DOS PROJECT2 REPORT

ADITYA BHANJE(UFID: 9697 1842)

AKASH R VASISHTA(UFID: 5395 5080)

BRIEF SUMMARY:

The project aims to implement the Gossip Algorithm for information propagation and Push-Sum Algorithm for sum computation on various types of networks.

Topologies Implemented:

The various topologies implemented are:

1) Full Network: Every node in the network knows all other nodes of the network.
2) Line Network: All the nodes are arranged in the form of a line where each node only knows the node before and after it (except the first and the last node)
3) Imperfect Line Network: Similar to the line network except that it also knows another random node from the network .
4) Random 2D Grid: All the nodes are mapped in a [0-1.0] x [0-1.0] square. A node is said to be neighbour of another node if they lie within a distance of 0.1
5) 3D Grid: All the nodes are distributed on a uniform 3D Grid
6) Torus: All the nodes are arranged in such a manner that they form a 3D Toroid.

IMPLEMENTATION DETAILS:

The program is given input in the form of:

<numOfNodes> <Topology> <Algorithm>

where

numOfNodes: Total number of nodes in the network (rounded for 3D Grid, Torus)

Topology: The type of network in which the nodes are to be arranged

Algorithm: The algorithm which we want to implement i.e. Gossip or Push Sum

Gossip Algorithm:

- It is initiated by the main process sending a rumour to any node (actor) from the given set of nodes which has it's neighbour list with it.
- Once this node receives the rumour, it increments the count of number of times it received the rumour.
- It then chooses one of the active nodes randomly from the neighbouring list and then transmits to it and so the node which received the rumour repeats the process above.

- A node stops transmitting the rumour only if the count of the number of times it received the rumour is 10. The transmission of the rumour to it's neighbours is asynchronous to the receive process.
- While doing transmission a case may arise such that a node has no alive nodes in it's neighbour list i.e. all those nodes have terminated as they have already received the message 10 times but it's own receive count is less than 10. So in this case we put the node itself in it's neighboring list after waiting for some time to check if it receives rumour from other nodes, such that it keeps transmitting to itself and then eventually terminates once it's receive count becomes 10.
- **The above decision was taken, considering the fact that it may so happen that a node might never receive a rumour after some time as the nodes in whose neighbour list it is present had never chosen it and terminated eventually or it might never be in anyone's neighbour list at all(might happen in random 2D Grid) due to which it might it never reaches the termination condition. So to make sure all the nodes do terminate , we have taken this decision to add itself to it's neighbouring list if it doesn't have any neighbours and terminate itself.**

Push-Sum Algorithm:

- Each node (actor) in the network maintains an initial state of (s=Node Number, w=1). The main process chooses one actor at random.
- This actor then updates it's 's' to s/2 and w to w/2 and sends a message in the form of (s/2,w/2) to any of the alive node in it's neighboring list randomly.
- A node upon receiving this message adds the received s and w to it's own s and w values checks if the sum computation and repeats the procedure above. A node stops sending message if the s/w ratio has not changed more than 10^{-10} in the last 3 times it received a message and the network is said to converge if all the actors have stopped sending the message.

OBSERVED RESULTS:

It is the time taken for the network to converge for the given algorithm (given in milliseconds). The time to converge is calculated as the time difference between the time the algorithm was started (not considering network creation) and the time when all the actors terminated.

For **Gossip Algorithm**, the following order was observed (in increasing order of convergence time):

Torus < 3D Grid < Imperfect Line < Random 2D Grid < Full < Line

For Push-Sum algorithm, since the Line network was not converging properly, we tried to device a solution by making sure that each node in neighbour list if a node gets selected as equally as possible for the message transmission. However the Line network still doesn't converge for large values of nodes.

Torus < Imperfect Line < 3D Grid < Full < Random 2D Grid< Line

OBTAINED RESULTS:

1) **Gossip Algorithm Convergence time (in milliseconds):**

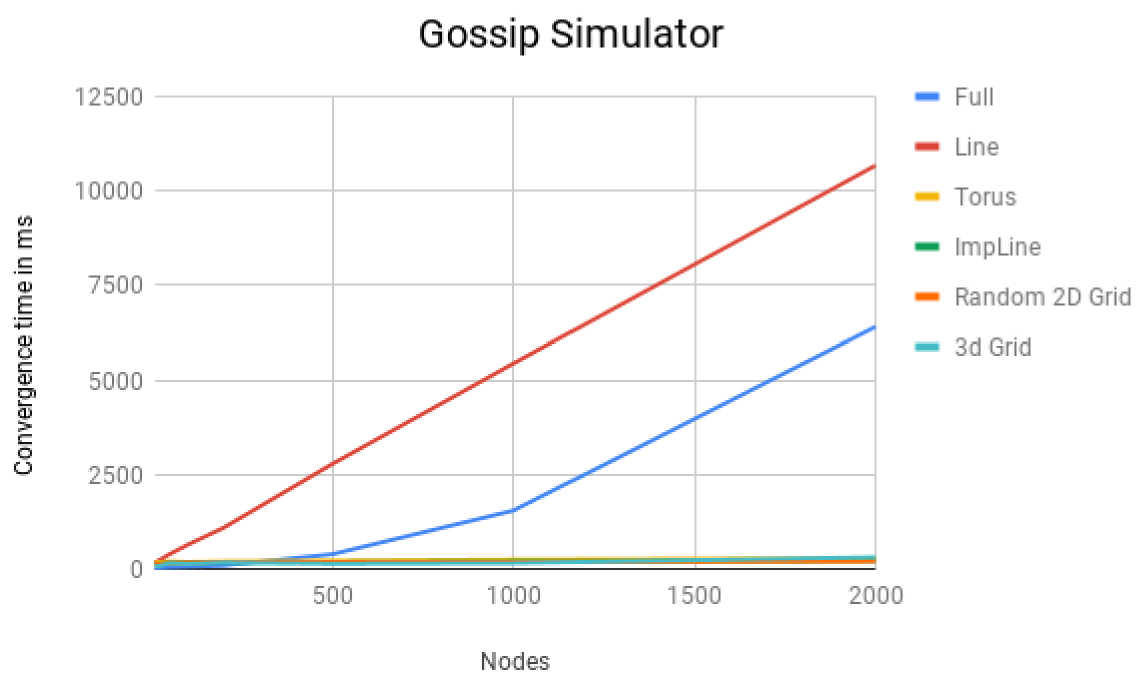
Number of Nodes	FULL	LINE	IMPERFECT LINE	3D GRID	TORUS	RANDOM 2D GRID
10	16	172	172	125	125	72
25	31	250	188	140	141	93
50	47	391	203	156	156	156
100	62	640	187	172	172	140
200	94	1094	203	172	187	172
500	391	2781	219	188	187	140
1000	1547	5437	250	203	188	156
2000	6406	10672	282	219	203	313

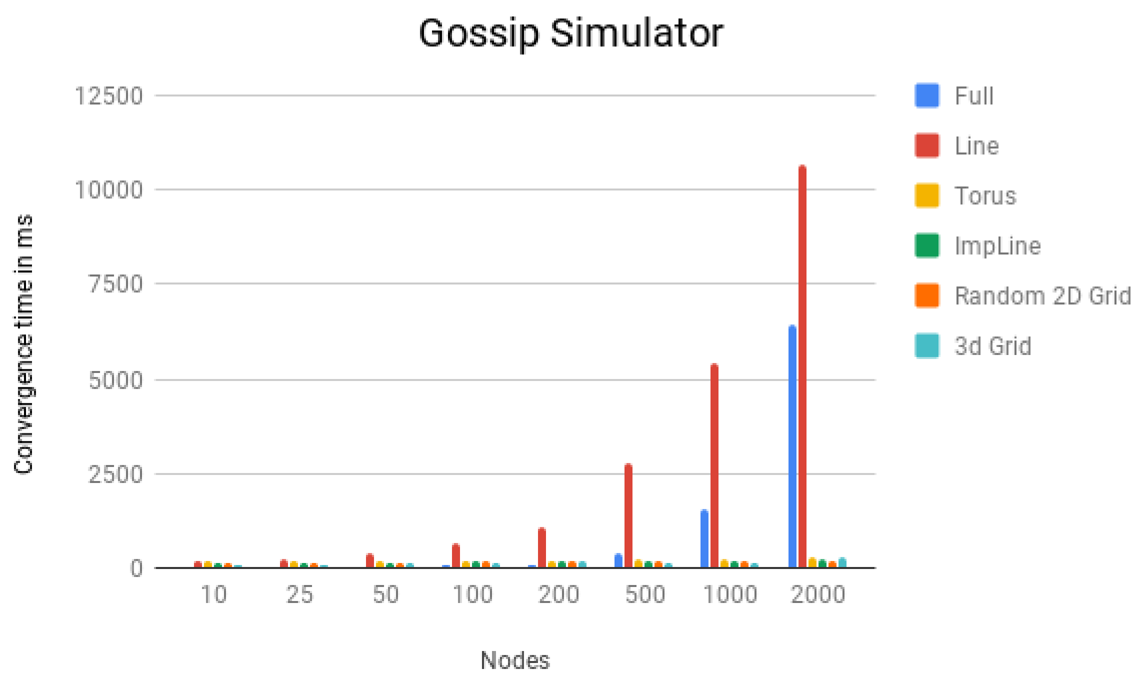
2) **Push-Sum Algorithm Convergence time (in milliseconds):**

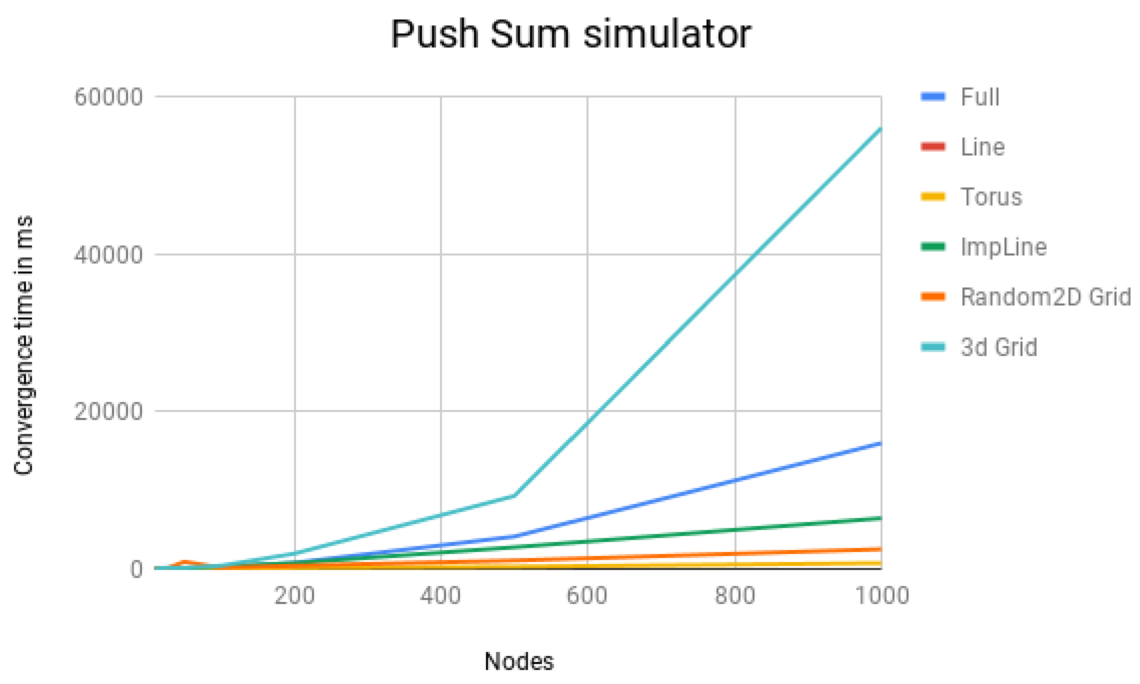
Number of Nodes	FULL	LINE	IMPERFECT LINE	3D GRID	TORUS	RANDOM 2D GRID
10	20	16	30	32	32	47
25	31	47	46	47	47	163
50	78	47	70	109	917	156
100	203	62	109	329	172	422
200	781	*93	163	750	422	1954
500	4125	N.A	297	2765	1078	9282
1000	16000	N.A	750	6453	2515	56000

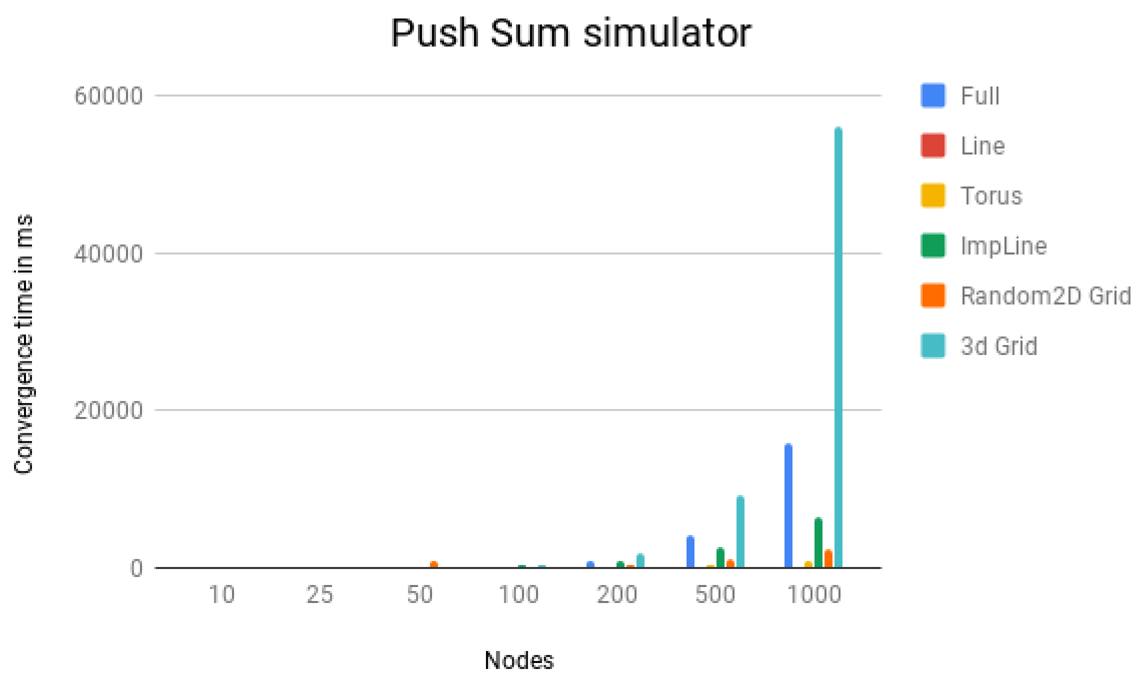
- and N.A. : Algorithm had to be re-run many times as the network was not converging for line topology and random 2D for both push sum and gossip algorithm

GRAPHS:


Gossip Simulator
Convergence time in ms
12500
10000
7500
5000
2500
0
500
1000
1500
2000
Nodes
Full
Line
Torus
ImpLine
Random 2D Grid
3d Grid


Gossip Simulator
Convergence time in ms
12500
10000
7500
5000
2500
0
10
25
50
100
200
500
1000
2000
Nodes
Full
Line
Torus
ImpLine
Random 2D Grid
3d Grid


Push Sum simulator
60000
40000
20000
0
200
400
600
800
1000
Convergence time in ms
Nodes
Full
Line
Torus
ImpLine
Random2D Grid
3d Grid


Push Sum simulator
60000
40000
20000
0
10
25
50
100
200
500
1000
Convergence time in ms
Nodes
Full
Line
Torus
ImpLine
Random2D Grid
3d Grid