DOS: Project 2

Kartik Rode (UFID: 8971-1791)
Aditi Page (UFID: 0427-9968)

Description

In this problem, Gossip and Push-Sum algorithms are implemented on the four topologies: full, line, 2D, and imperfect 2D. These algorithms are used to propagate information between actors in these topologies. A concise summary of the working of these algorithms -

- Gossip: This algorithm initiates a process starting with one actor and then the first actor propagates the message to the next arbitrarily chosen actor in the topology and the process continues. The point of convergence is reached when each of the actors listen to the message 10 times, in this case.
- Push-Sum: This algorithm also propagates messages but the messages are sent in the form of pairs (s, w) where 's' is the value of the given actor and 'w' is 1 initially. The point of convergence is reached when the s/w ratio is constant when compared to a predefined value, 10^-10 in this case, for three consecutive times.

The topologies used are:

1. Full Network: Every actor is connected to every other actor in the network.
2. Line Topology: Every actor is connected to the previous actor and the one next to it except the first and the last actors. The first actor is only connected to the second actor and the last actor is connected to its previous actor. The messages in this topology can be sent in two ways. The actors in the middle can send and receive messages from the previous actor and the actor in front of it. The first actor can send messages to the second actor and receive from the second actor only. The last actor can receive messages from the second last actor and can send the message to the second last actor only.
3. 2D Grid: Actors are arranged similar to the orientation in a 2D grid and the messages from one actor to another actor can only be forwarded or received if they are neighbours.
4. Imperfect 2D: It can be thought of as an extension of 2D grip topology where the actors are in 2D and additionally, a random actor is selected to become another neighbour from all the actors.

Implementation Details

Each of the two algorithms is implemented on each of the four topologies. Implementing the cases from the the possible combinations:

Gossip Algorithm: We have implemented gossip such that a boss actor is present which tracks the convergence of all the actors. Additionally, other actors are created such that they each have ‘gossip’ messages and ‘dummy’ messages. The ‘dummy’ messages are the messages that the actor sends to itself and the ‘gossip’ messages are the ones that are sent to other actors. A map of actors is used where the neighbours of each of the actors is listed. This map is used for facilitating the process of finding neighbours. The command ‘Start’ initializes all the neighbours using this map. A counter is used for tracking the number of messages received by an actor. If that number is, in this case, 10, then this information is reported to the boss actor.

1. Gossip-Full: In the case of Full Network topology, 100% convergence is observed. Since all actors are connected to all other actors, there is no situation where the process is halted.
2. Gossip-Line: In the case of Line topology, a halt condition can be encountered since actors send messages to their neighbours. This means, that if the neighbour actors are already converged, then the process of propagating messages in the network will stop. To resolve this issue, we have used ‘dummy’ messages that send messages to self. This way the actor can still receive the messages. When a dummy message is received by an actor for the 100th time that has already been converged, the actor will forward the message to another actor so that working in the topology is not halted. It is observed that even after using this method, 100% convergence is not achieved. So, the convergence criteria for Line topology is set to 95%.
3. Gossip-2D: In this case, a 2-d array is created and used for storing the references to all the actors. Like in the previous cases, a map is used but for this case, the map is created between the array for references and their indices. ‘Gossip’ and ‘Dummy’ messages are used similarly. To keep track of neighbours of any actor, we build an array for neighbours depending on the location of the actor in the 2D array. For example, if the actor is at the location (0, 0), then this actor will have 2 neighbours in the 2D grid. An array of these two neighbours will be created. Similarly, for the actors situated in between the 2D array, the number of neighbours will be 4 and so, an array for 4 neighbours will be created. To forward messages to the neighbours, any random neighbour is selected from this neighbour array to which the message is forwarded. When the actors converge, it is reported to the boss. A similar situation like in Line topology arises and so, the dummy messages implemented help in avoiding the halt situation. After introducing this, 100% convergence can be achieved in 2D Grid topology.
4. Gossip-Imperfect 2D: This is implemented in a similar manner as Gossip algorithm in 2D. An addition to it is that we maintain another array which contains the nth actor corresponding to its position in the 2D array. For example, the actor at (0, 0) will be #0, the actor at (n-1, n-1) would be #n^2. While picking a random neighbour in this

case, we first check if the randomly picked neighbour from this list is an already existing neighbour. If it is, then another neighbour is randomly picked. This is repeated until an actor which is not an existing neighbour is picked. Imperfect 2D converges 100%.

Push-Sum: In this algorithm, we use only one message (no dummy message like in Gossip algorithm). Initially, all actors are assigned the pairs (s, w) where s is the actor number and w=1. Any actor randomly starts the message forwarding process. When an actor sends a message, it sends (s/2, w/2) to another actor which leaves the sender actor with (s/2, w/2). When an actor receives a message from another actor, it adds the values to its current values: (s_old + s/2, w_old+w/2). When the s/w ratio does not change for 3 consecutive rounds, then the actor converges. In contrast to the Gossip algorithm, we have not used self-messages that the actor sends to itself in this Push-Sum algorithm.

5. Push-Sum Full: For this topology, once the (s, w) and the list of neighbours is given, Full network topology works perfectly and converges 100%.
6. Push-Sum-Line: In Line topology, the first priority is always given to the neighbours for sending a message, but in the case that the neighbours are converged, then another actor is chosen. This topology alo requires that (s, w) and list of neighbours is passed.
7. Push-Sum-2D: Similar to Line topology, neighbours are given preference and if the neighbours are converged already, then another actor is chosen as the neighbour and the message is forwarded to that actor. In this case, along with the (s, w) and neighbour list, coordinates of the actors are also passed.
8. Push-Sum-Imperfect 2D: This works similar to 2D grid and is also given the same parameters: (s, w), list of neighbours and coordinates of the actors.

Interesting Observations

1) Gossip

- In Full network and Imperfect-2D grid topologies, 100% convergence was observed
- In 2D grid topology, 95% convergence was observed
- In Line topology, 90% convergence was observed

To tackle the problem in 2D grid and line topology where neighbours of an actor are converged and the actor does not receive any messages further, ‘dummy’ messages are sent to the actor itself and after 100 such messages, it will send a message to one of its neighbours. This way further convergence can be achieved. After implementing this, we get the following results for 2D grid and Line topology:

- In 2D grid, 100% convergence is observed

- In Line topology, 95% convergence is observed and hence, the stopping criteria is set to 95%

2) Push-Sum

- In Full network and Imperfect-2D grid topologies, 100% convergence was observed
- In 2D grid and Line topology, 90% convergence was observed

To tackle the problem in 2D grid and line topology where neighbours of an actor are converged and the actor does not receive any messages further, then the actor can pick any actor other than its current neighbour and itself as a new neighbour and send the new neighbour a message. This way the process of passing the messages in the topology will continue. After implementing this, we get the following results for 2D grid and Line topology:

- In 2D grid, 100% convergence is observed
- In Line topology, 97.5% convergence is observed

Instructions to Run

1. Unzip the file
2. Run the project using:

- dotnet fsi --langversion:preview proj2.fsx numNodes topology algorithm

(if your system is running a language version lower than 5.0)

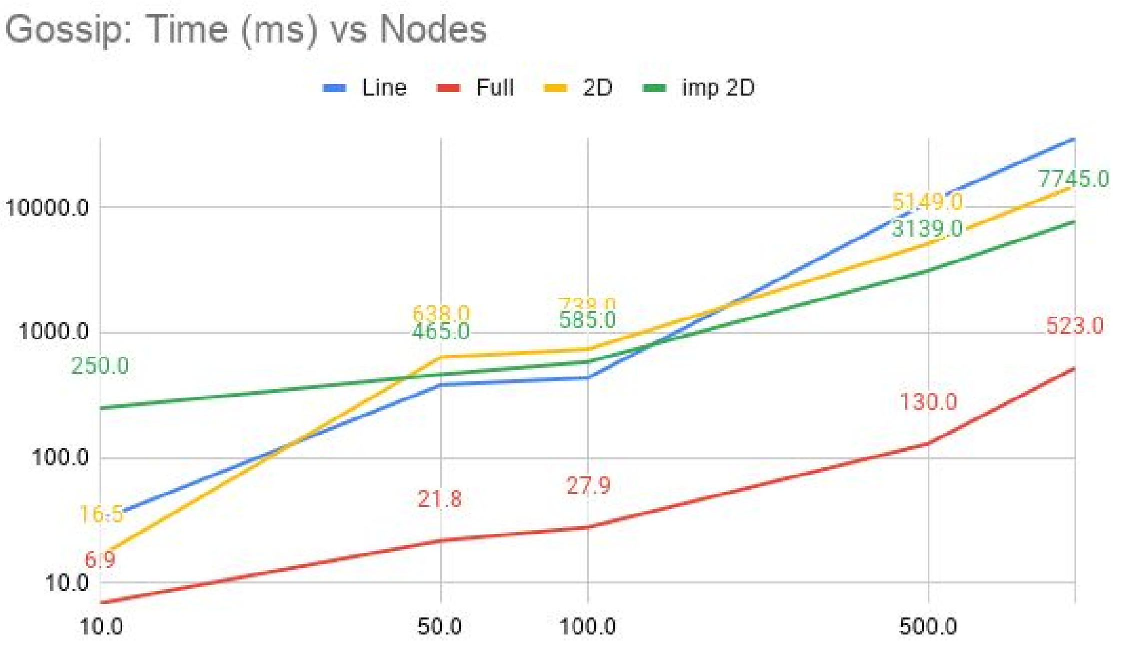
IMPORTANT: Use the following while running the commands

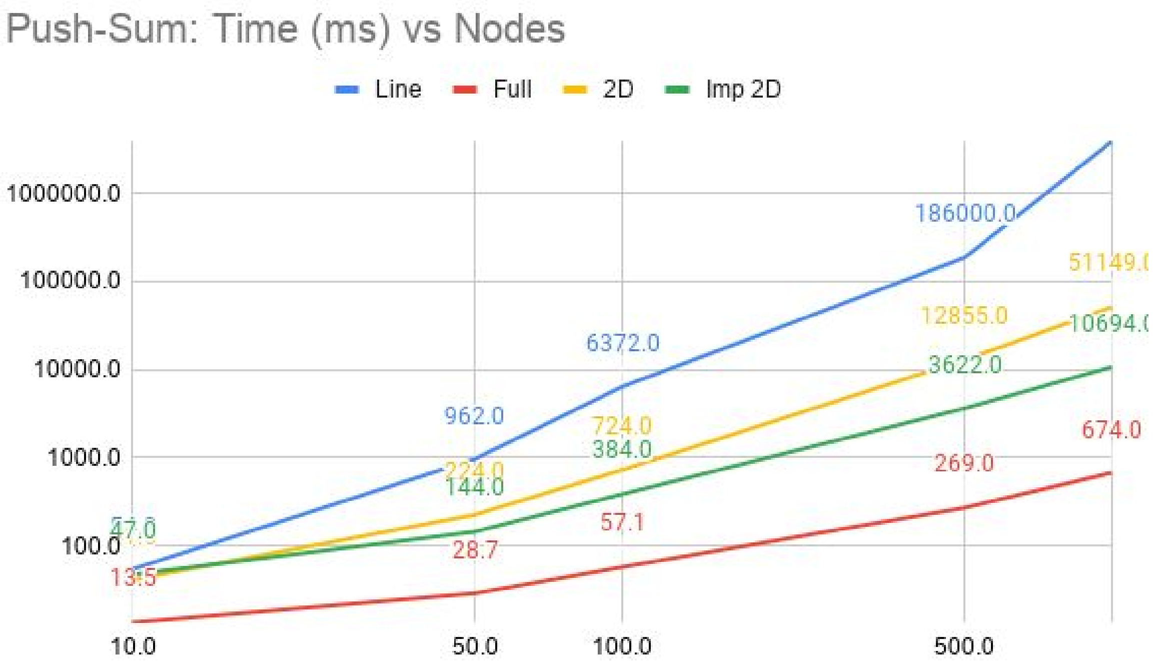
- Case Full Gossip: *dotnet fsi --langversion:preview proj2.fsx numNodes full gossip*
- Case Line Gossip: *dotnet fsi --langversion:preview proj2.fsx numNodes line gossip*
- Case 2D Gossip: *dotnet fsi --langversion:preview proj2.fsx numNodes 2D gossip*
- Case Imperfect 2D Gossip: *dotnet fsi --langversion:preview proj2.fsx numNodes imp2D gossip*
- Case Full Push-Sum: *dotnet fsi --langversion:preview proj2.fsx numNodes full gossip*
- Case Line Push-Sum: *dotnet fsi --langversion:preview proj2.fsx numNodes line gossip*
- Case 2D Push-Sum: *dotnet fsi --langversion:preview proj2.fsx numNodes 2D gossip*
- Case Imperfect2D Push-Sum: *dotnet fsi --langversion:preview proj2.fsx numNodes imp2D gossip*

Results

In the graphs below, the convergence time in milliseconds is plotted against the number of actors/nodes in each of the 4 topologies. The graphs are plotted for both Gossip and

Push-Sum algorithms. Both the axes are converted to log scale while plotting for clearer results.


Gossip: Time (ms) vs Nodes


Push-Sum: Time (ms) vs Nodes