COP5615 - Distributed Operating System Principles
PROJECT-2

The goal of this project was to determine the convergence of gossip algorithms through a simulator based on the actors. Gossip type algorithms can be used both for group communication and aggregate for aggregate computation. Since actors are fully asynchronous, the particular type of Gossip implemented is the so-called Asynchronous Gossip.

Technologies used: Erlang.

Authors-

1) Pranav Gautam (UFID: 74088982)
2) Shikhar Panwar (UFID: 47449974)

What is working?-

In this implementation of the gossip algorithm, a node ends after hearing a rumor 10 times, stopping the transmission of the rumor to a random neighbor. When every node in the network has ended, the gossip implementation's convergence is assessed. We round the number of nodes for 2D grid and imperfect 2D grid topologies to the closest perfect square.

Every node in our implementation of the Push-Sum algorithm is started with the settings of s = i and w = 1, which are recommended in the project handout. For 2D grid and Imperfect 2D grid topologies, the number of nodes is rounded to the next perfect square, much as in our gossip implementation. The primary process starts by requesting the start of a random node, which then sends a message to a random neighbor that contains the tuple (s/2, w/2) while maintaining the values of s/2 and w/2 as its state. An actor adds a message tuple to its state, maintains half of its value, and passes the other half to a random node when it gets a message tuple. The actor terminates, or stops delivering a tuple to a random neighbor, when its s/w ratio does not change by more than 1.0e-10 for three consecutive iterations. The technique converges when the sum estimates, or s/w, converge to the average of the sum.

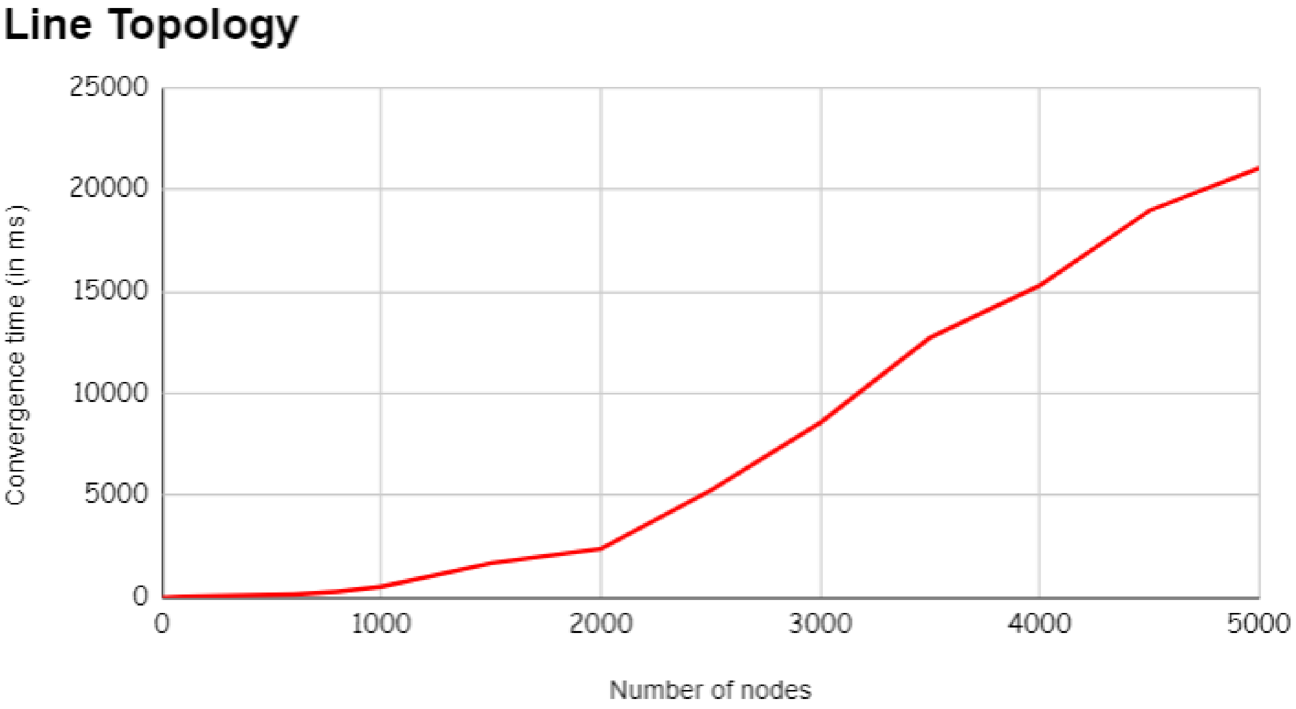
What is the largest network for each type of topology and algorithm?-

The real network topology has a significant impact on how quickly gossip protocols spread. The convergence times for each of the following network topologies were investigated. The topologies determine who is considered as a neighbor.

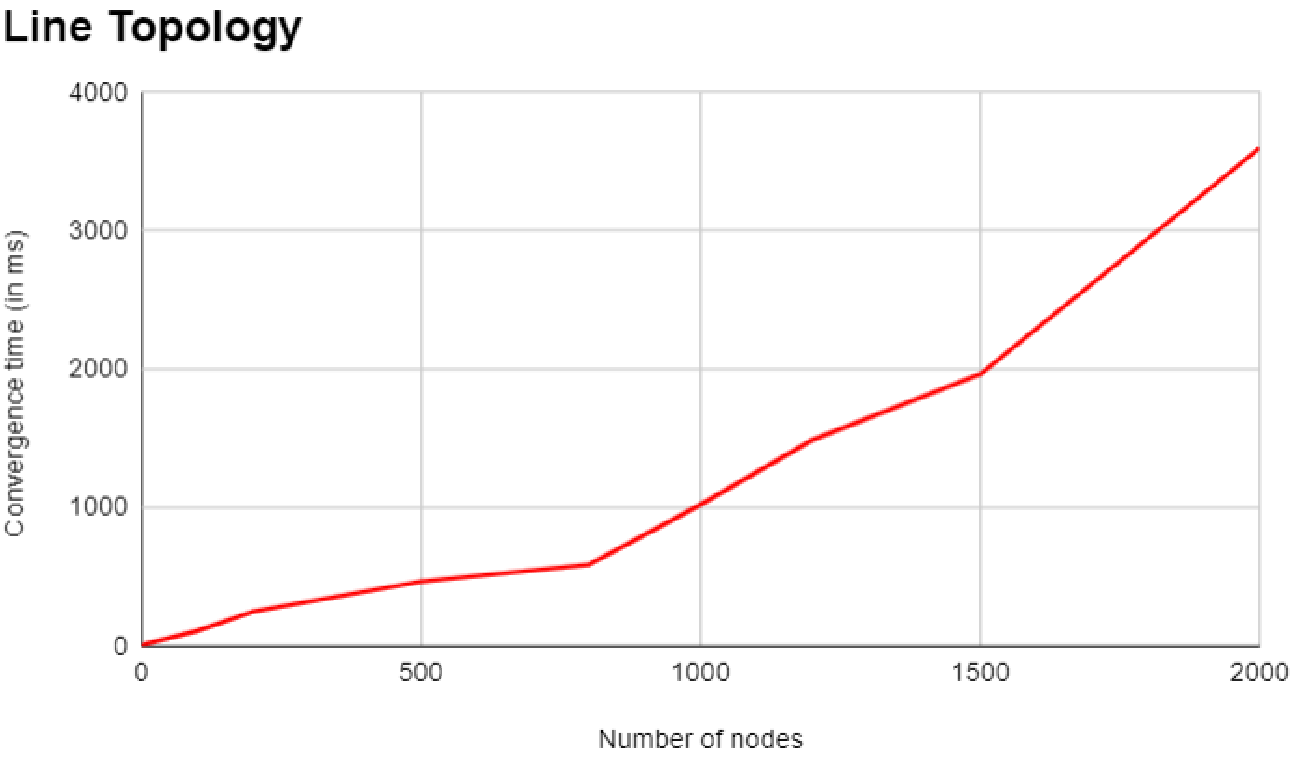
Below are the graphs for different types of topologies for gossip and push sum algorithm.

1) Line Topology - 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).

Gossip:

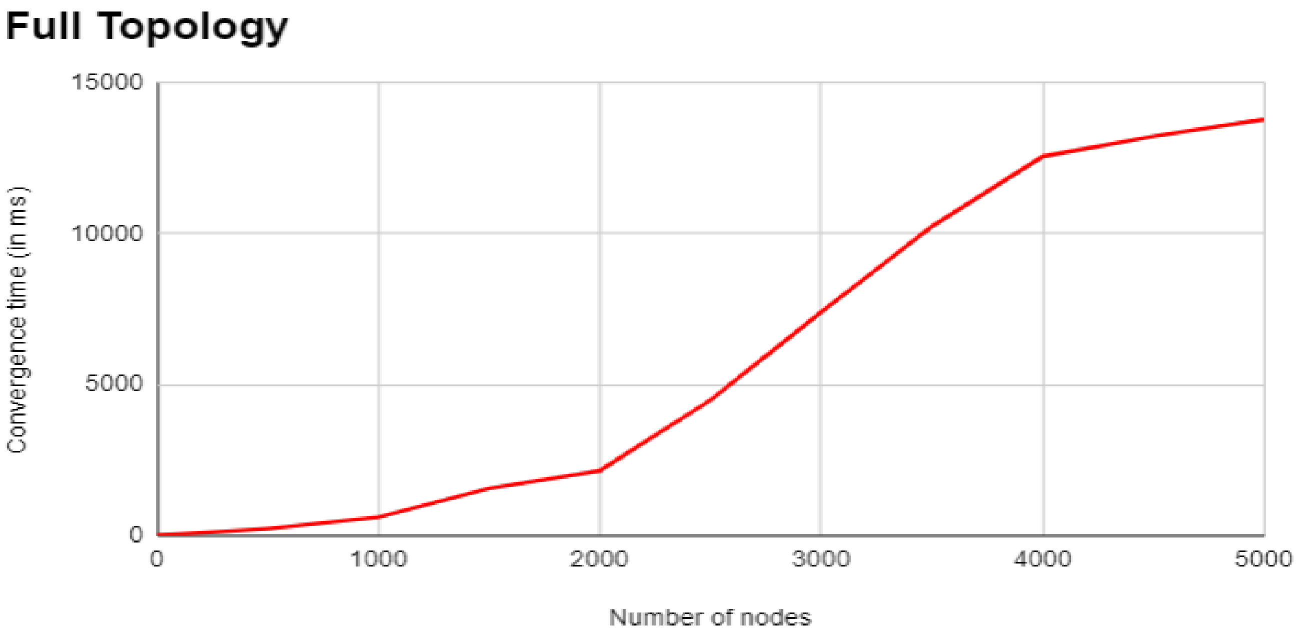


Push Sum:

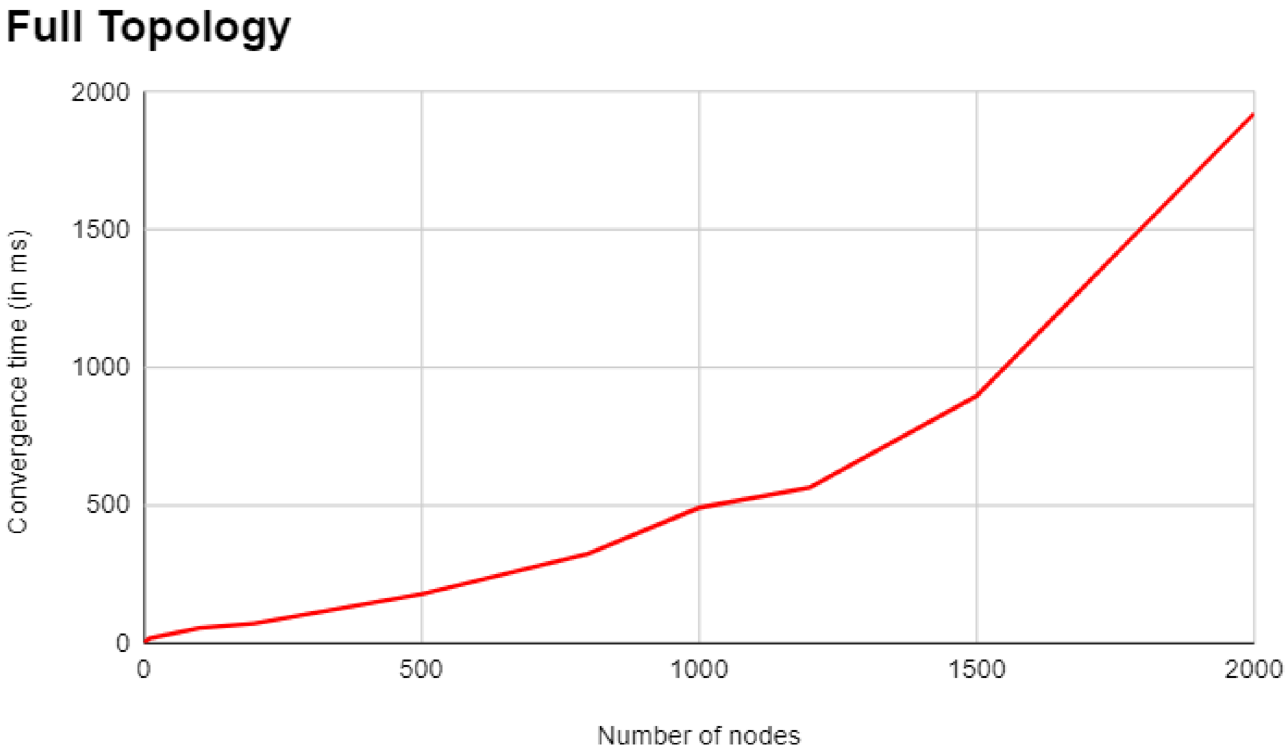


(2) Full Topology - Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.

Gossip:

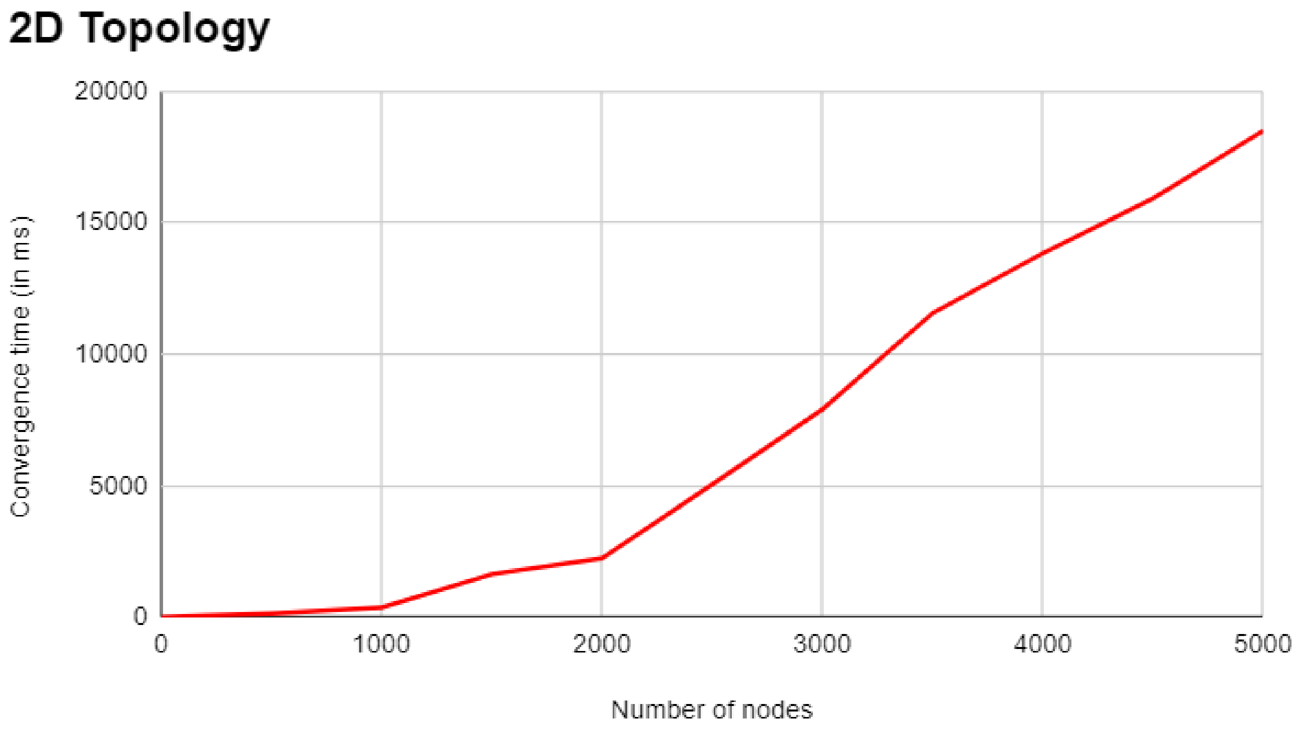


Push Sum:

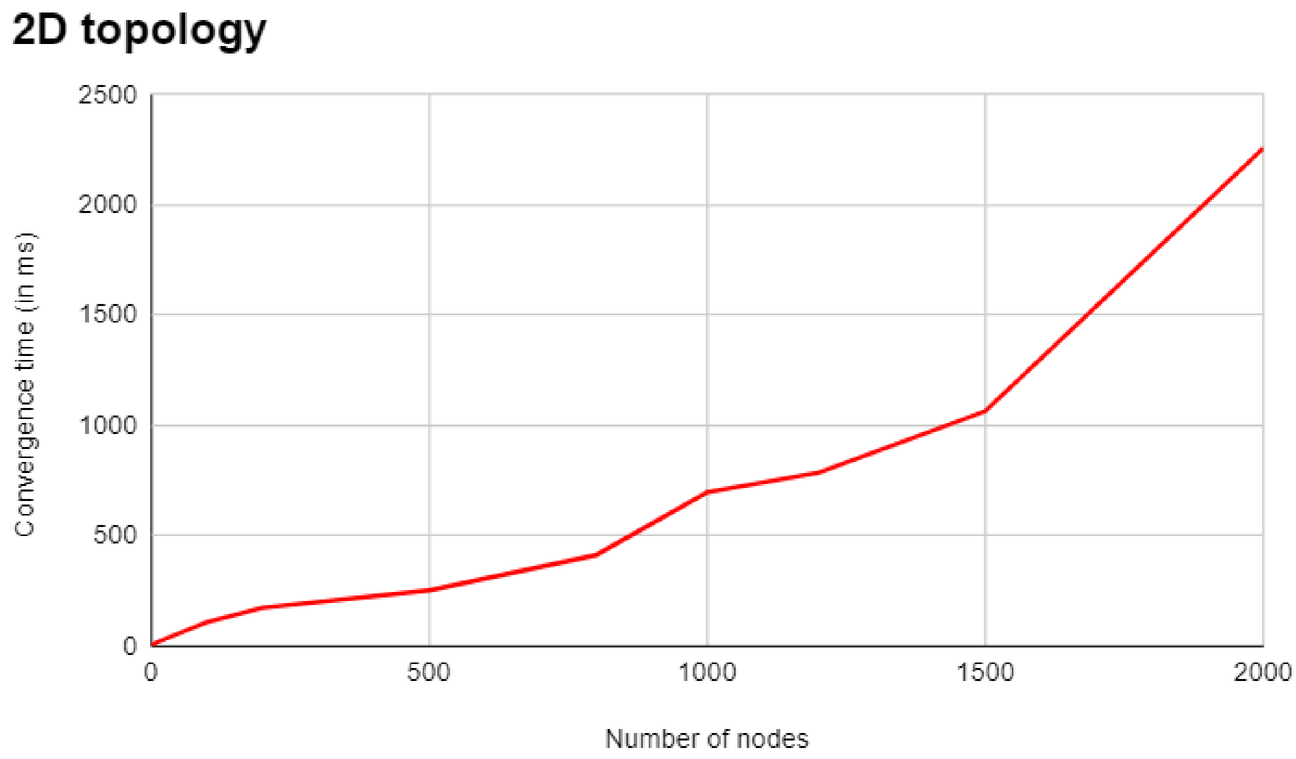


(3) 2D Topology - Actors form a 2D grid. The actors can only talk to the grid neighbors.

Gossip:

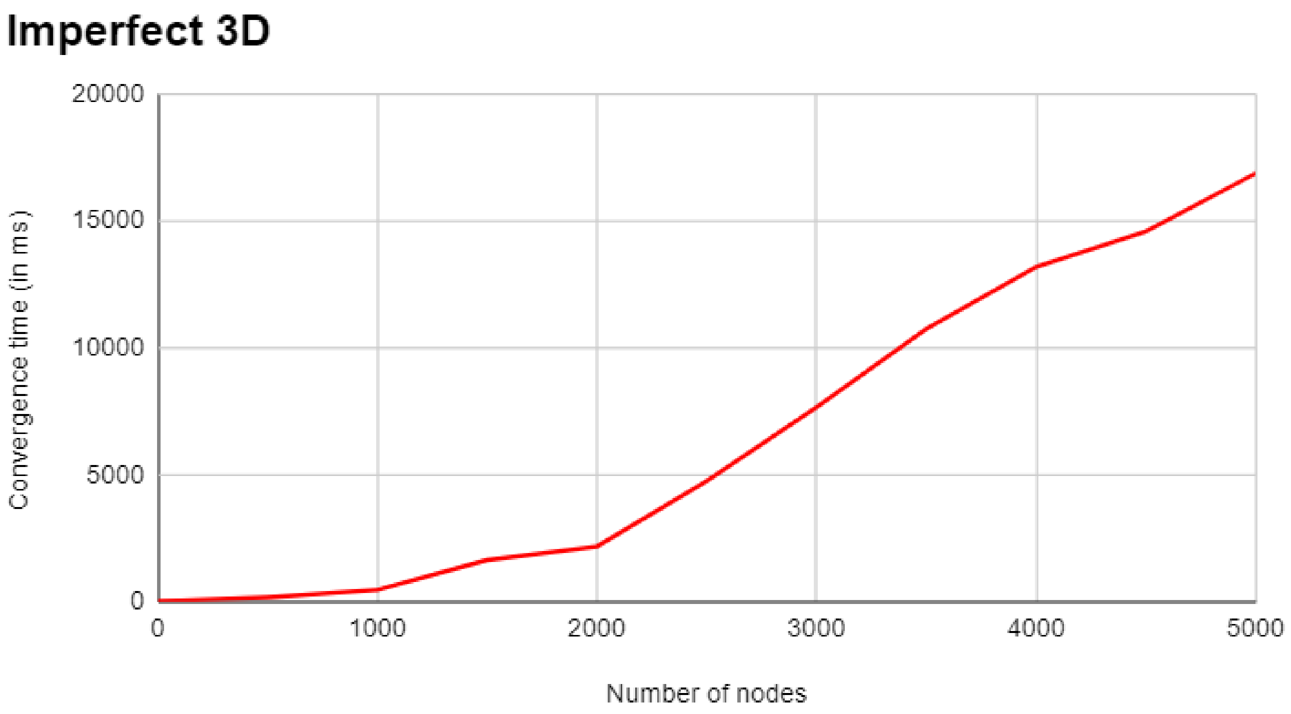


Push Sum:

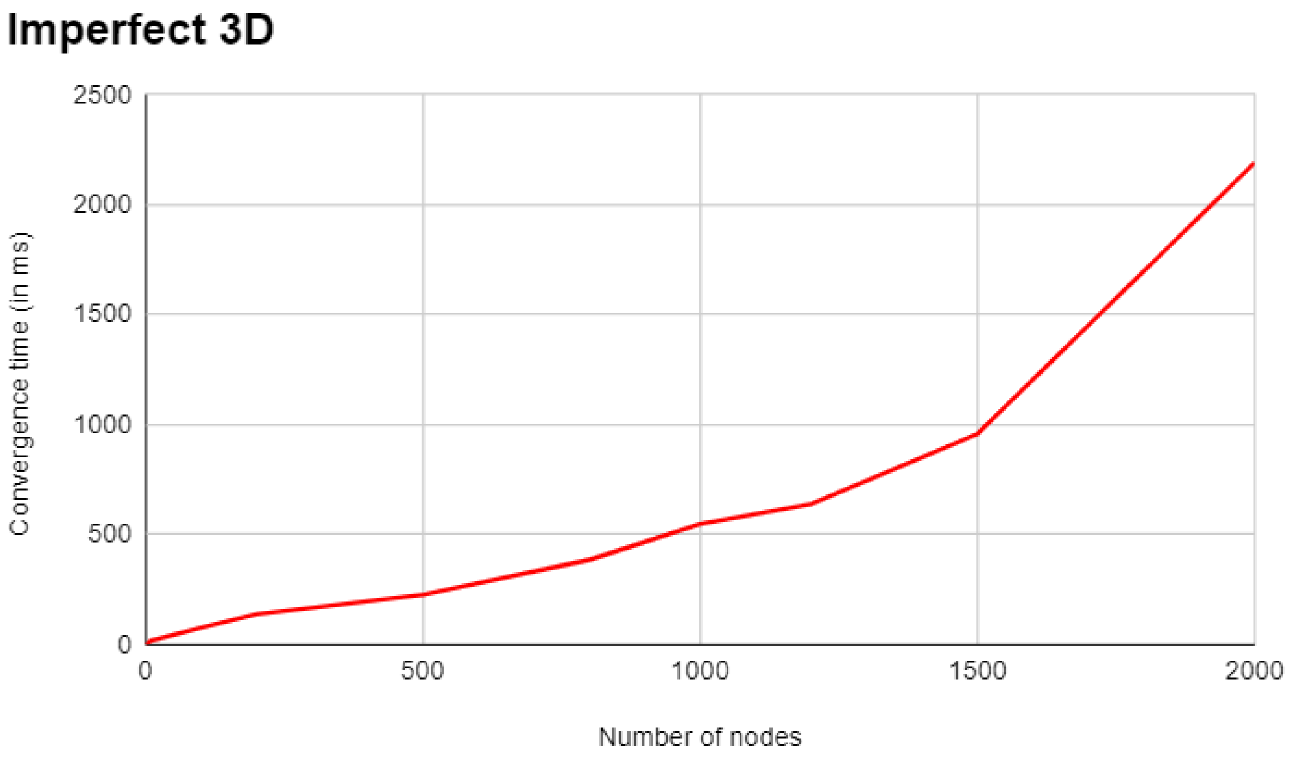


(4) Imperfect 3D Topology - Grid arrangement but one random extra neighbor is selected from the list of all actors.

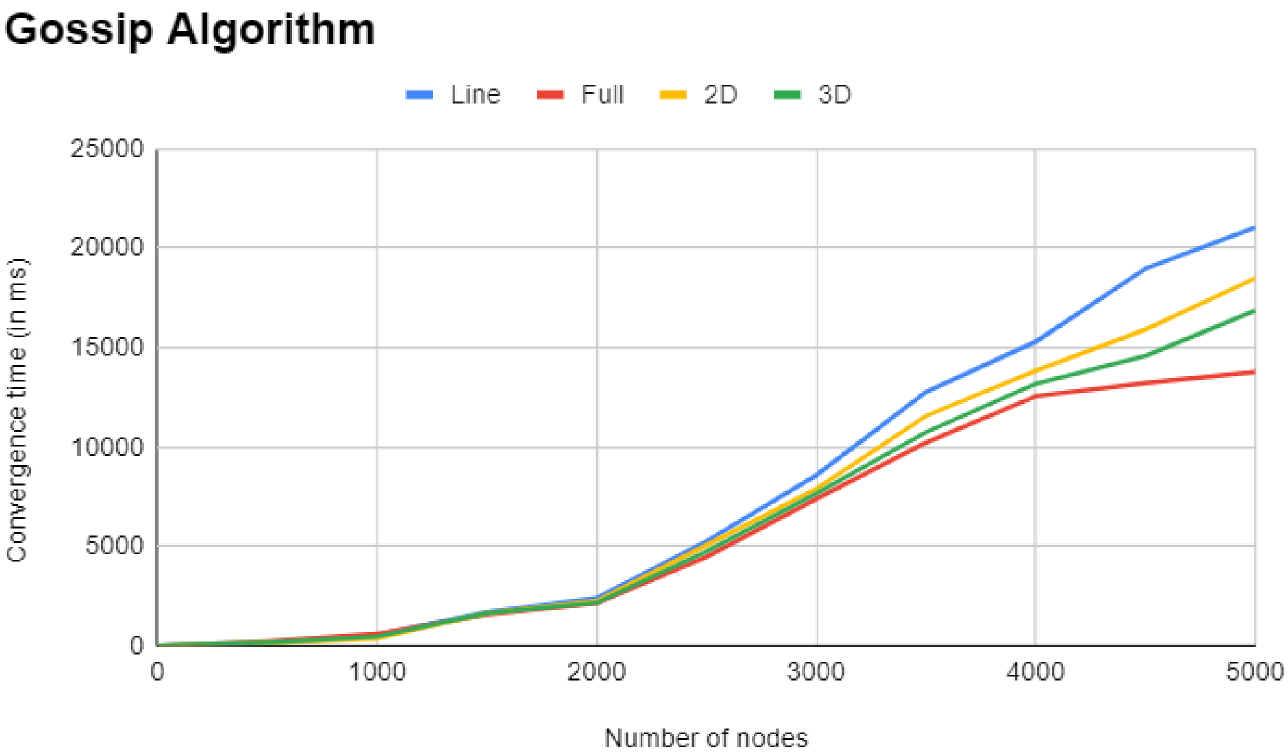
Gossip:



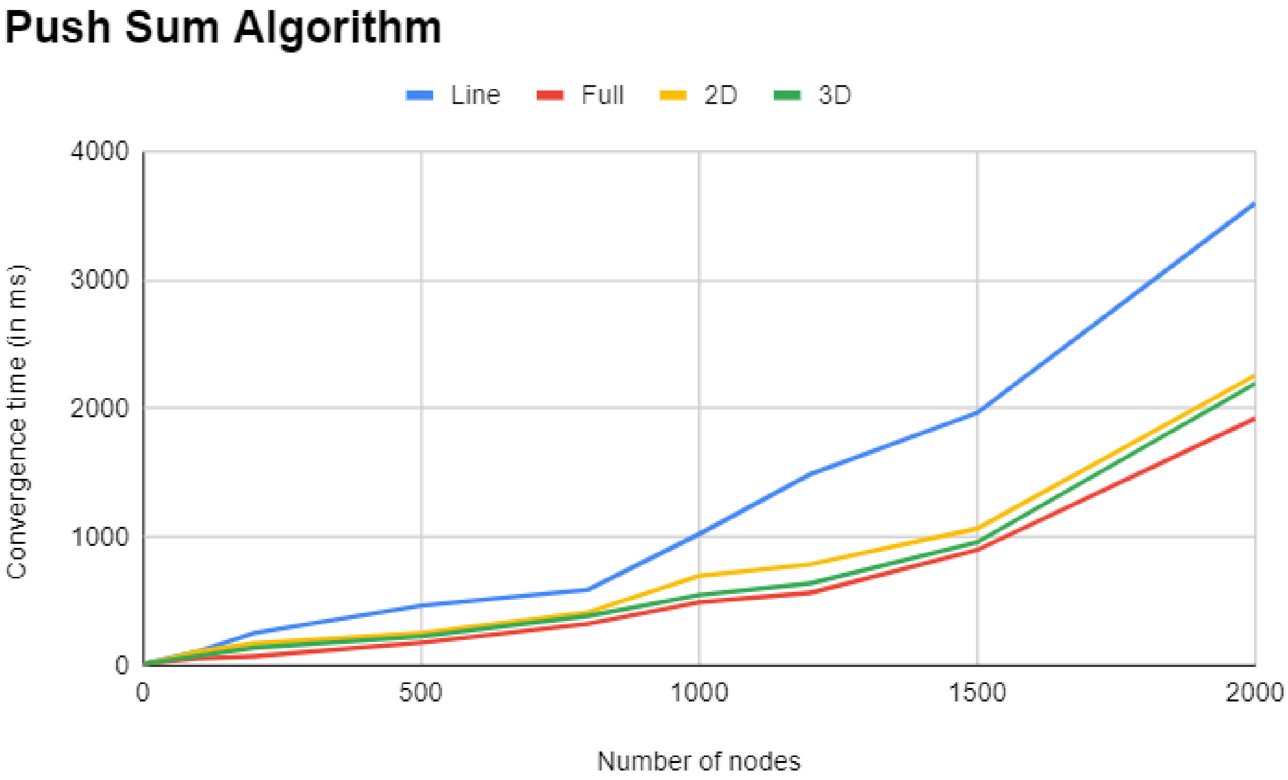
Push Sum:



The below graph plots the convergence times in milliseconds vs number of nodes for different topologies for the gossip algorithm.



The below graphs plot the convergence times of push-sum algorithm vs number of nodes for different topologies.

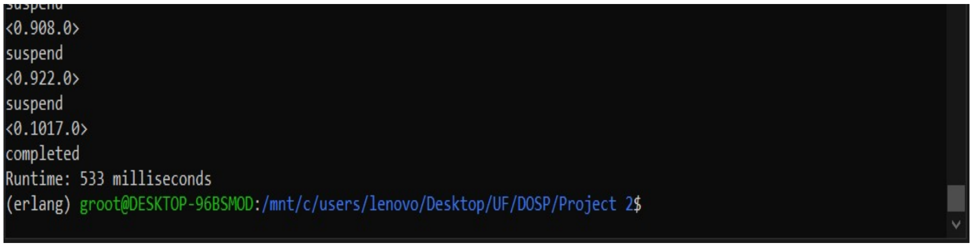


The largest network these topologies run on are given in the below table:

Gossip Algorithm	**Network**	**Push-Sum Algorithm**	**Network**
Line Topology	10000	Line Topology	8000
Full Topology	15000	Full Topology	12000
2D Grid Topology	10000	2D Grid Topology	10000
Imperfect 3D Topology	10000	Imperfect 3D Topology	10000

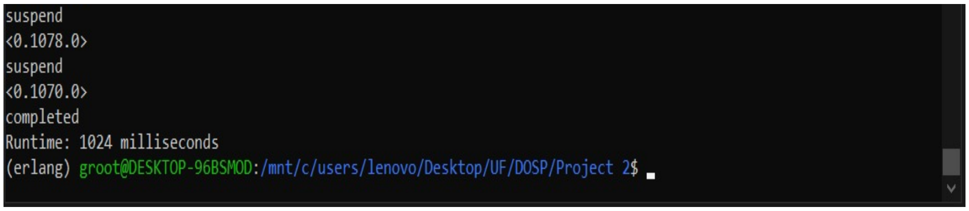
Findings and Analysis-

1) Our observations confirm that line topology is the most inefficient in both Gossip and Push-Sum implementations. In Particular, it takes a large amount of time in the case of the Push-Sum algorithm.



```
suspend
<0.908.0>
suspend
<0.922.0>
suspend
<0.1017.0>
completed
Runtime: 533 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

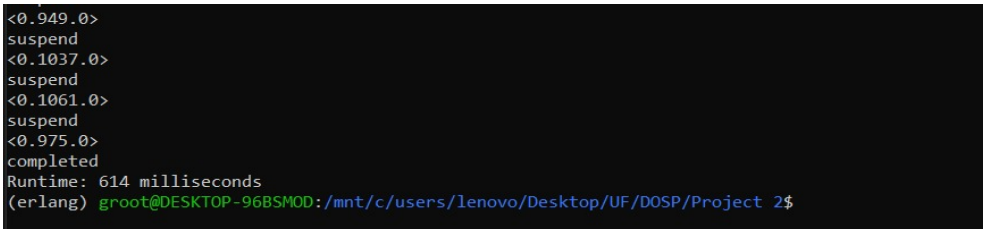
Line topology for gossip algorithm



```
suspend
<0.1078.0>
suspend
<0.1070.0>
completed
Runtime: 1024 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

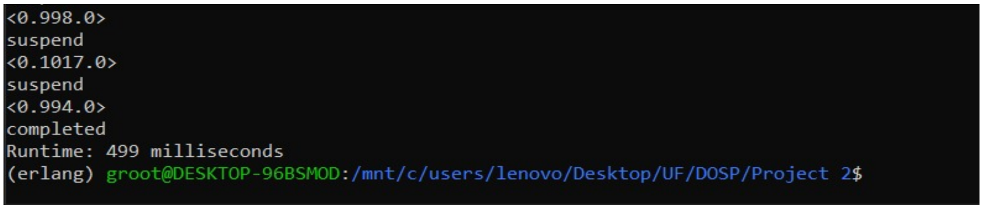
Line topology for push-sum algorithm

2) Full topology comes out to be the most efficient in both Gossip and Push-Sum algorithm, this is because each node is connected to every other node and the gossip spreads much faster compared to every other topology.



```
<0.949.0>
suspend
<0.1037.0>
suspend
<0.1061.0>
suspend
<0.975.0>
completed
Runtime: 614 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

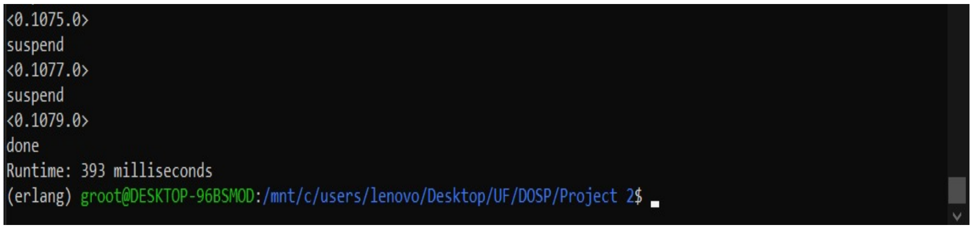
Full topology for gossip algorithm



```
<0.998.0>
suspend
<0.1017.0>
suspend
<0.994.0>
completed
Runtime: 499 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

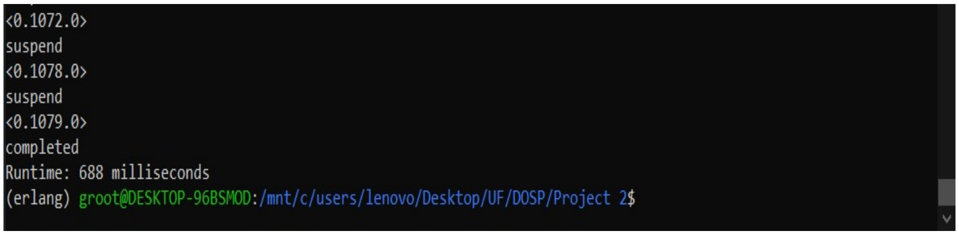
Full topology for push-sum algorithm

3) 2D grid topology improves the performance of line topology as it has more neighbors so the gossip spreads faster. This is true for both cases.



```
<0.1075.0>
suspend
<0.1077.0>
suspend
<0.1079.0>
done
Runtime: 393 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

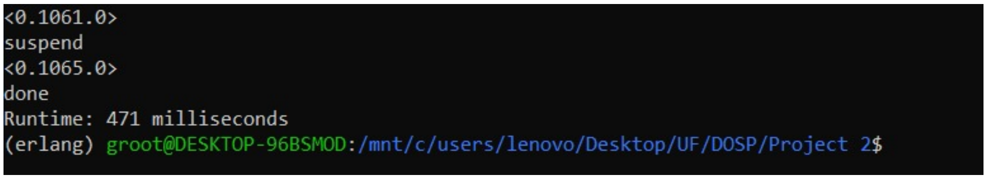
2D grid topology for gossip algorithm



```
<0.1072.0>
suspend
<0.1078.0>
suspend
<0.1079.0>
completed
Runtime: 688 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

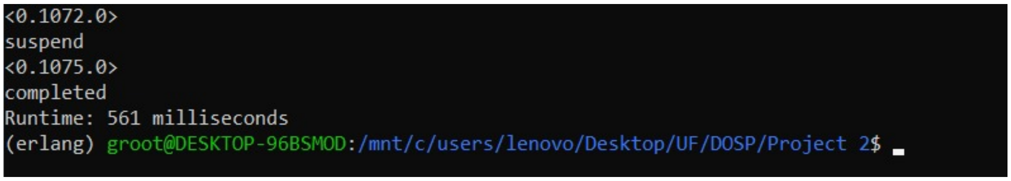
2D grid topology for push-sum algorithm

4) Our findings confirmed that the imperfect 3D grid topology gives better performance than 2D grid topology because of the extra random assignment. This holds for gossip as well as push-sum algorithm.



```
<0.1061.0>
suspend
<0.1065.0>
done
Runtime: 471 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

Imperfect 3D grid topology for gossip algorithm



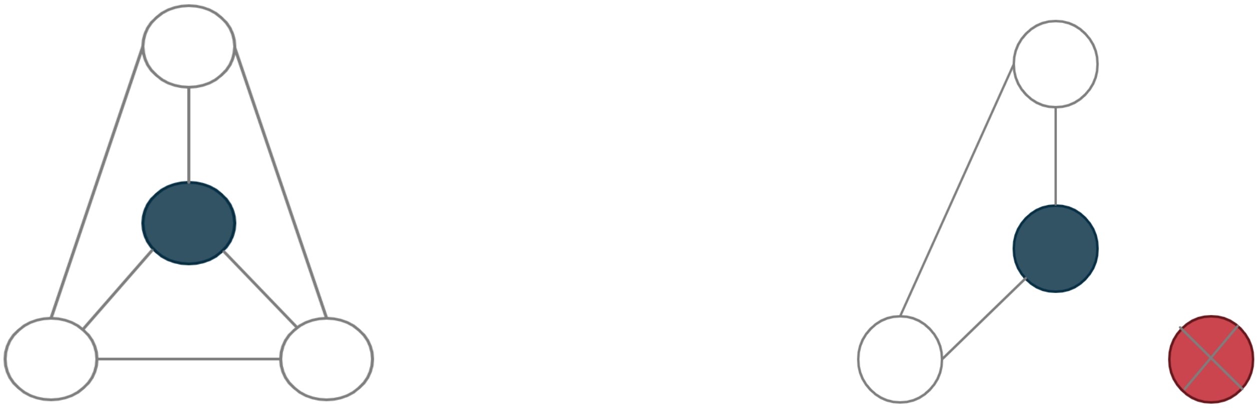
```
<0.1072.0>
suspend
<0.1075.0>
completed
Runtime: 561 milliseconds
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/lenovo/Desktop/UF/DOSP/Project 2$
```

Imperfect 3D grid topology for push-sum algorithm

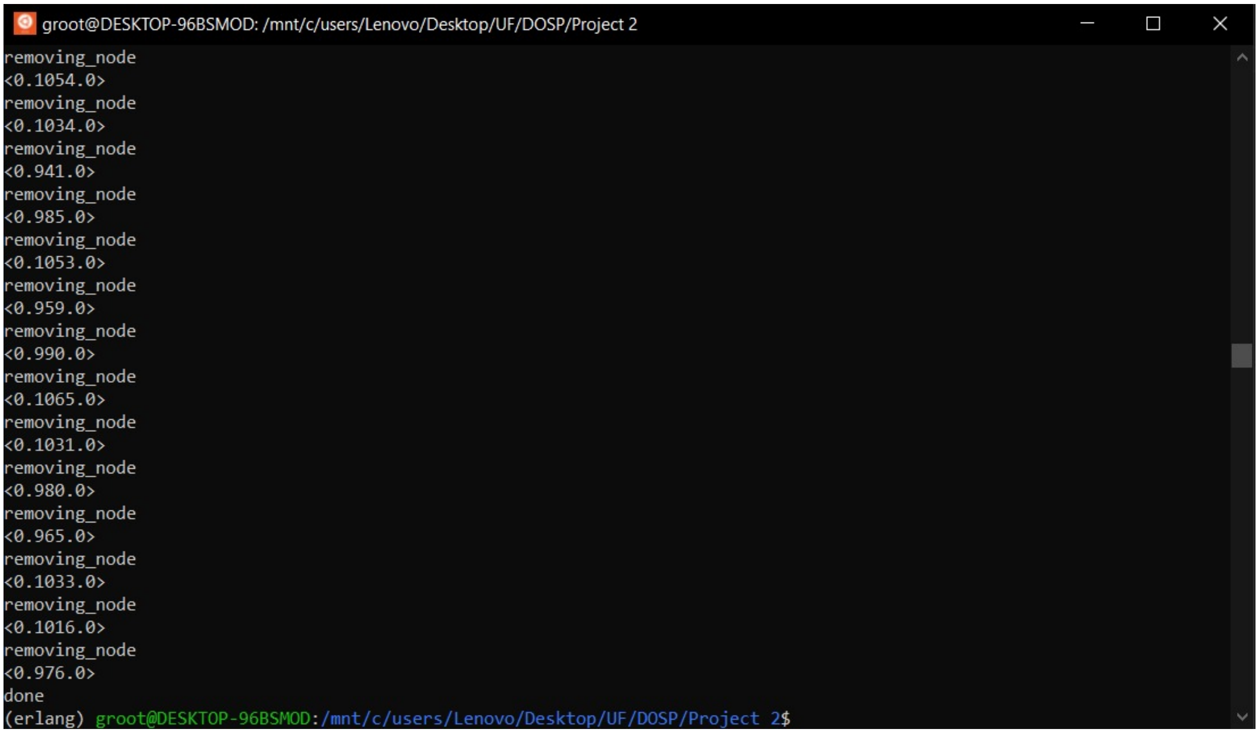
5) The order of convergence was similar in gossip and push-sum algorithm with full topology being better in both cases and line topology being the worst. The reason is simple: in line topology each node is connected only with its left and right neighbors (unless it’s the first or last one) whereas in full topology each node is connected with every other node. Imperfect 3D grid comes out to be better than 2D because of the extra random assignment but both these topologies are in between line and full topologies.

Fault Tolerance:

For fault tolerance we have created a supervisor for full topology. In this every actor is connected to a supervisor. So as soon as a node dies, the supervisor removes it from the list and rearranges the topology. This way the topology won’t break and the dead actor can be removed from the list. In the end supervisor will be left with only one actor, also we can spin up alternate multiple actors with the help of supervisor to instead of removing dead actors to handle load efficiently.



Here the black circle represents the supervisor and the white circle represents the nodes, the supervisor will be connected with each and every node of the topology.


groot@DESKTOP-96BSMOD: /mnt/c/users/Lenovo/Desktop/UF/DOSP/Project 2

```
removing_node
<0.1054.0>
removing_node
<0.1034.0>
removing_node
<0.941.0>
removing_node
<0.985.0>
removing_node
<0.1053.0>
removing_node
<0.959.0>
removing_node
<0.990.0>
removing_node
<0.1065.0>
removing_node
<0.1031.0>
removing_node
<0.980.0>
removing_node
<0.965.0>
removing_node
<0.1033.0>
removing_node
<0.1016.0>
removing_node
<0.976.0>
done
(erlang) groot@DESKTOP-96BSMOD:/mnt/c/users/Lenovo/Desktop/UF/DOSP/Project 2$
```