Project Report

❖ Group Members

- Name: Smrati Pandey
  UFID: ****-****
- Name: Sachit Verma
  UFID: ****-****

❖ Topologies implemented:

- Full Network: Every actor is a neighbor of all other actors. That is, every actor can talk directly to any other actor.
- 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).
- Random 2D Grid: Actors are randomly position at x, y coordinates on a [0-1.0] x [0-1.0] square. Two actors are connected if they are within .1 distance to other actors.
- 3D torus Grid: Actors form a 3D grid. The actors can only talk to the grid neighbors. And, the actors on outer surface are connected to other actors on opposite side, such that degree of each actor is 6.
- Honeycomb: Actors are arranged in form of hexagons. Two actors are connected if they are connected to each other. Each actor has maximum degree 3.
- Honeycomb with a random neighbor: Actors are arranged in form of hexagons (Similar to Honeycomb). The only difference is that every node has one extra connection to a random node in the entire network.

❖ Algorithms Implemented

Gossip Algorithm for information propagation

Our model for Gossip algorithm selects the random participant from the method and starts sending the rumor to its random neighbor. Each actor keeps the track of the number of times it is going to receive the rumor and keeps sending the message to one of its random neighbors. Our algorithm converges when at least 90 percent of the actors involved in sending the rumor have heard the rumor at least one time. The time taken for the propagation is displayed for the same.

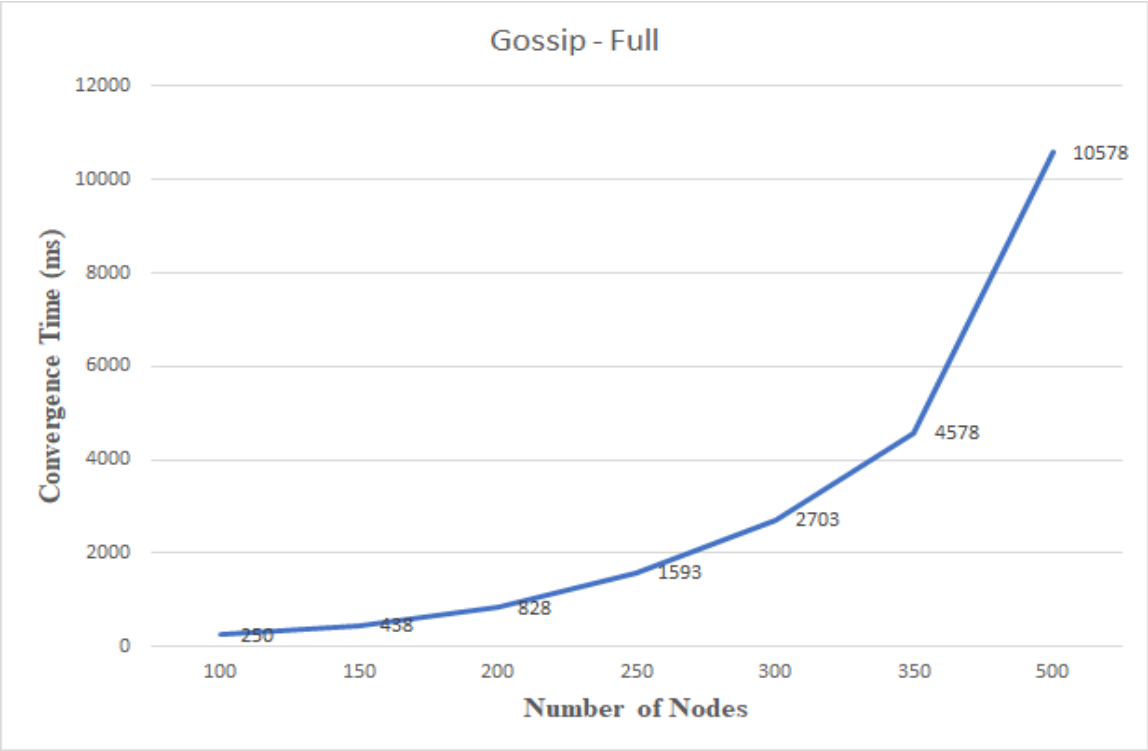
Push-Sum algorithm for sum computation:

- State: Each actor Ai maintains two quantities: s and w. Initially, s = xi = i (that is actor number i has value i) and w = 1.
- Starting: One actor is chosen randomly which starts from the main process.
- Receive: Messages sent and received are pairs of the form (s, w). Upon receiving, an actor adds received pair to its own corresponding values. Then, 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 ratio s/w did not change more than 10 ^ -10 in 3 consecutive rounds the actor terminates. Convergence is achieved when 90% of the actors have been terminated

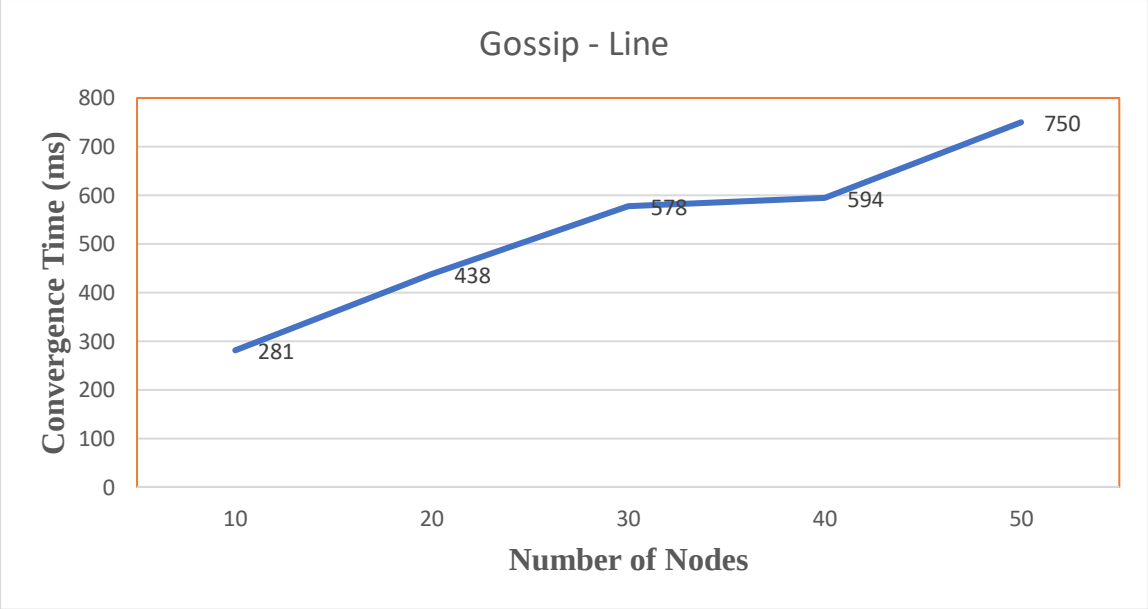
❖ Observation for Gossip algorithm for various topologies:

We have implemented a graph of Convergence Time (milliseconds) vs Number of Nodes.

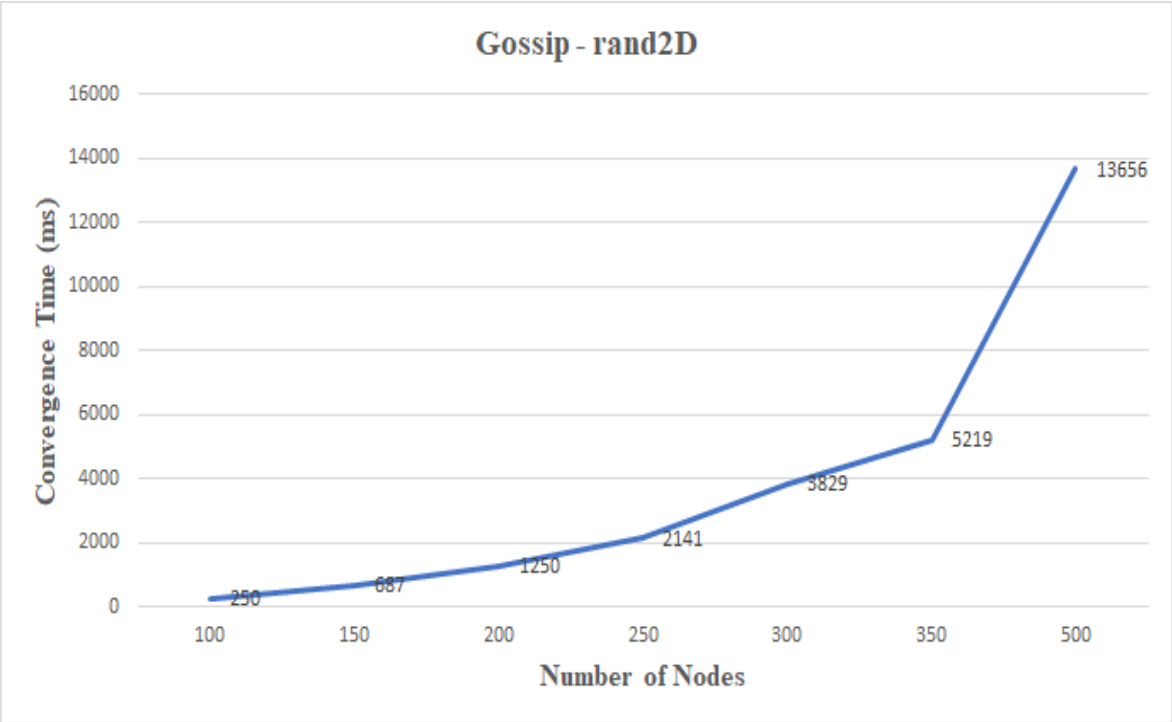
- Full Network: Our implementation of Gossip algorithm supports maximum of 500 nodes and is converged with time of around 10578 milliseconds for Full Network Topology.



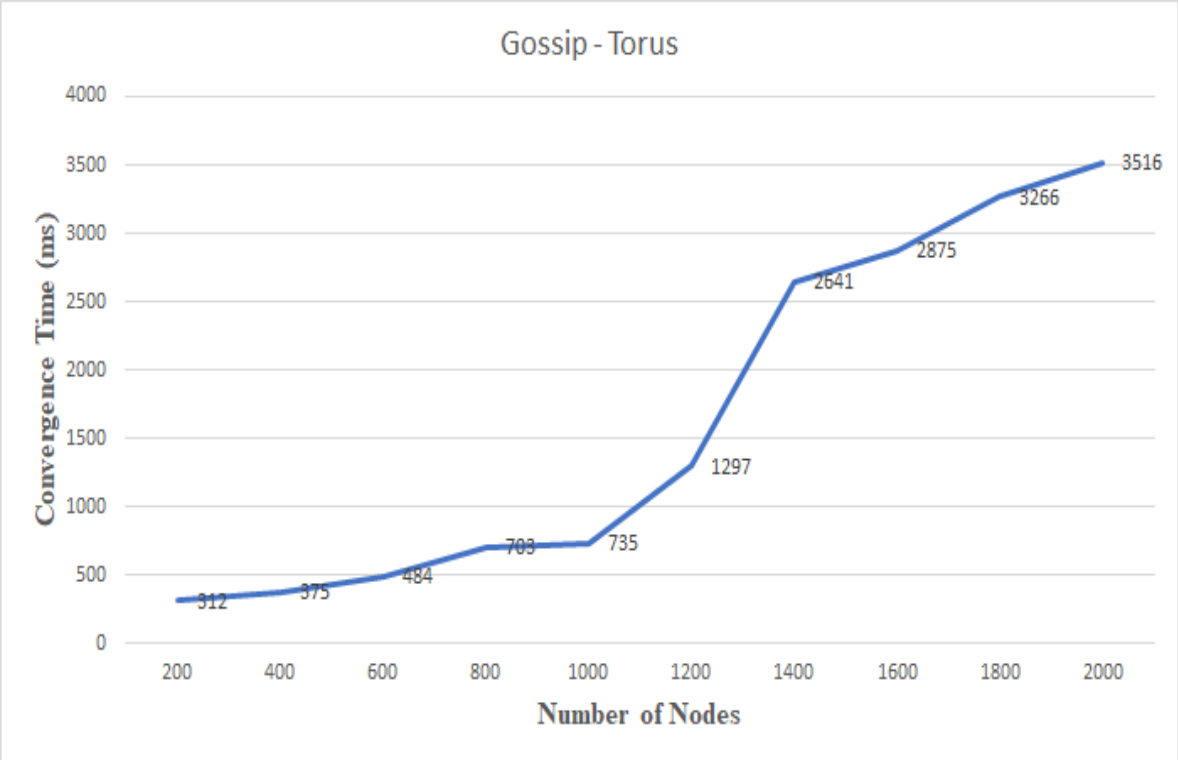
- Line: We tried converging higher number of nodes with our implementation of Gossip algorithm but we were able to calculate the convergence time of up to 50 nodes as sometimes convergence is not at all achieved even for the smallest number of nodes as the node selection is random in the topology.



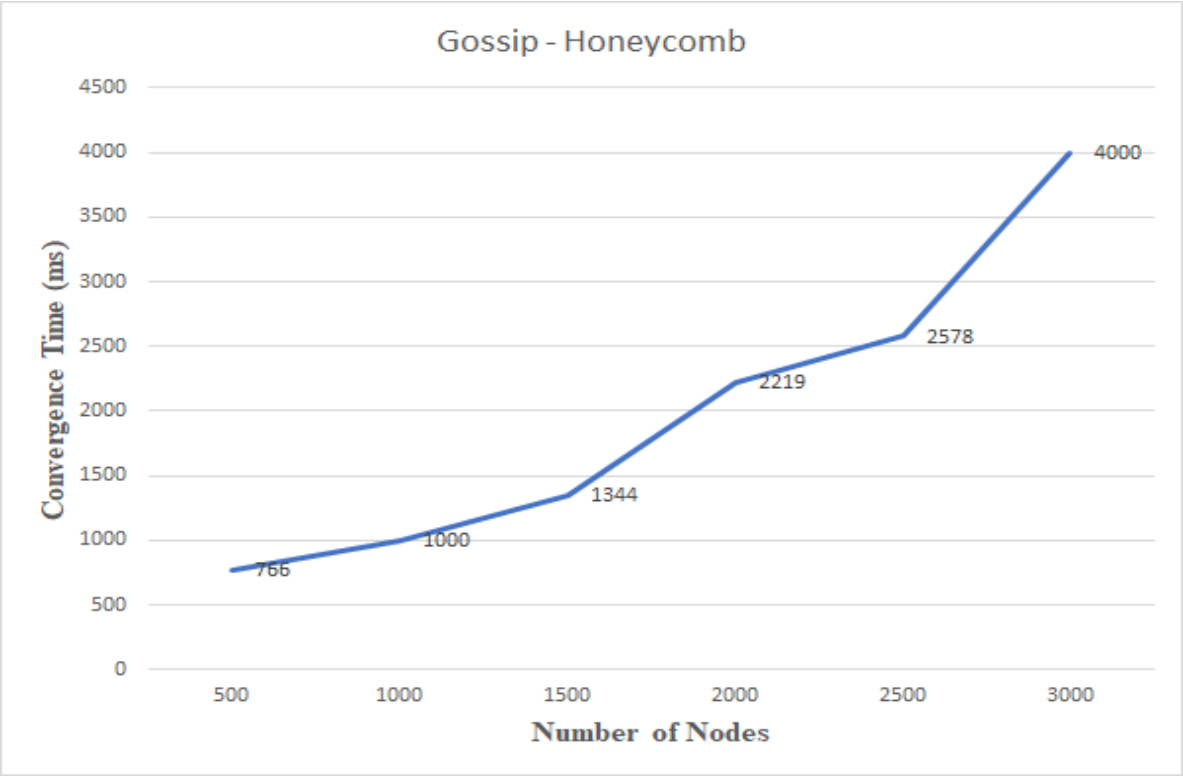
- Random 2D Grid: Our implementation of Gossip algorithm supports maximum of 500 nodes and is converged with time of around 13656 milliseconds for Random 2D Grid.



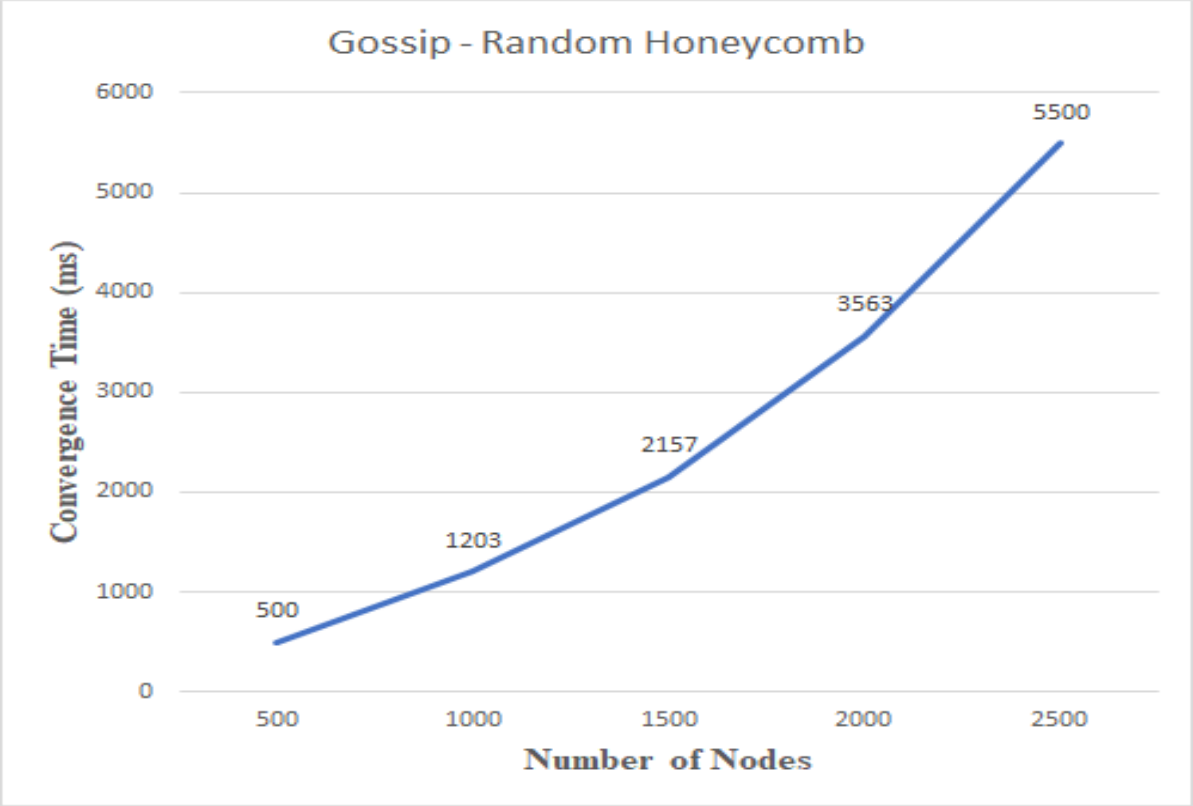
- 3D Torus Grid: Our implementation of Gossip algorithm supports maximum of 2000 nodes and is converged with time of around 3516 milliseconds for 3D torus Grid Topology.



- Honeycomb: Our implementation of Gossip algorithm supports maximum of 3000 nodes and is converged with time of around 4000 milliseconds for Honeycomb topology.



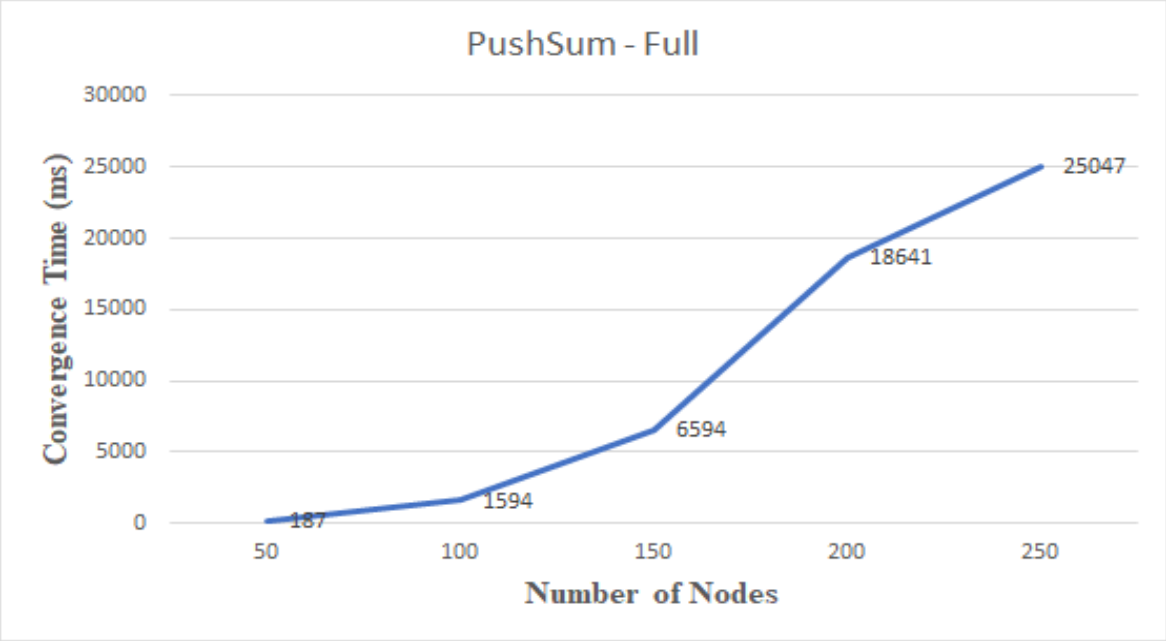
- Honeycomb with a random neighbor: Our implementation of Gossip algorithm supports maximum of 2500 nodes and is converged with time of around 5500 milliseconds for Random Honeycomb topology.



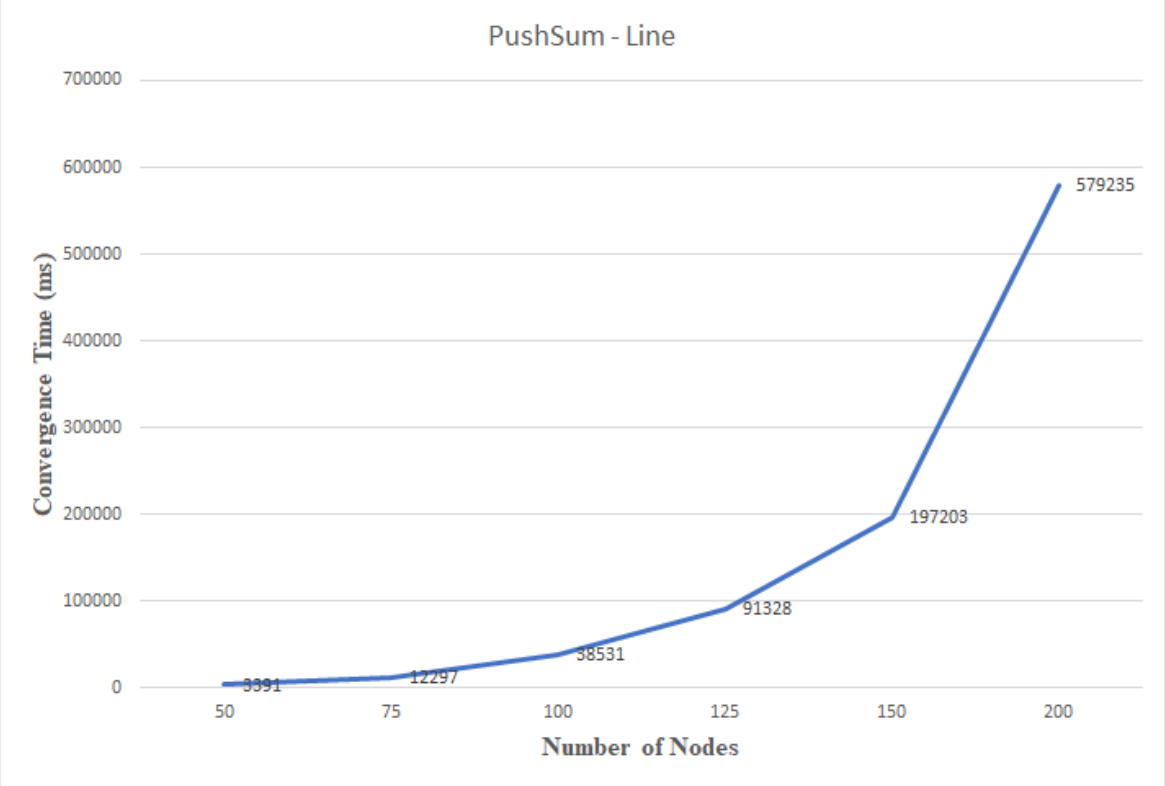
❖ Observation for Push Sum algorithm for various topologies:

We have implemented a graph of Convergence Time (milliseconds) vs Number of Nodes.

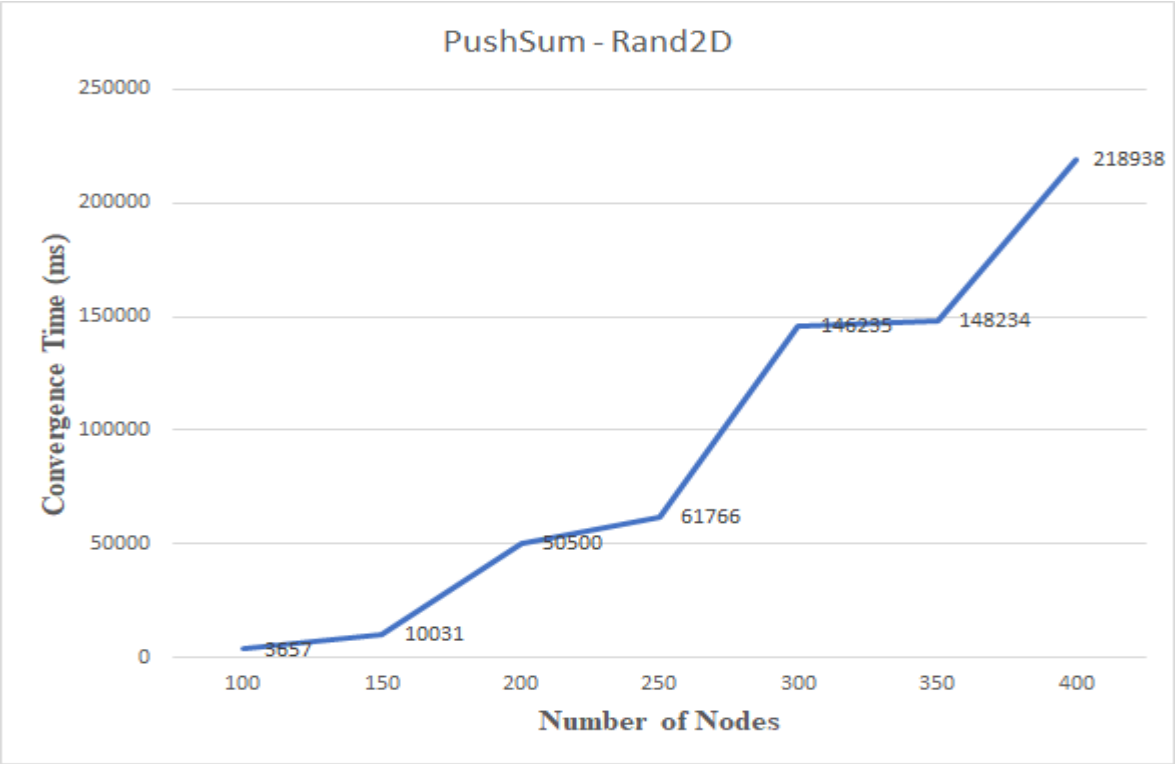
- Full Network: Our implementation of Push Sum algorithm supports maximum of 250 nodes and is converged with time of around 25047 milliseconds for Full Network Topology.



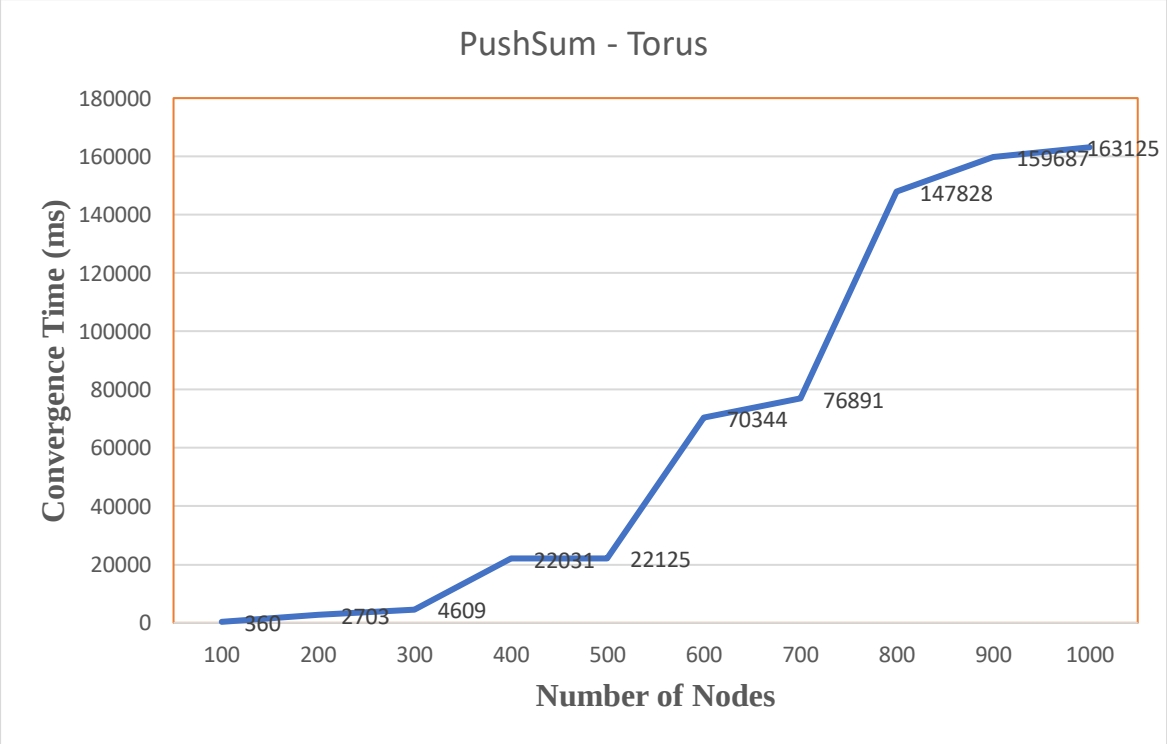
- Line: Our implementation of Push Sum algorithm supports maximum of 200 nodes and is converged with time of around 579235 milliseconds for Line Topology.



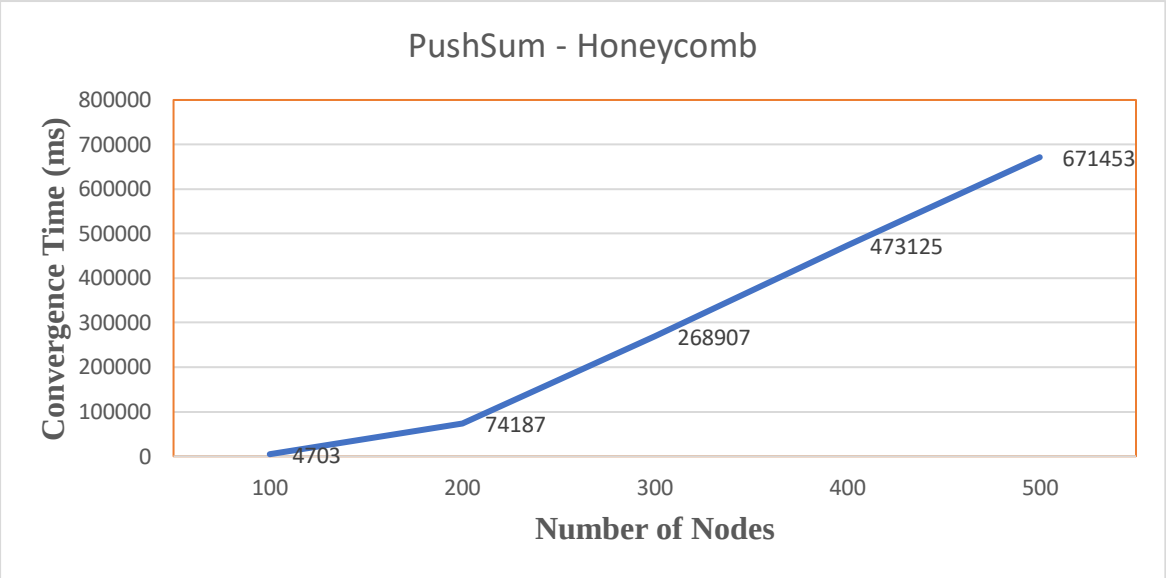
- Random 2D Grid: Our implementation of Push Sum algorithm supports maximum of 400 nodes and is converged with time of around 218938 milliseconds for Random 2D Grid.



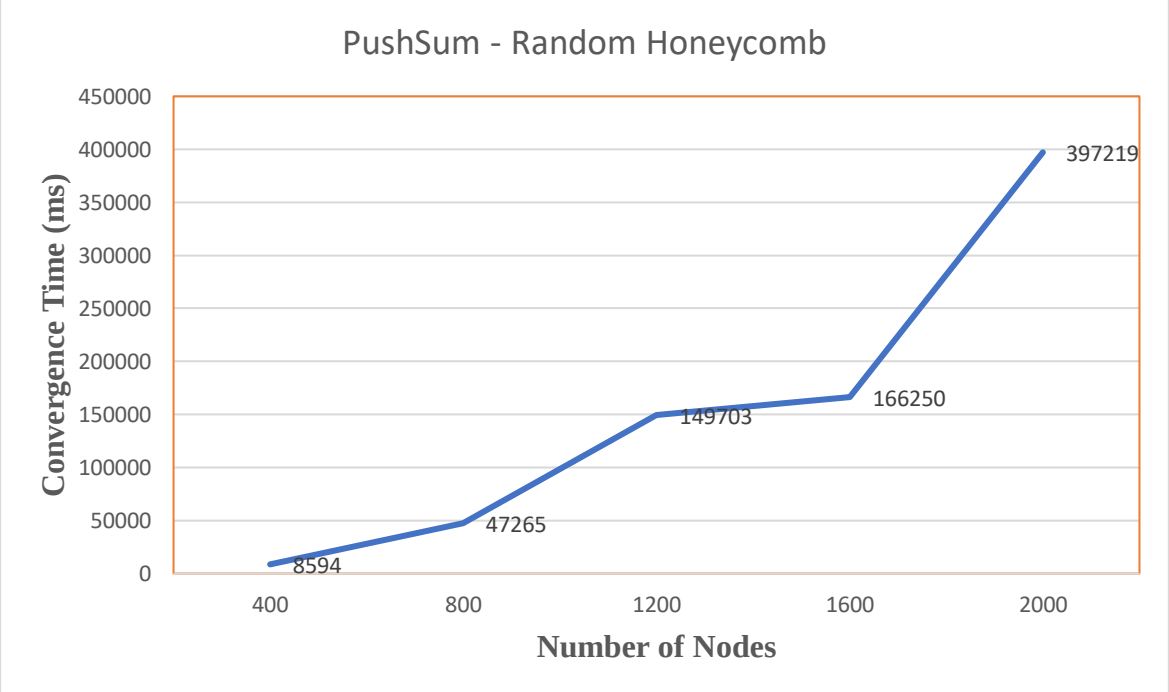
- 3D Torus Grid: Our implementation of Push Sum algorithm supports maximum of 2000 nodes and is converged with time of around 3516 milliseconds for 3D torus Grid Topology.



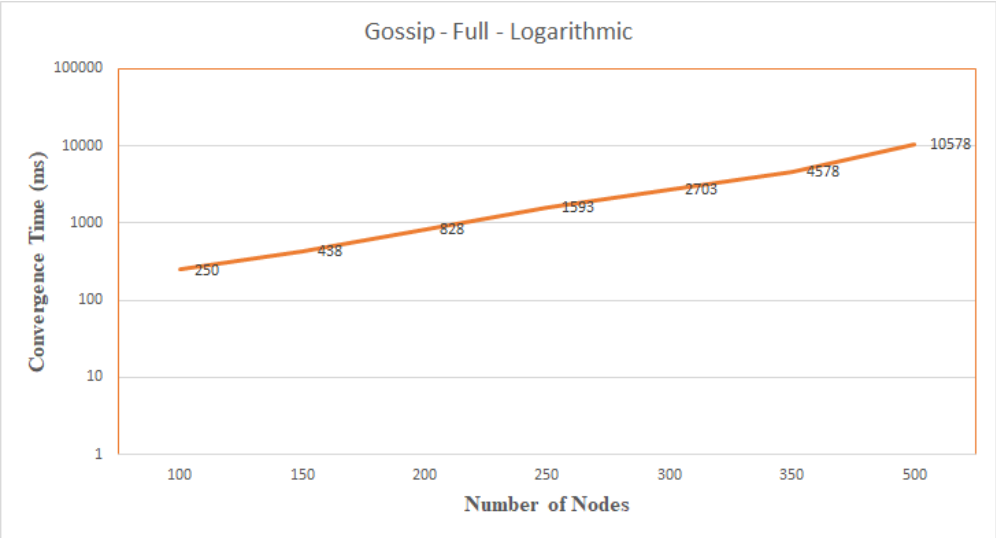
- Honeycomb: Our implementation of Push Sum algorithm supports maximum of 500 nodes and is converged with time of around 671453 milliseconds (approx. 11 minutes) for Honeycomb topology.

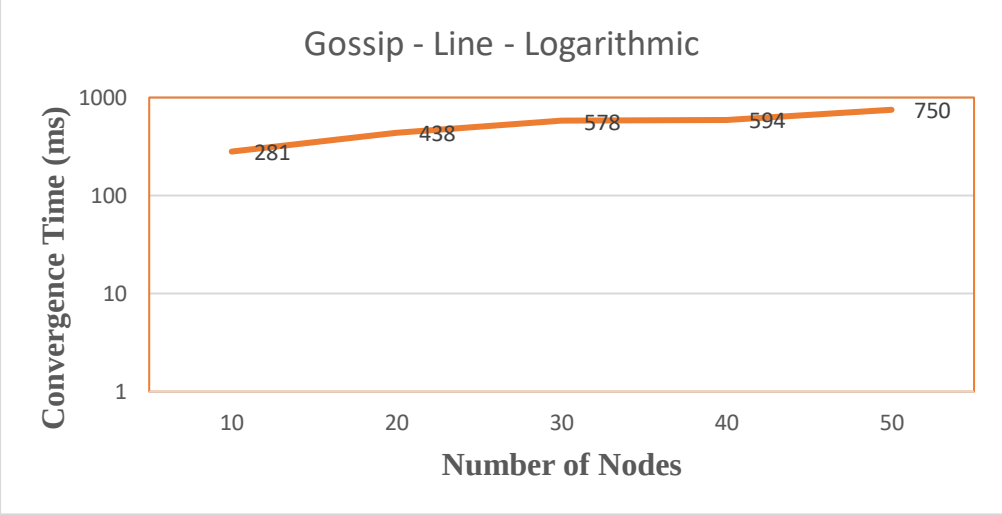


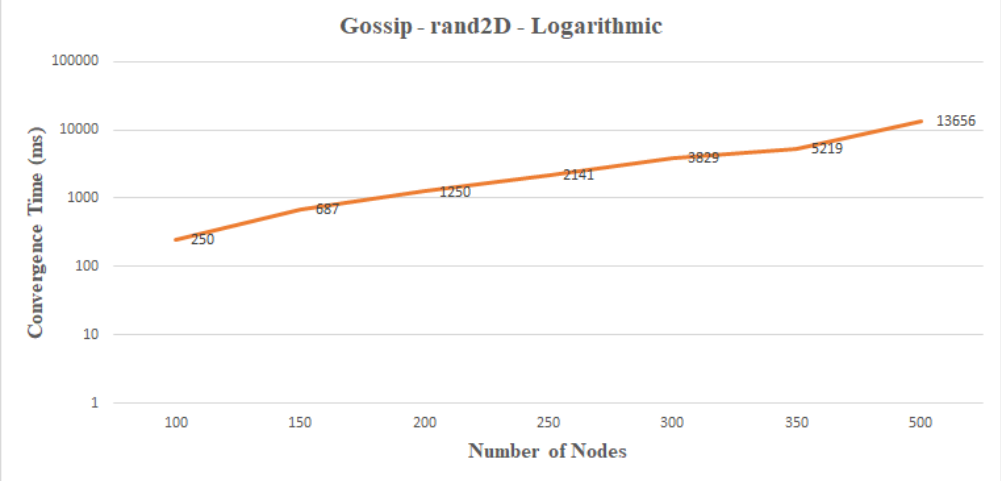
- Honeycomb with a random neighbor: Our implementation of Push Sum algorithm supports maximum of 2500 nodes and is converged with time of around 5500 milliseconds for Random Honeycomb topology.

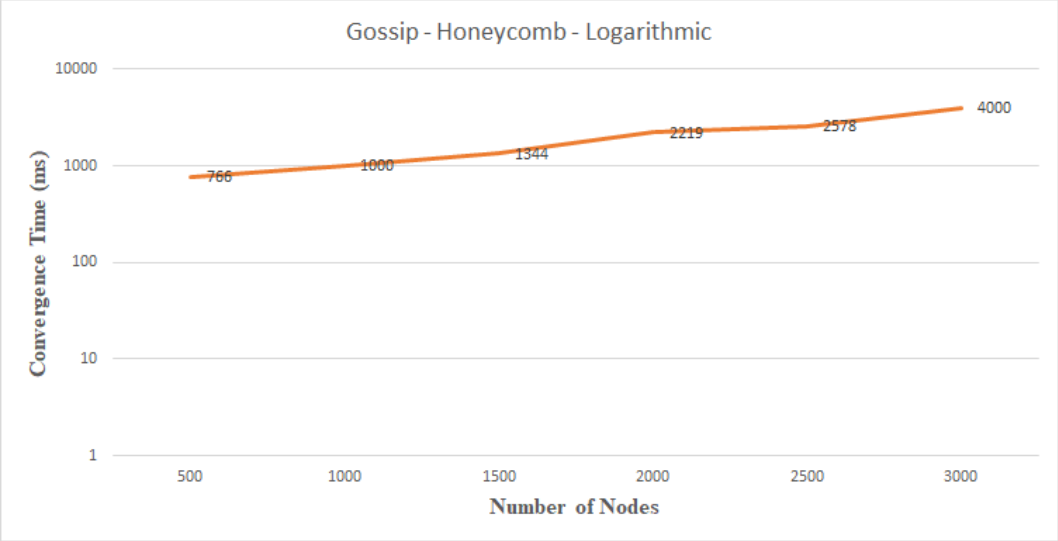


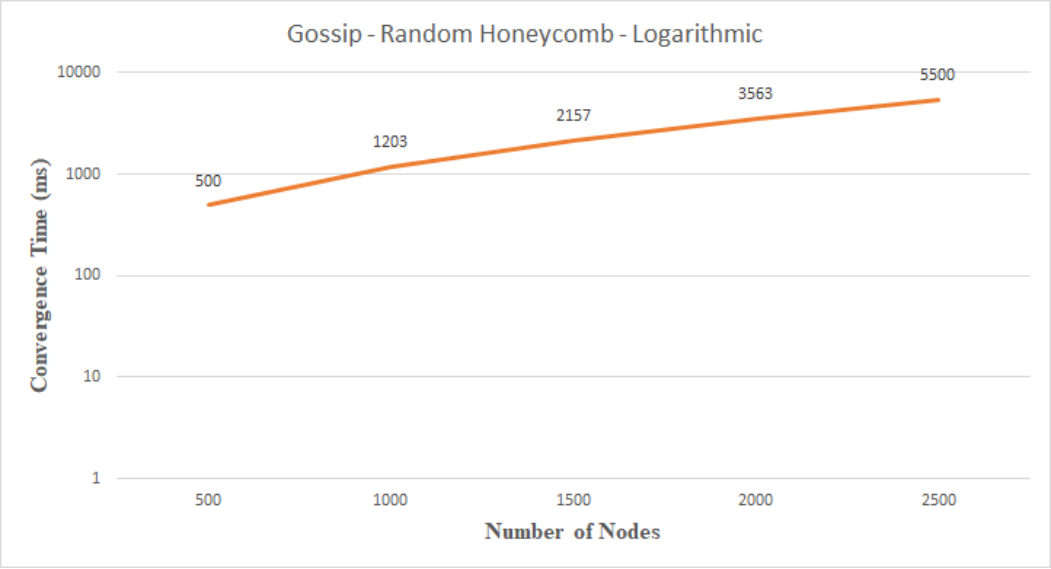
❖ Logarithmic time - Gossip

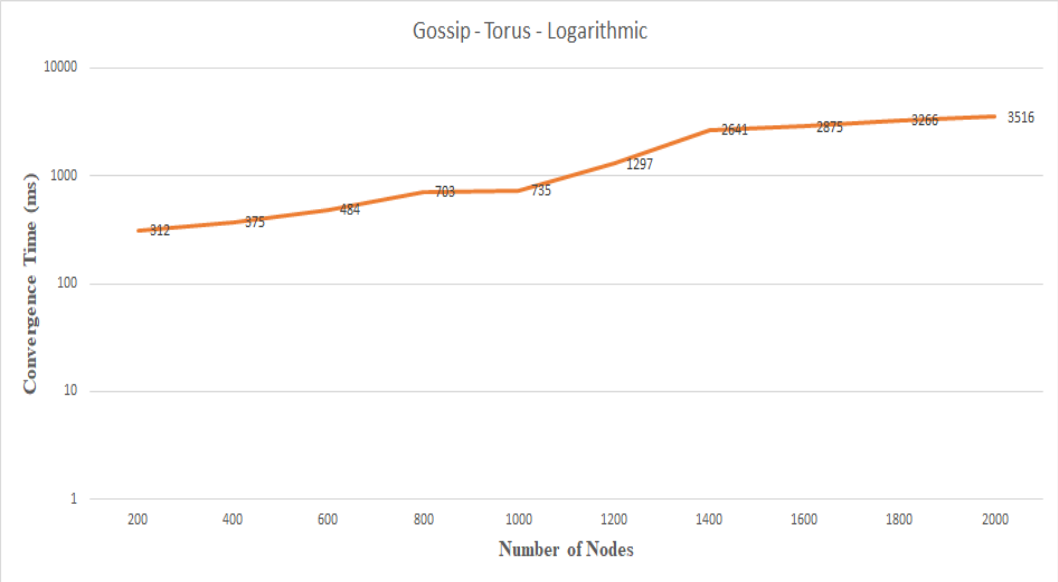




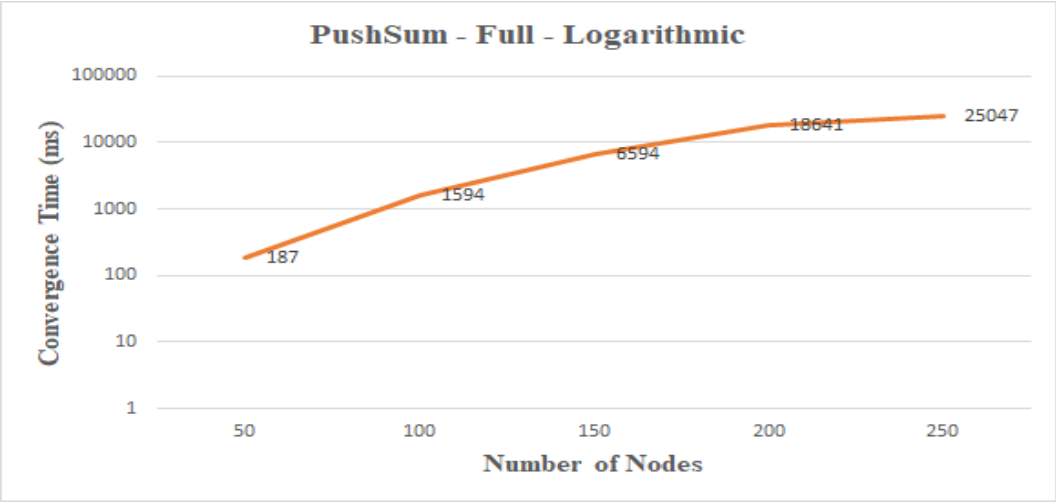


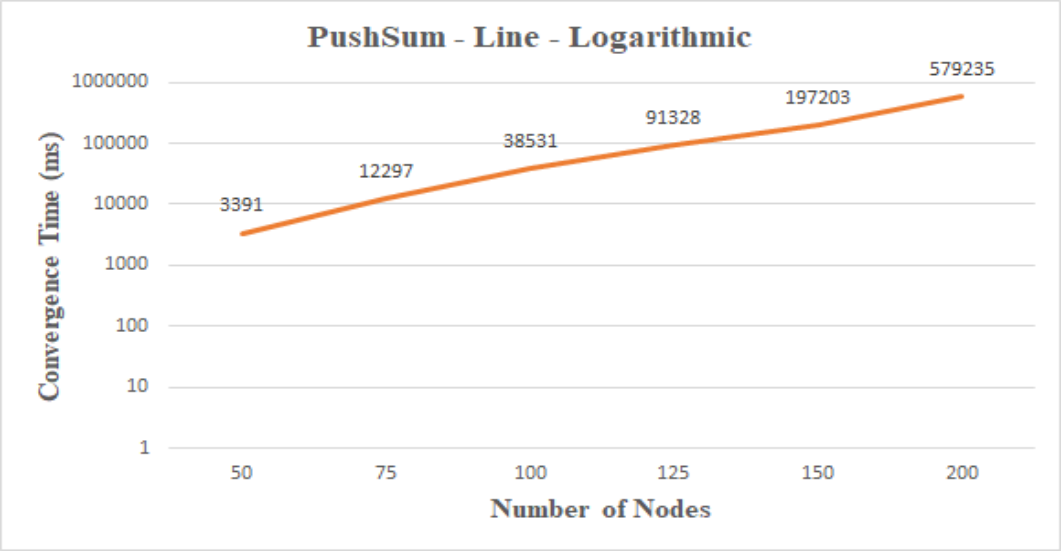

Gossip - Honeycomb - Logarithmic
Convergence Time (ms)
10000
1000
100
10
1
766
1000
1344
2219
2578
4000
500
1000
1500
2000
2500
3000
Number of Nodes

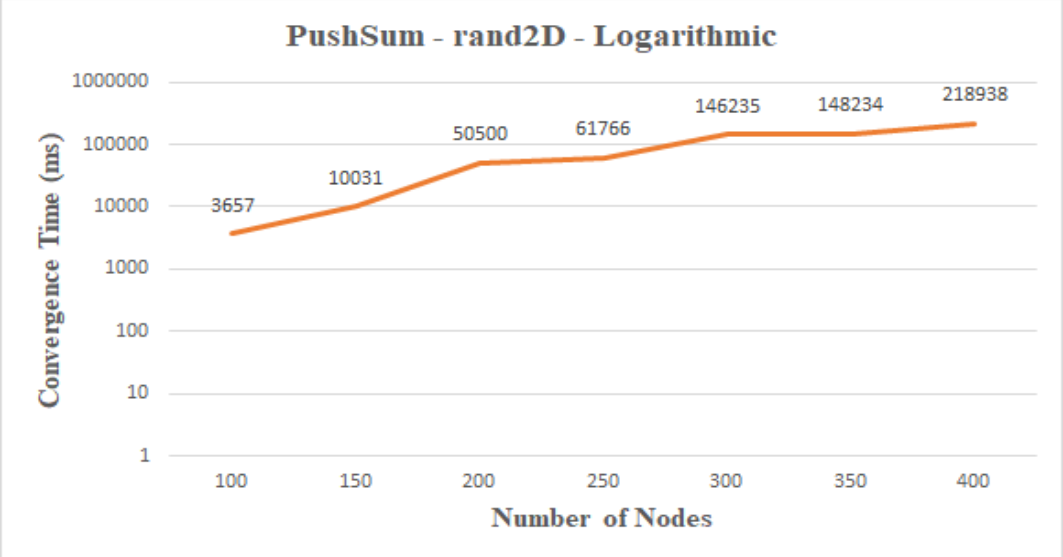

Gossip - Random Honeycomb - Logarithmic
Convergence Time (ms)
10000
1000
100
10
1
500
1203
2157
3563
5500
500
1000
1500
2000
2500
Number of Nodes

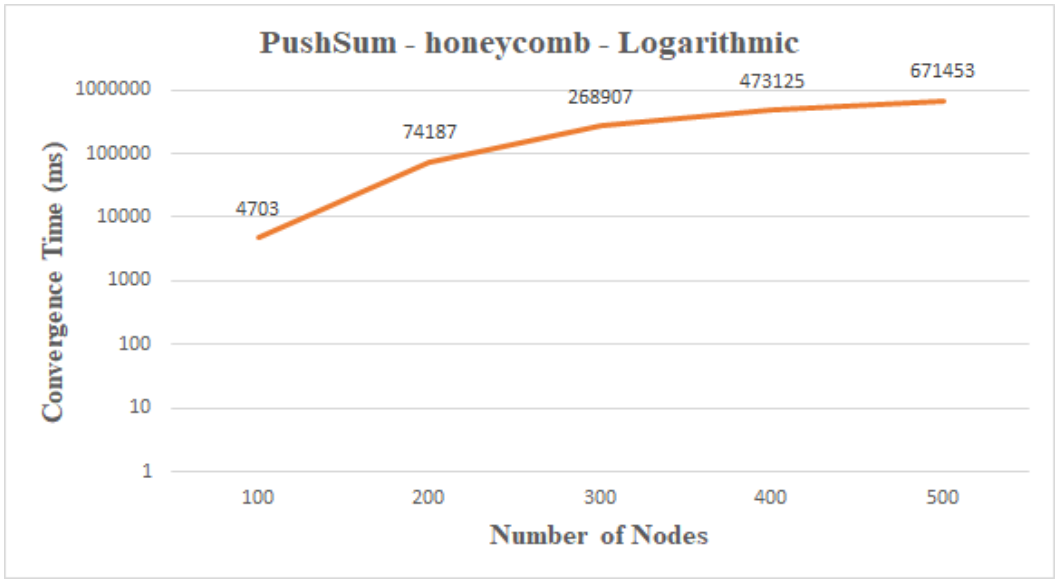

Gossip - Torus - Logarithmic
Convergence Time (ms)
10000
1000
100
10
1
312
375
484
703
735
1297
2641
2875
3266
3516
200
400
600
800
1000
1200
1400
1600
1800
2000
Number of Nodes

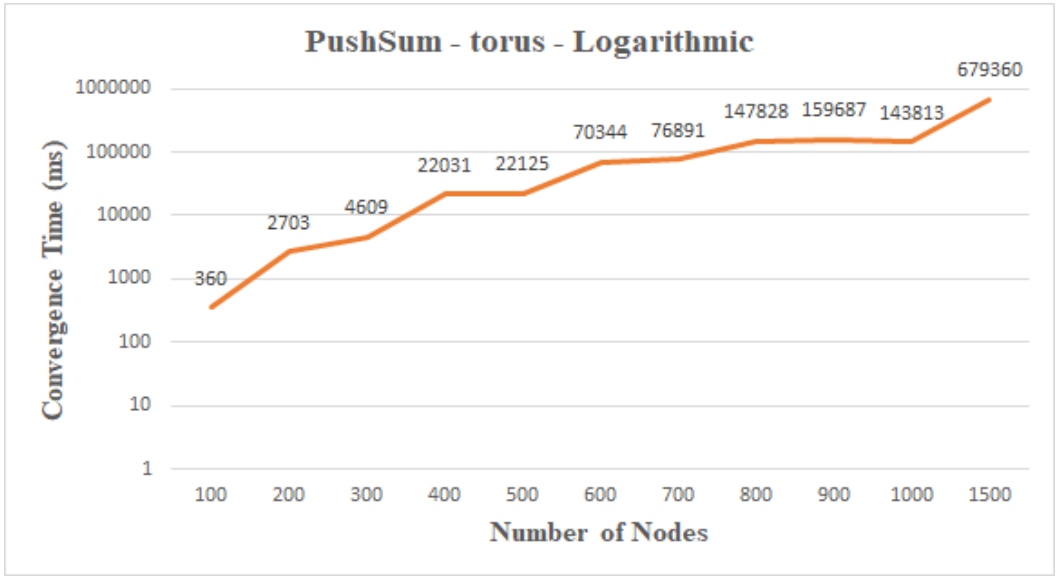
❖ **Logarithmic time – Push Sum**

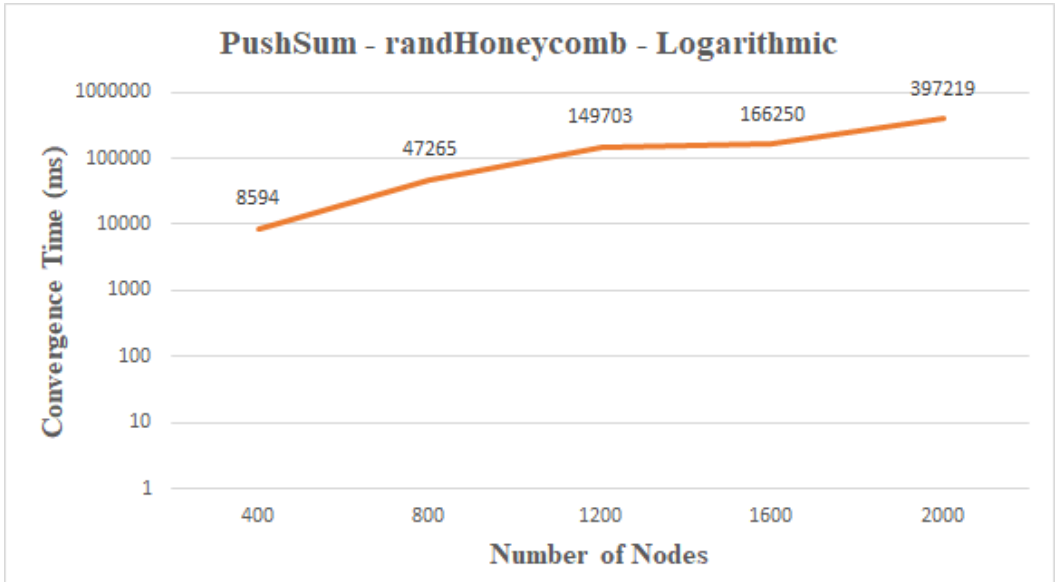







PushSum - honeycomb - Logarithmic
Convergence Time (ms)
1000000
100000
10000
1000
100
10
1
4703
74187
268907
473125
671453
100
200
300
400
500
Number of Nodes


PushSum - torus - Logarithmic
Convergence Time (ms)
1000000
100000
10000
1000
100
10
1
360
2703
4609
22031
22125
70344
76891
147828
159687
143813
679360
100
200
300
400
500
600
700
800
900
1000
1500
Number of Nodes


PushSum - randHoneycomb - Logarithmic
Convergence Time (ms)
1000000
100000
10000
1000
100
10
1
8594
47265
149703
166250
397219
400
800
1200
1600
2000
Number of Nodes

❖ Please describe how you measured the time in your report

We calculated the starting and ending time for Gossip and PushSum algorithm using System.monotonic_time(:millisecond).

❖ Interesting Observations:

- According to our experiment conducted we observed that Gossip algorithm runs faster than Push-Sum algorithm for the topologies even where the performance of gossip is the worst. Additionally, for the cases where the performance of Push sum is good, we observe that Gossip beats the Push Sum algorithm in time. So, this matches the theory that Gossip algorithm performs better than Push Sum algorithm.

- For the Line topology, we have observed that the convergence of the actors is quite unpredictable. As for the Gossip algorithm, the actors are chosen at random and the number of options for every actor is limited to one or two neighbors. This leads to uncertain results.
  For example:
  Number of Actors: 75
  Time taken for running it three times: (500, 391,375) milliseconds
  Number of Actors: 175
  Time taken for running it three times: (766, 1406,1171) milliseconds
  We observe a great difference in time for higher number of actors.

- For the line topology we also observe that sometimes convergence is not at all achieved even for the smallest number of nodes using gossip algorithm as the node selection is random in the topology.