Distributed Operating Systems Project 2
Gossip Algorithms

Tanvi Jain & Delora Almeida

Problem Definition:
As described in class Gossip type algorithms can be used both for group communication and for aggregate computation. The goal of this project is to determine the convergence of such algorithms through a simulator based on actors written in Erlang. Since actors are fully asynchronous, the particular type of Gossip implemented is the so-called Asynchronous Gossip.

Gossip Algorithm for information propagation:
- Starting: A participant(actor) told/sent a rumor (fact) by the main process
- Step: Each actor selects a random neighbor and tells it the rumor.
- Termination: Each actor keeps track of rumors and how many times he has heard the rumor. It stops transmitting once it has heard the rumor 10 times (10 is arbitrary, you can select other values).

Push-Sum algorithm for sum computation:
- State: Each actor Ai maintains two quantities: s and w. Initially, s = xi = i (that is actor number i has value i, play with other distributions if you so desire) and w = 1.
- Starting: Ask one of the actors to start from the main process.
- Receive: Messages sent and received are pairs of the form (s, w). Upon receiving, an actor should add the received pair to its own corresponding values. Upon receiving, each actor selects a random neighbor and sends it a message.
- Send: When sending a message to another actor, half of s and w is kept by the sending actor, and half is placed in the message.
- Sum Estimate: At any given moment of time, the sum estimate is s/w where s and w are the current values of an actor.
- Termination: If an actor's ratio s/w did not change more than 10^{-10} in 3 consecutive rounds the actor terminates.

Topologies:
The actual network topology plays a critical role in the dissemination speed of Gossip protocols. The topology determines who is considered a neighbor in the above algorithms.
- Full Network: Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.
- 2D Grid: Actors form a 2D grid. The actors can only talk to the grid neighbors
- 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 3D Grid: Grid arrangement but one random other neighbor is selected from the list of all actors (8+1 neighbors).

Input: The input provided (as a command line to your project2) will be of the form:

project2 numNodes topology algorithm

Where, numNodes is the number of actors involved (for 2D-based topologies you can round up until you get a square), topology is one of full, 2D, line, imp2D, the algorithm is one of gossip, push-sum.

Results:

In order to run the program, first compile all the files using:

```
c(project2).
c(node_termination).
c(select_topology_2D).
c(select_topology_imp3D).
c(select_topology_full).
c(select_topology_line).
```

And then enter:

```
project2:main().
project2 <numNodes> <topology> <algorithm>
```

The largest network dealt with Gossip Algorithm and all the four topologies: full, line, 2D and Imperfect 3D is *5000*. Similarly, the largest network dealt with Push-Sum Algorithm and all the four topologies is *1000*.

Gossip Algorithm-

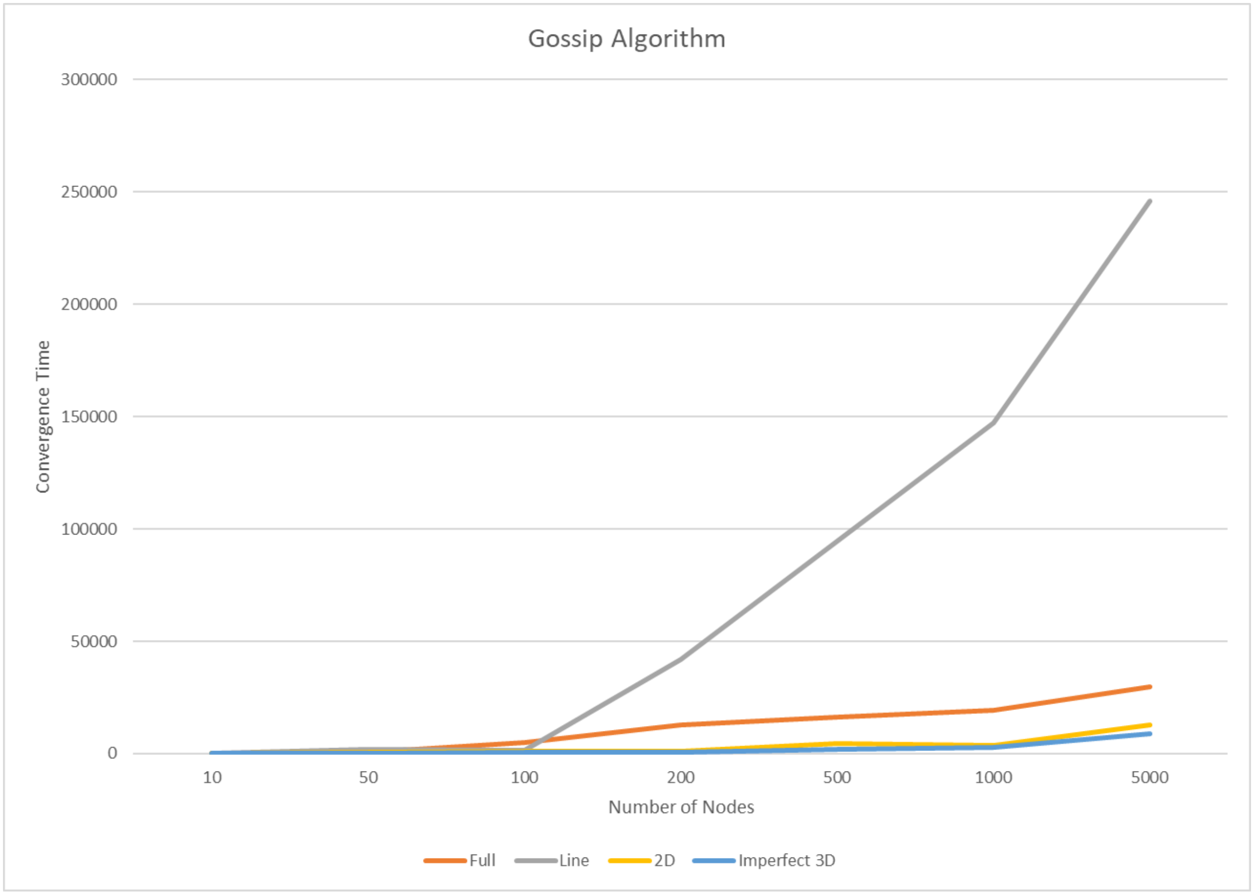
In the **Gossip Protocol**, an actor selects a random neighbor to send the rumor to. The nodes that reach the limit, which is 10 in our case, stop transmitting but participate in the gossip until all the nodes receive the gossip 10 times, else we would not be able to achieve convergence.

Table of executed values for Gossip:

Number of Nodes	**Full Topology**	**Line Topology**	**2D Topology**	**Imperfect 3D Topology**
10	36	11	9	12
50	575	1702	646	232
100	5094	1413	870	498

200	12673	41798	1027	713
500	16354	94650	4534	1840
1000	19384	147264	3504	2549
5000	29774	245984	12890	8977

Graph for the Gossip Algorithm:



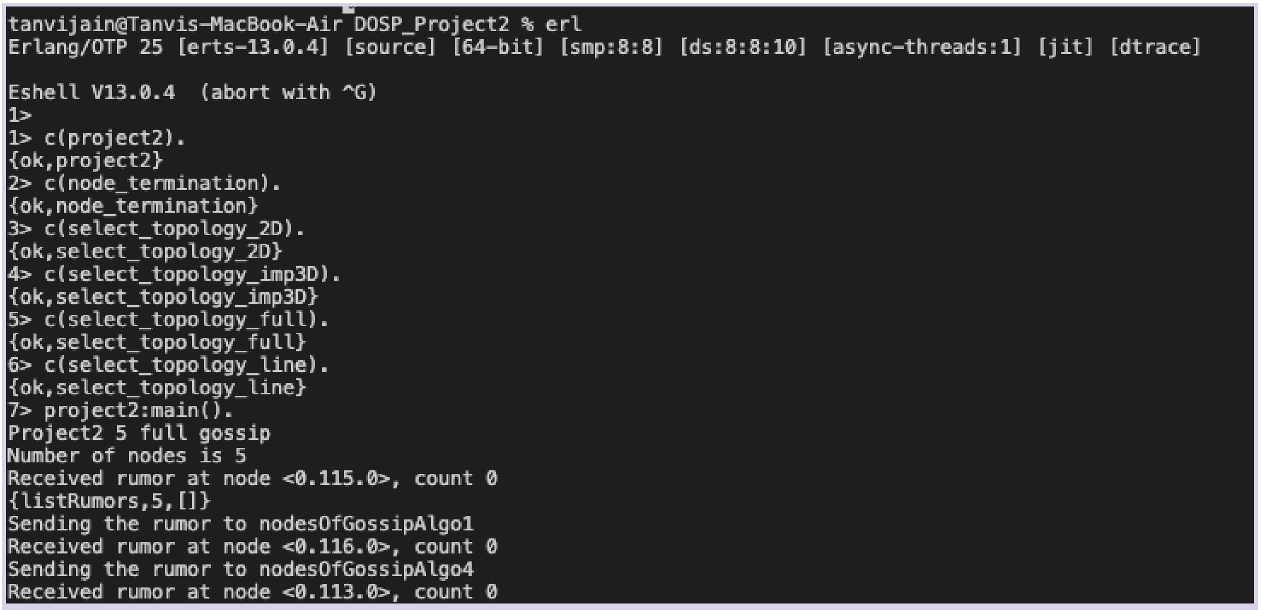
Observations and Findings:

- Line Topology is the slowest to converge. This is probably because it only has two neighbors.
- Full Topology works well for a smaller number of nodes but has a higher running time for a larger number of nodes. On the contrary, it takes the longest time to converge. This is because there are many neighbors and the s/w ratios may pile up leading to some nodes not converging because they are waiting on others. We thought that this would be the fastest, however it needs to spread the rumor to a huge number of nodes. So, as the number of nodes increases, the convergence time increases.

- If the number of nodes in a 2D grid are a perfect square, then the convergence time is lesser.
- Imp3D and 2D have similar convergence times. However, as nodes increase imp3D has a better convergence time.
- Hence, according to our observations, line topology has the worst convergence time and imp3D has the best.

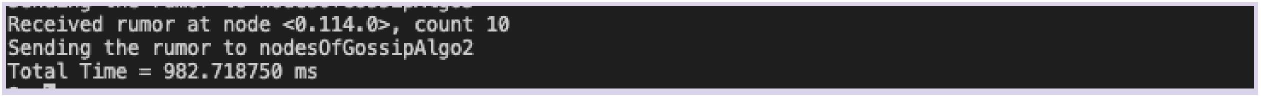
Examples:

"> *project2 10 full gossip*"



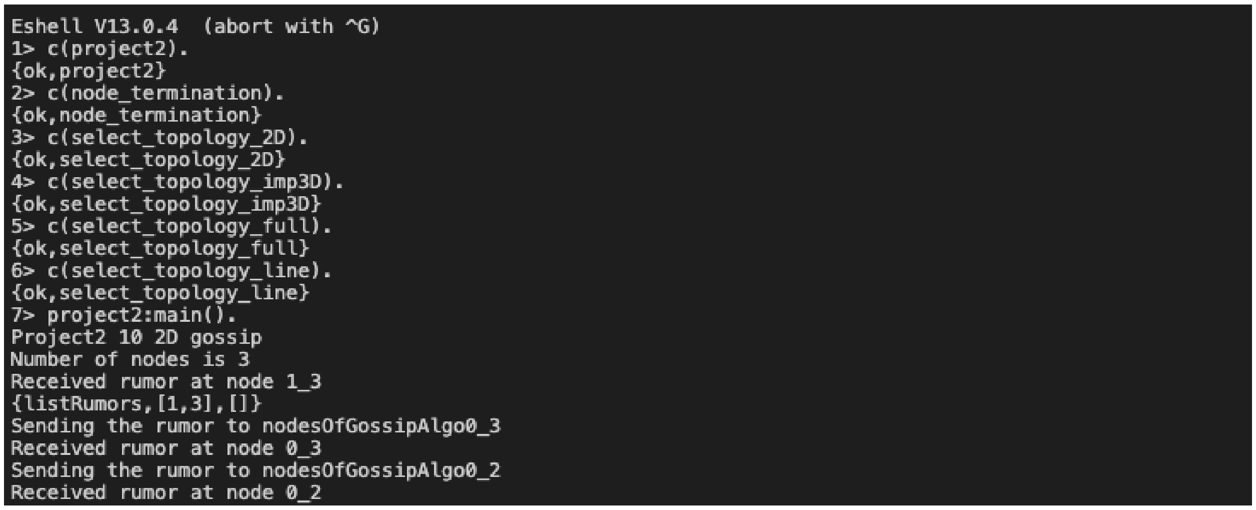
```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1>
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 5 full gossip
Number of nodes is 5
Received rumor at node <0.115.0>, count 0
{listRumors,5,[]}
Sending the rumor to nodesOfGossipAlgo1
Received rumor at node <0.116.0>, count 0
Sending the rumor to nodesOfGossipAlgo4
Received rumor at node <0.113.0>, count 0
```

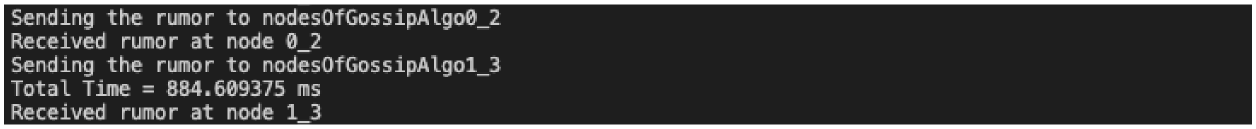


```
Received rumor at node <0.114.0>, count 10
Sending the rumor to nodesOfGossipAlgo2
Total Time = 982.718750 ms
```

"> *project2 10 2D gossip*"

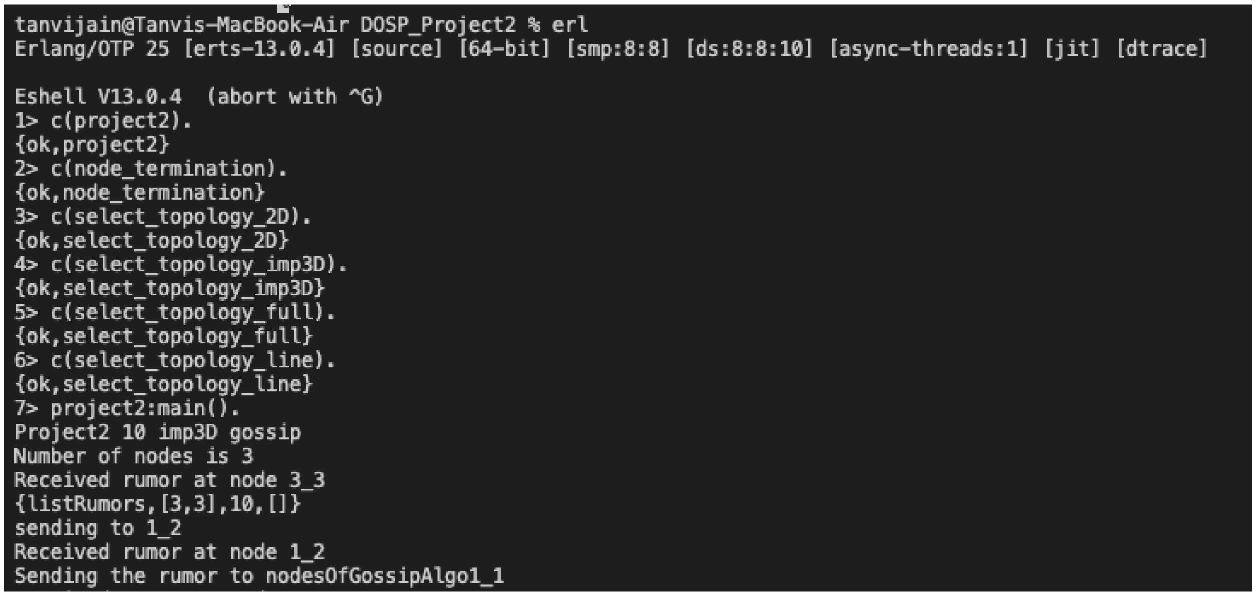


```
Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 2D gossip
Number of nodes is 3
Received rumor at node 1_3
{listRumors,[1,3],[]}
Sending the rumor to nodesOfGossipAlgo0_3
Received rumor at node 0_3
Sending the rumor to nodesOfGossipAlgo0_2
Received rumor at node 0_2
```



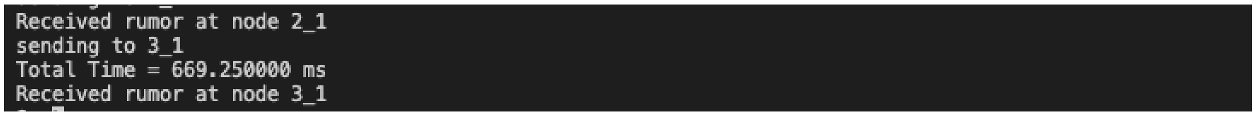
```
Sending the rumor to nodesOfGossipAlgo0_2
Received rumor at node 0_2
Sending the rumor to nodesOfGossipAlgo1_3
Total Time = 884.609375 ms
Received rumor at node 1_3
```

“> ***project2 10 imp3D gossip***”



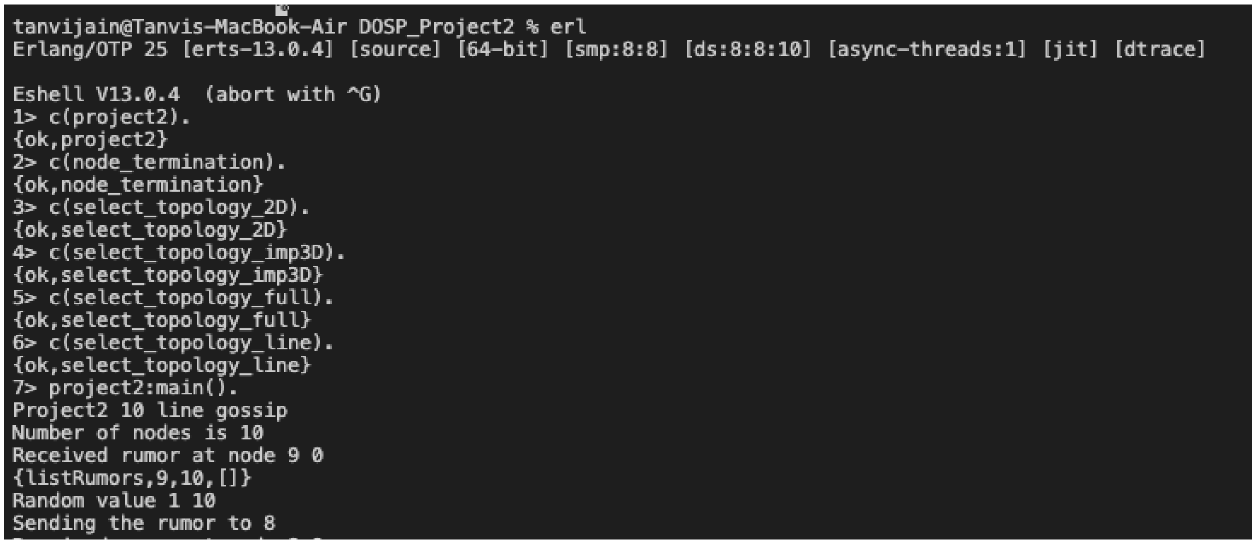
```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 imp3D gossip
Number of nodes is 3
Received rumor at node 3_3
{listRumors,[3,3],10,[]}
sending to 1_2
Received rumor at node 1_2
Sending the rumor to nodesOfGossipAlgo1_1
```



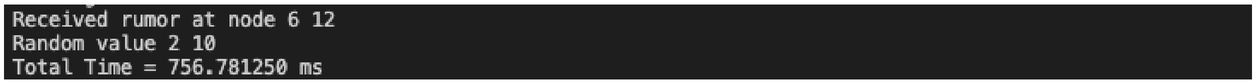
```
Received rumor at node 2_1
sending to 3_1
Total Time = 669.250000 ms
Received rumor at node 3_1
```

“> ***project2 10 line gossip***”



```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 line gossip
Number of nodes is 10
Received rumor at node 9 0
{listRumors,9,10,[]}
Random value 1 10
Sending the rumor to 8
```



```
Received rumor at node 6 12
Random value 2 10
Total Time = 756.781250 ms
```

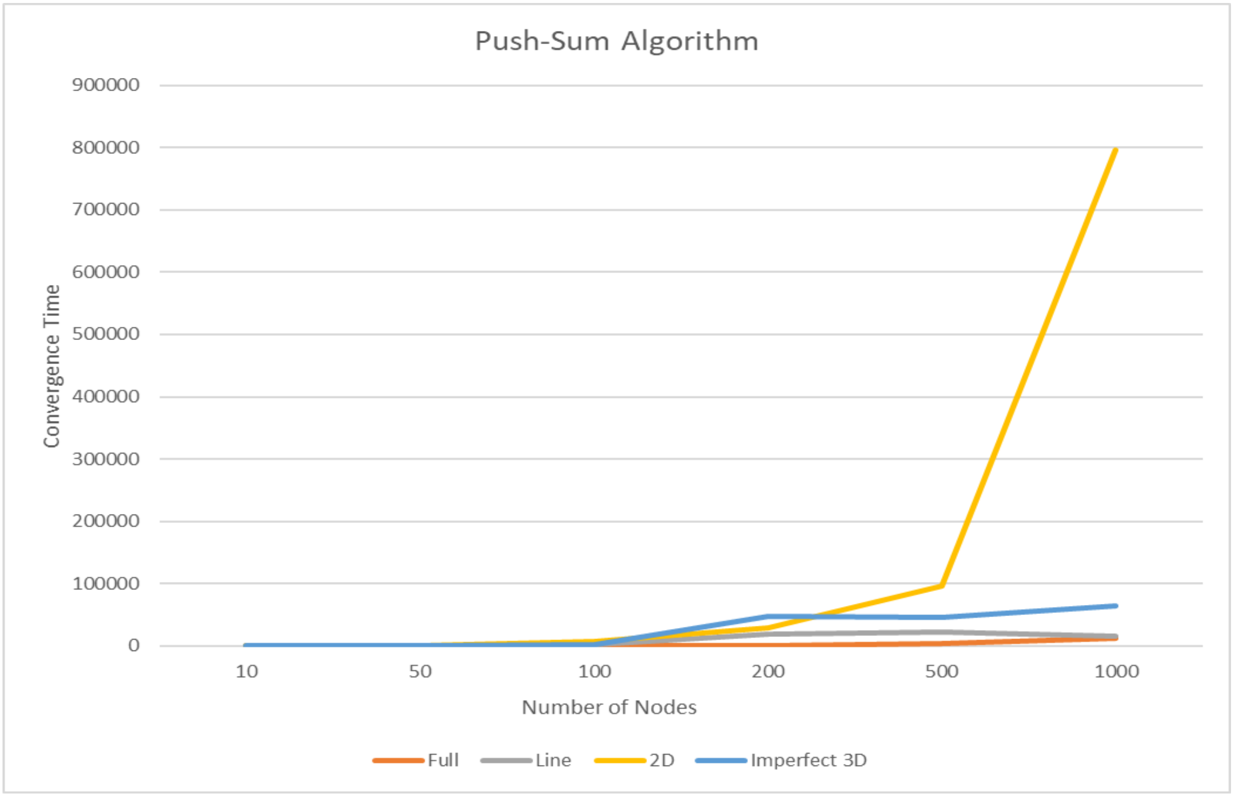
Push-Sum Algorithm-

In the **Push-Sum protocol**, if the ratio of s/w does not change for more than 10^{-10} for 3 consecutive rounds, it is terminated.

Table of executed values for Push-Sum:

Number of Nodes	Full Topology	Line Topology	2D Topology	Imperfect 3D Topology
10	5	10	17	13
50	271	785	821	647
100	674	1653	7829	2905
200	1362	18743	29017	47858
500	4126	21854	96738	45620
1000	12736	15673	796547	63789

Graph for the Push-Sum Algorithm:

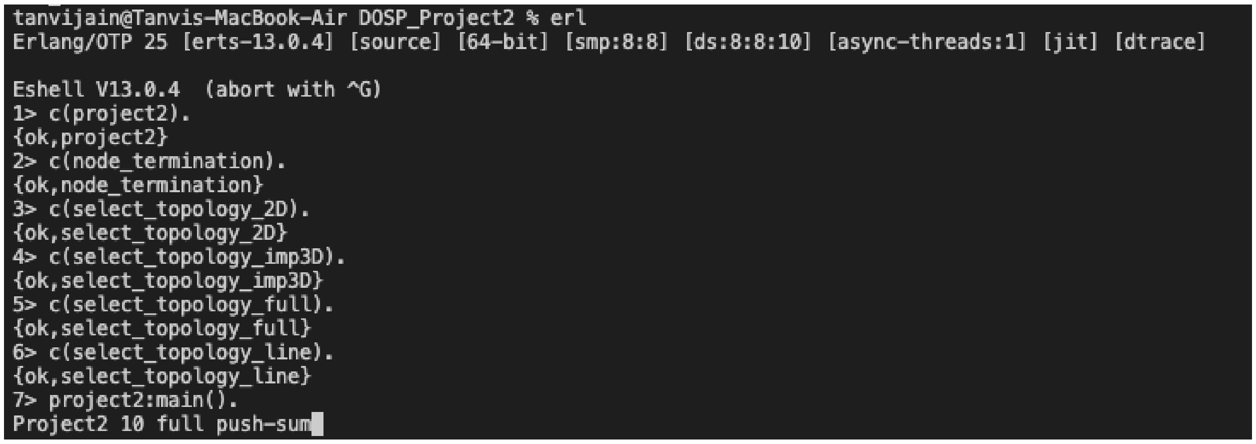


Observations and Findings:

- Push-sum initially has better convergence times comparatively mostly because we divide s/2 and w/2 before propagating the message.
- We see that full topology would be the fastest as we can spread the rumor quickly. Since, there are many neighbors and the s/w ratios make work faster.
- Imperfect 3D initially worked well but with the increase in number of nodes, it took more time to converge than line and full topologies.
- 2D is probably the slowest as the rumor needs to spread in all the directions of grid values and the neighbors are not sequential.
- We see that full has the best convergence time and 2D has the worst convergence time.

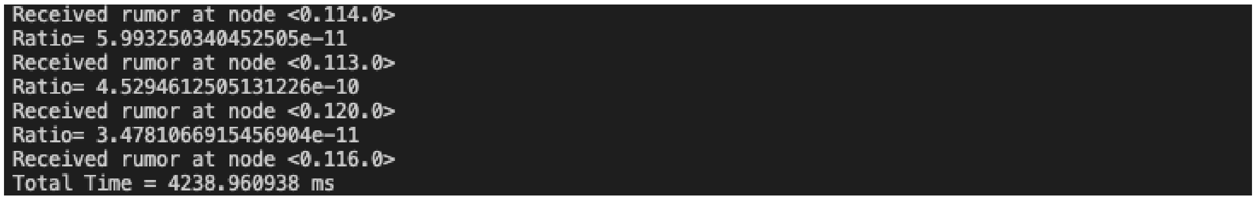
Examples:

“> *project2 10 full push-sum”*



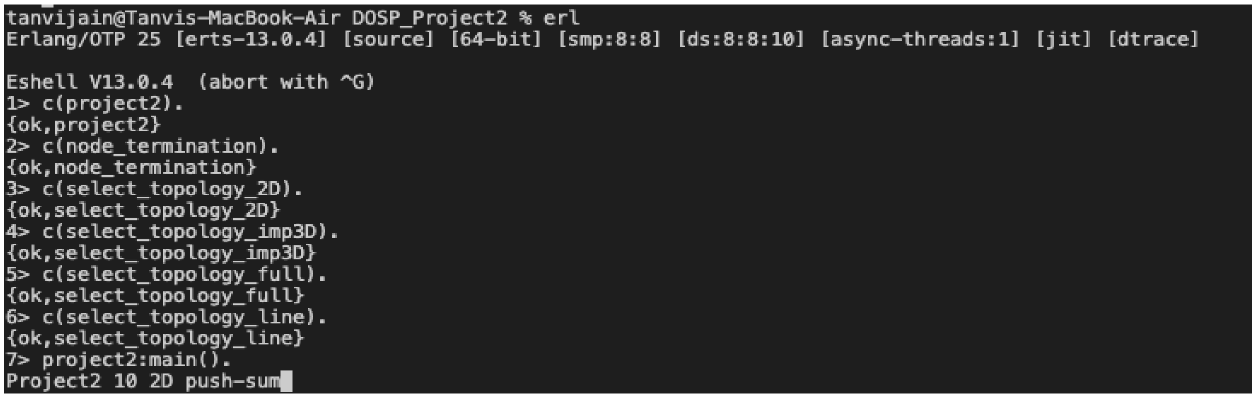
```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 full push-sum
```



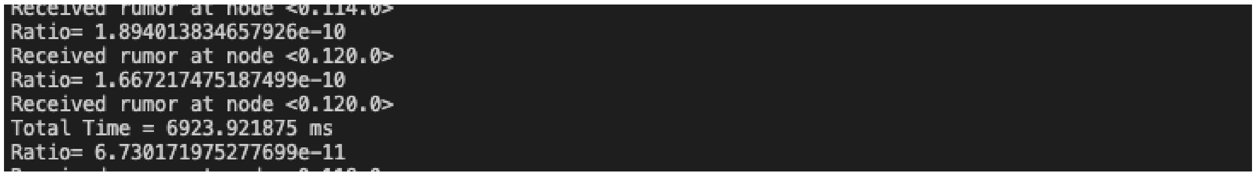
```
Received rumor at node <0.114.0>
Ratio= 5.993250340452505e-11
Received rumor at node <0.113.0>
Ratio= 4.5294612505131226e-10
Received rumor at node <0.120.0>
Ratio= 3.4781066915456904e-11
Received rumor at node <0.116.0>
Total Time = 4238.960938 ms
```

“> *project2 10 2D push-sum”*



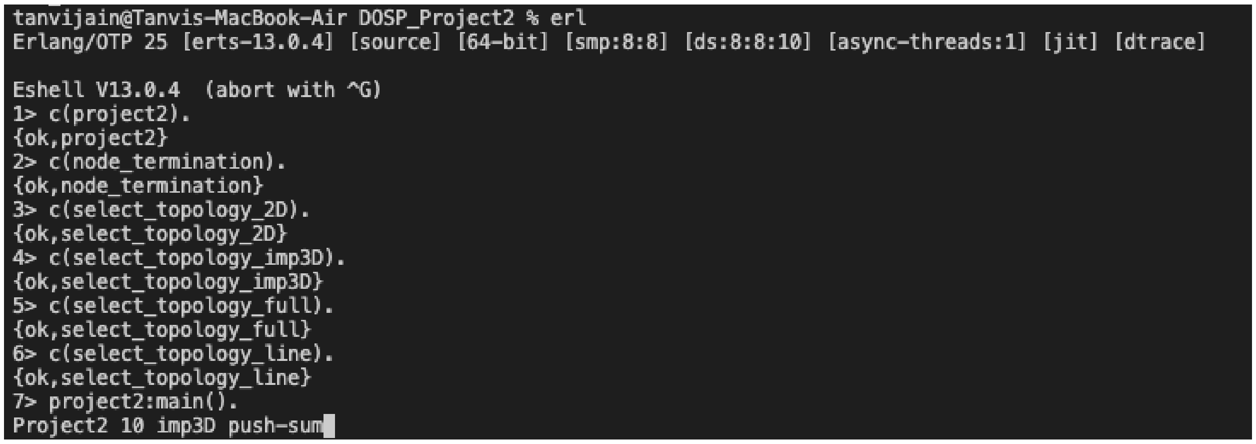
```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 2D push-sum
```



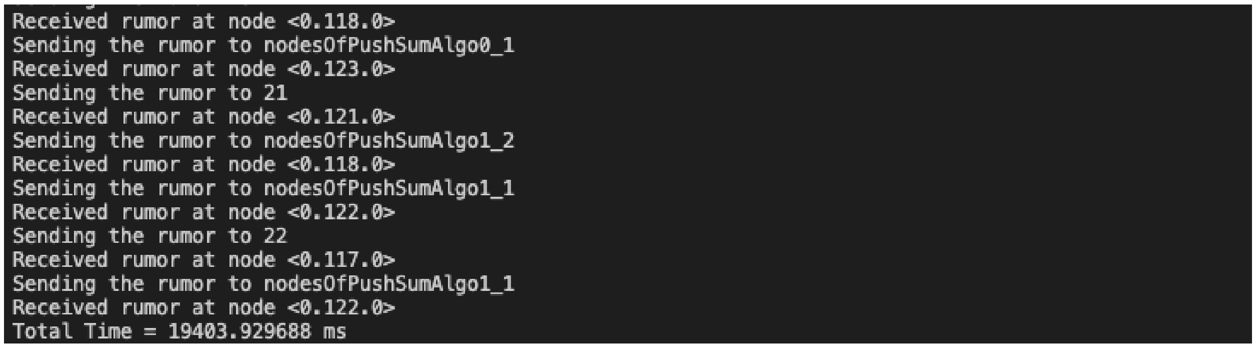
```
Received rumor at node <0.114.0>
Ratio= 1.894013834657926e-10
Received rumor at node <0.120.0>
Ratio= 1.667217475187499e-10
Received rumor at node <0.120.0>
Total Time = 6923.921875 ms
Ratio= 6.730171975277699e-11
```

"> ***project2 10 imp3D push-sum***"



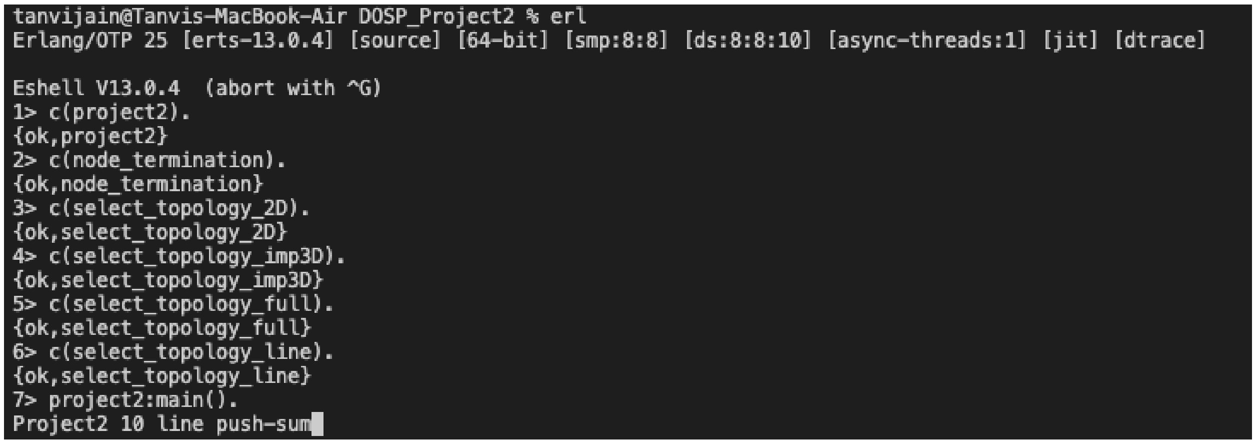
```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 imp3D push-sum
```



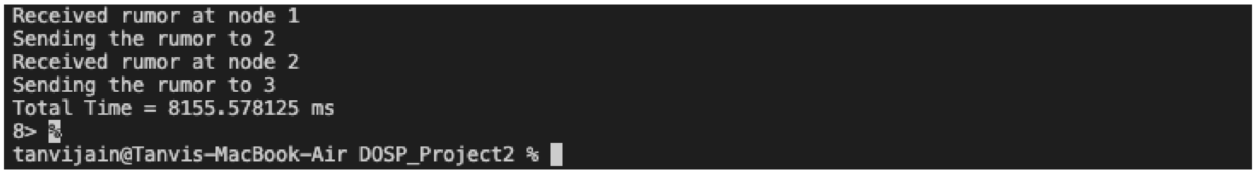
```
Received rumor at node <0.118.0>
Sending the rumor to nodesOfPushSumAlgo0_1
Received rumor at node <0.123.0>
Sending the rumor to 21
Received rumor at node <0.121.0>
Sending the rumor to nodesOfPushSumAlgo1_2
Received rumor at node <0.118.0>
Sending the rumor to nodesOfPushSumAlgo1_1
Received rumor at node <0.122.0>
Sending the rumor to 22
Received rumor at node <0.117.0>
Sending the rumor to nodesOfPushSumAlgo1_1
Received rumor at node <0.122.0>
Total Time = 19403.929688 ms
```

"> ***project2 10 line push-sum***"



```
tanvijain@Tanvis-MacBook-Air DOSP_Project2 % erl
Erlang/OTP 25 [erts-13.0.4] [source] [64-bit] [smp:8:8] [ds:8:8:10] [async-threads:1] [jit] [dtrace]

Eshell V13.0.4  (abort with ^G)
1> c(project2).
{ok,project2}
2> c(node_termination).
{ok,node_termination}
3> c(select_topology_2D).
{ok,select_topology_2D}
4> c(select_topology_imp3D).
{ok,select_topology_imp3D}
5> c(select_topology_full).
{ok,select_topology_full}
6> c(select_topology_line).
{ok,select_topology_line}
7> project2:main().
Project2 10 line push-sum
```



```
Received rumor at node 1
Sending the rumor to 2
Received rumor at node 2
Sending the rumor to 3
Total Time = 8155.578125 ms
8> %
tanvijain@Tanvis-MacBook-Air DOSP_Project2 %
```