PROJECT 2 REPORT-BONUS

DISTRIBUTED OPERATING SYSTEMS PRINCIPLES
COP 5615
GOSSIP PROTOCOL IMPLEMENTATION

Group Info:

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

Implementing failed nodes

To implement the failed nodes, we assign every node a life status. The life status is a boolean which tells us if it's alive or not. A failed node would return false. During the initialization period, every node is alive. After the topology is set up and the neighbours are assigned, we use the percentage, given to us as an input, to get the number of nodes that need to be killed. We then choose those many nodes at random and change the alive status to false. We return the number of remaining alive nodes as the new total. The global counter should be equal to this new total for the algorithm to converge.

Handling failed nodes

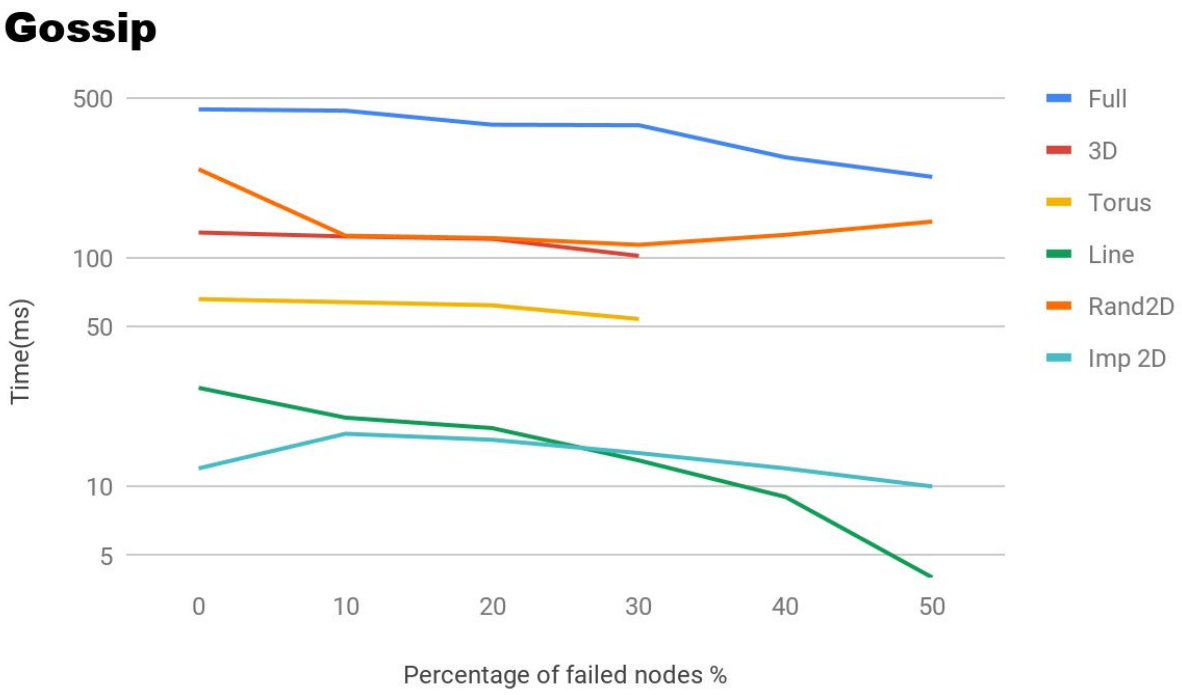
Apart from the standard gossip and push sum algorithm that we implemented, there we some specific changes we did to simulate failed nodes in this program. You can refer to the details of our implementation of the standard algorithms in Project2 Report.pdf

Gossip Algorithm
Before starting the algorithm, every node had neighbours assigned and every node's life status was also updated. To begin the rumour, we chose a random node and checked if it was alive. If it wasn't we recursively look for a live starting node until we find one. Once the node receives the rumour, it checks if it is alive. If it is and all the other conditions to transmit the rumour satisfy, the rumour is sent as usual. But if it isn't alive, the rumour isn't transmitted thus simulating a failed node or a failed connection.

<u>Interesting Finds: Gossip</u>

1. When the % of failed nodes reaches 30, 3D grid and Torus take a long time to converge due to which you see no input on the graph above. This can be attributed to some nodes being secluded out, surrounded by failed nodes.

2. The line topology converges the fastest among all the topologies. Mostly, a single failed node would divide the network into 2, thus not allowing it to converge but in the rare case that all the failed nodes are the edge the network size decreases thus reducing the convergence time.
3. Random 2D grid converges faster than full network topology. In the case that the random 2D is able to form a grid with all the nodes connected, every node has fewer neighbours than full network thus allowing faster convergence.



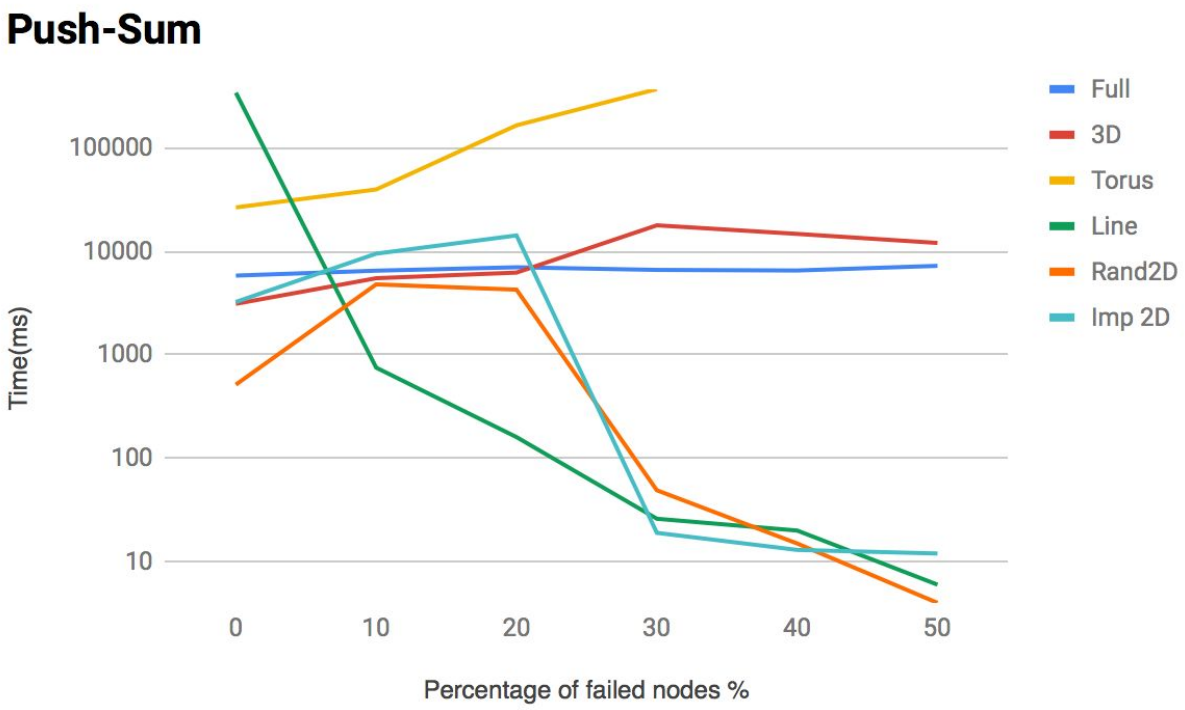
Push-Sum Algorithm

The mechanism to get the starting node is the same for push-sum as for the gossip algorithm. Once a node receives the {s, w} pair, it checks it's life status. If the node is alive, everything happens according to the standard algorithm. If it's dead, it asks the sender node to send the same {s, w} pair to a different node thus simulating failed neighbours.

Interesting Finds: Push-Sum

1. Line and Imperfect Line converge for both the algorithms similarly whereas Random 2D converges much faster in case of Push-Sum as compared to Gossip. Even though the problem size decreases in the case of line and imperfect line, random 2D has a larger reach than both of them and has the same advantage of a smaller problem size as them thus explaining the convergence.

2. One interesting finding was that in both the algorithms, Torus stop converging after more



than 30% nodes fail. The same explanation for this can be given, failed nodes create some secluded nodes which inhibit the convergence of all the nodes.