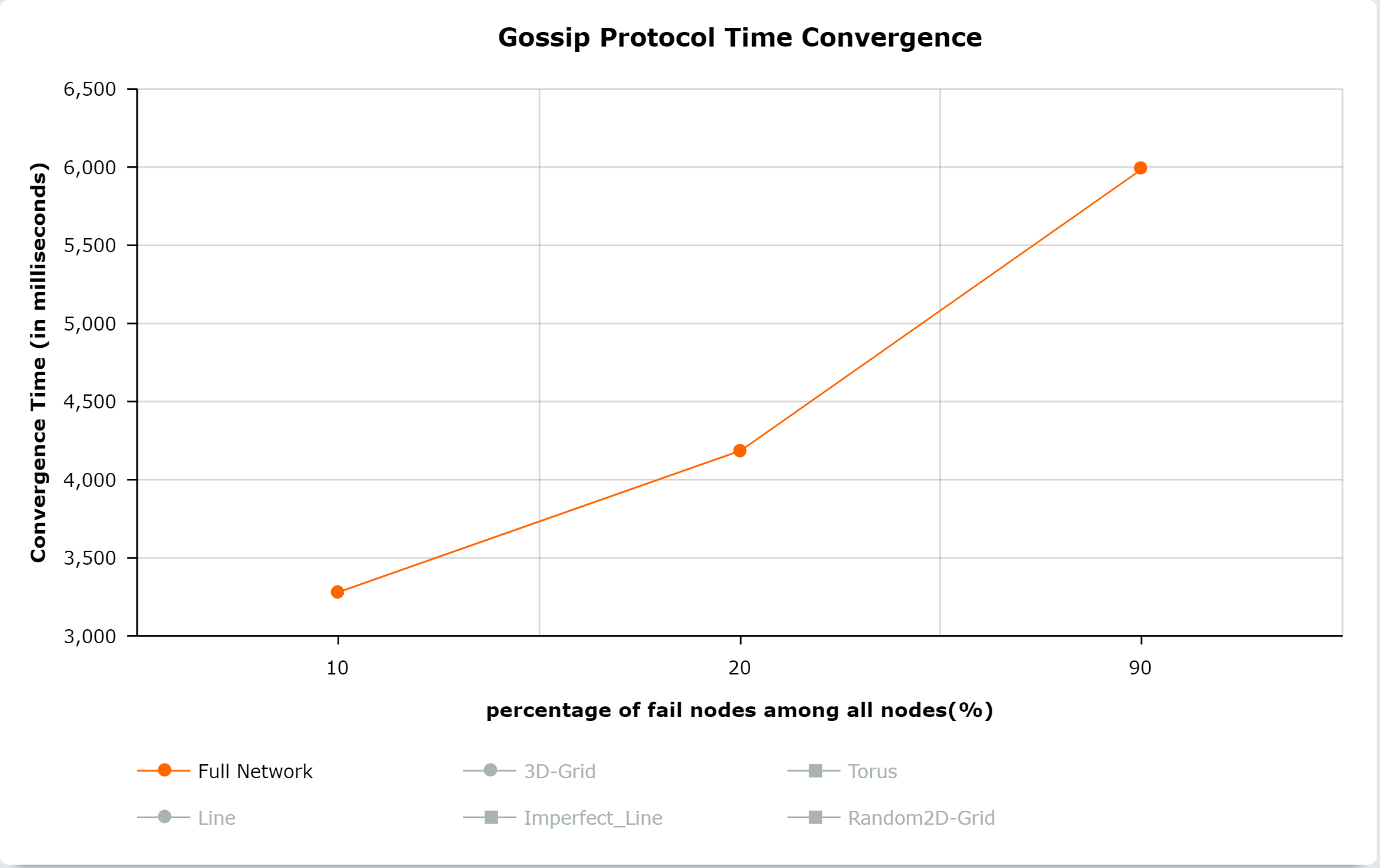
* 1. **Imperfect Line:**

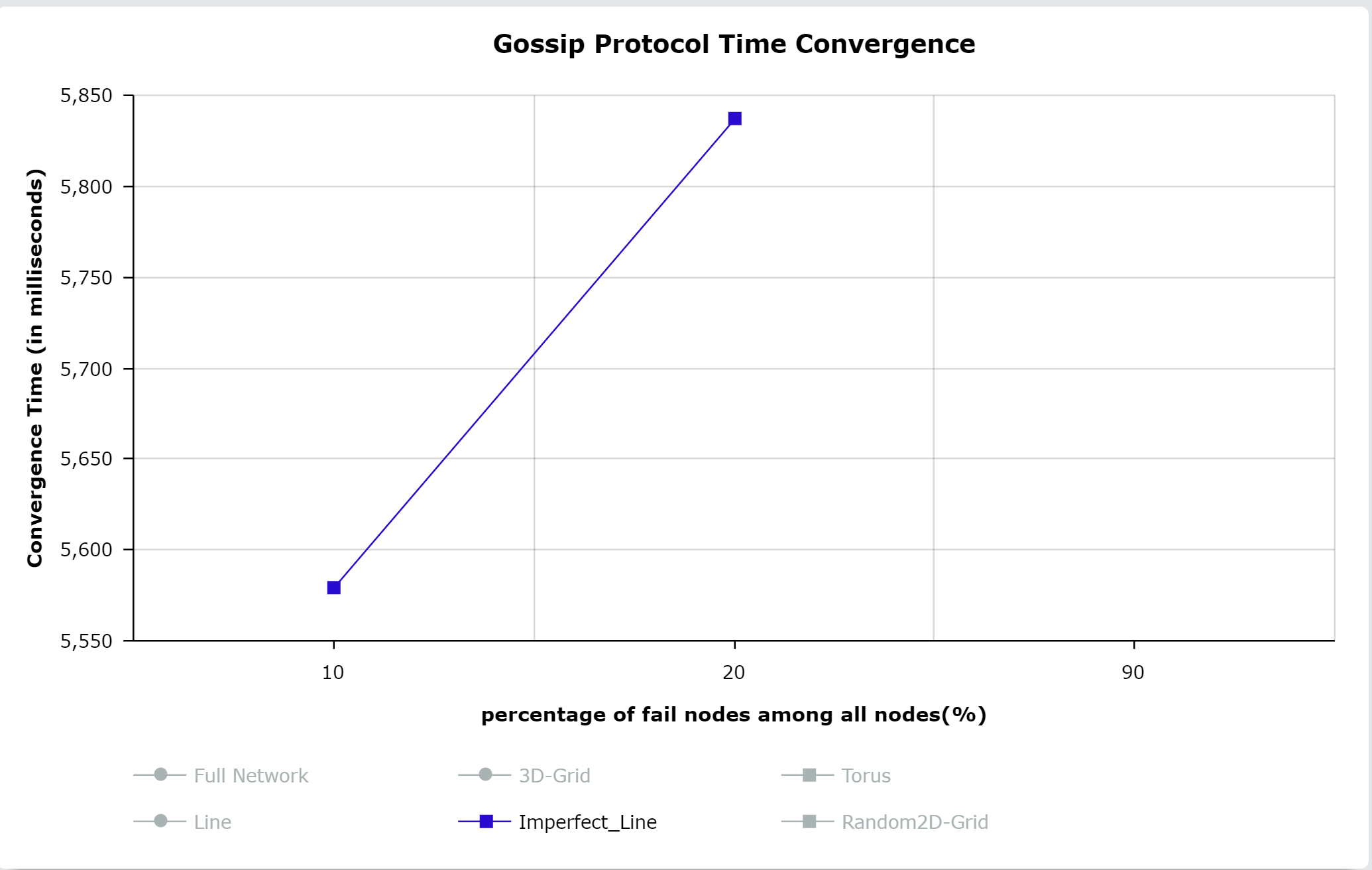
|  |  |
| --- | --- |
| Kill Nodes number | Convergence Time(ms) |
| 10 | 5579 |
| 20 | 5837 |
| 90 | \* |

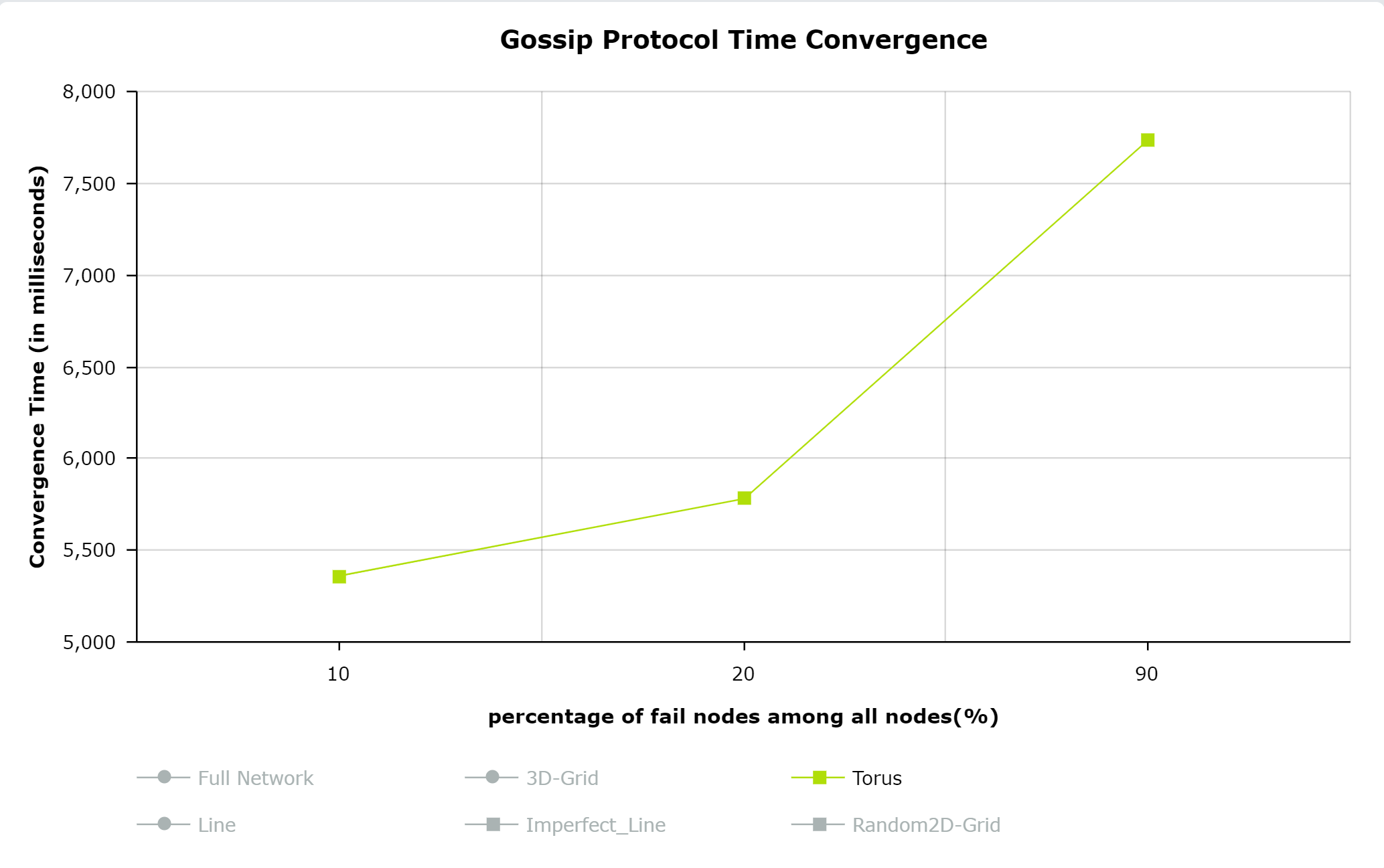
**The below will be graphs for topologies’ convergence time with Gossip algorithms for different node number.**

**Here number of nodes taken were 100, So 10% failure model means 10 nodes, 20% means 20 nodes and 90% means 90 nodes.**



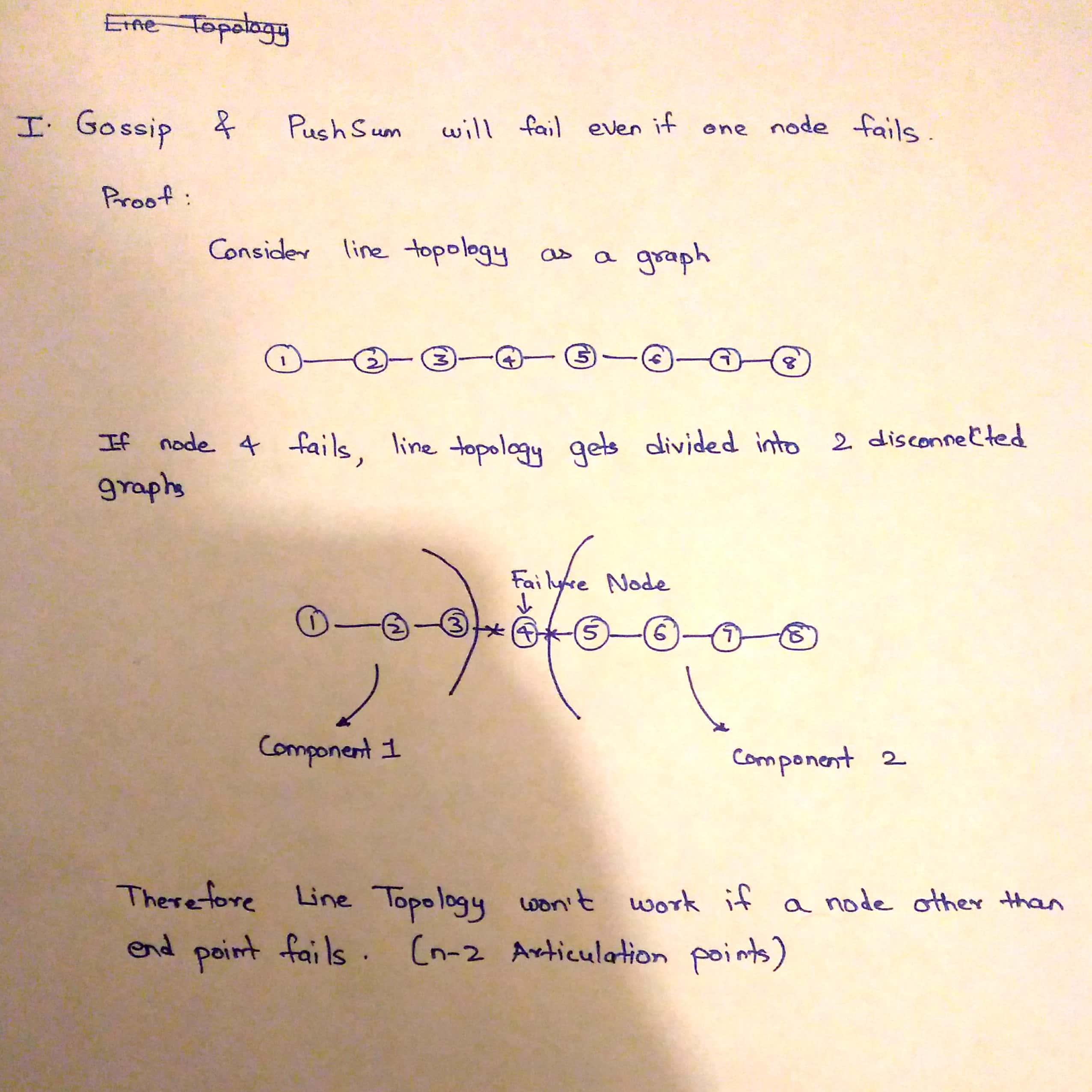






**Interesting finding for Gossip Protocol**

1. Line Topology will never converge in any failure model.



2. For Full Topology, even if 90% of nodes fail, gossip protocol still converges.

(Here number of nodes should be greater than 20)

3. For Torus Topology as well, even when 90% of nodes fail, gossip protocol still converges.

(Here number of nodes should be greater than 3)

4. For all other topologies, when 90% of nodes fail, the algorithm doesn’t converge when n is 100.

5. But all gossip will terminate for all other topologies as well.

