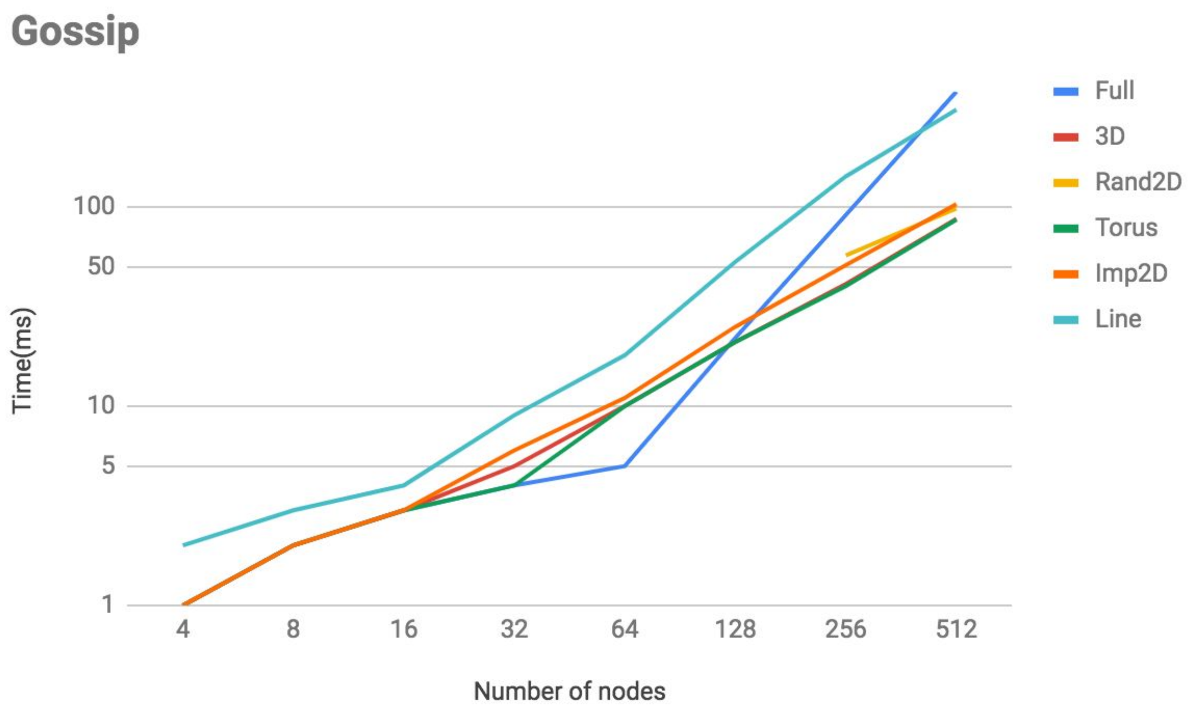
PROJECT2
DISTRIBUTED OPERATING SYSTEMS PRINCIPLES
COP 5615
GOSSIP PROTOCOL IMPLEMENTATION

Group Info:

	Name	UFID
1.	Manishkumar Chopra	17967121
2.	Nimish Kochhar	61394423

Gossip Algorithm

For the gossip algorithm, we choose a random node to start a rumour. Once a node has received a rumour, it chooses one of it neighbours at random and transmits to that rumour. This happens until it hears the rumour at most 10 times, after which it stops transmitting. There is a global counter which keeps track of how many nodes have heard the rumour. When a node hears it for the first time, it increments the counter by 1. We stop the program once this counter is equal to the number of nodes. So,the algorithm converges when every node has heard the



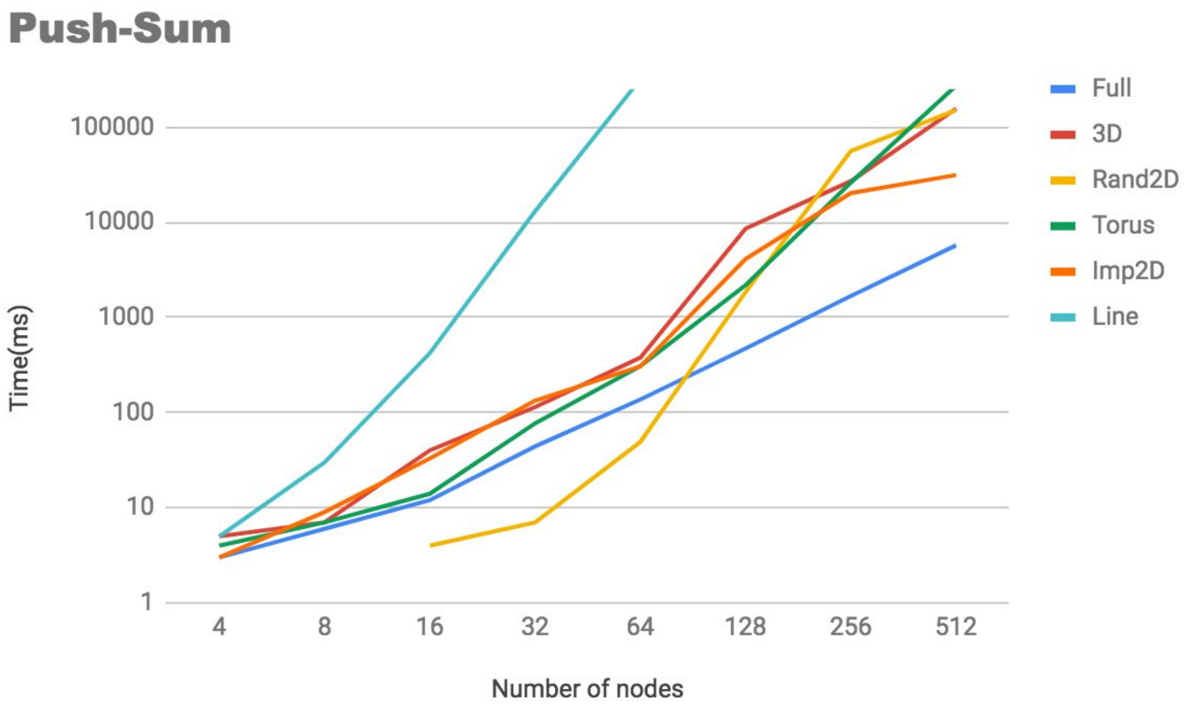
rumour at least once. We record the system time before starting the algorithm and just before exiting the program to get the convergence time for the algorithm.

<u>Interesting Finds</u>

1. In case of small n, random 2D has some nodes which are completely isolated. It can be seen on the graph that there is no input for n<128 for random 2D.
2. Another interesting one is the full network, as n gets large, it takes longer for it to converge than the line topology. A possible explanation for this would be that each node has a large number of neighbours and a node is chosen at random from them to be sent the rumour to. This may delay a few nodes from hearing the rumour.

Push-Sum

For the push sum algorithm, we start by assigning each node a {s, w} pair value where s is the node's index and w is 1. We choose a random node and send it a pair {0, 0}. Upon receiving the pair, it adds it to its current {s, w}. It then calculates the difference in the ratio (s/w) for both the {s, w} pair that existed before and the new pair resulting from the addition. If this difference is less than 10^-10, it updates its own personal counter keeping track of how many times this has happened. When this counter hits 3, it increments a global counter keeping track of how many nodes have converged. When the global counter is equal to the number of nodes, the algorithm converges. We record the system time before starting the algorithm and just before exiting the program to get the convergence time for the algorithm.



Interesting Finds

1. The line topology takes a lot longer than other topologies to converge for large n. In this case, we can see that, for n=64, line topology is moving away from other topologies.
2. We can see that, for random 2D and small n, it takes the least time to converge. Given that having a successful random 2D grid for small n is highly unlikely, we notice that when we do get a random 2D grid it converges the fastest. We attribute it to the randomness of the neighbours and to the random number of neighbours a node might have.
3. For large n, full network converges the fastest. A possible explanation for this would be the large number of neighbours that a node would have and with it the different s values it would receive might help it converge faster.