Project 2

Distributed Operating System Principles
COP5615

Lars Pontus Ovhagen (UFID 2992-9498)
James Howes (UFID 9262-9312)

1 Introduction

In this report we outline the performance when simulating different topologies using the *Asynchronous Gossip* and *Push-Sum* algorithm. Furthermore, we measure the convergence time for different network sizes and present our findings for each topology. For *Gossip* this is when all the nodes in the network has heard the rumor and for *Push-sum* it is when the node ratio frequently changes by less than 10^{-10}. The source code is exclusively written to leverage the actor model in Elixir, which allows us to simulate node communication and monitor convergence in real-time.

2 The Results

Below we present the result from running our simulations on all the topologies. The size of the network is gradually increased for each simulation, spanning from 8-4096 nodes in size. Furthermore, a simulation is done by initializing and carrying out the two algorithms on all the topologies for ten rounds. Then finally the average convergence time is calculated over these ten rounds. Another important thing to note is that the number of rounds needed for a *Push-Sum* node to converge was set to 5 as supposed to 3 (as stated in the project description). *Figure* 1 represents the results yielded from running the *Gossip* simulation and *Figure* 2 shows the results for the *Push-Sum* algorithm. Note that the convergence time is measured in the **number of messages** and not clock time. The reason for this is the apparent latency present when running a large network, which significantly affected the actual time for running our simulations.

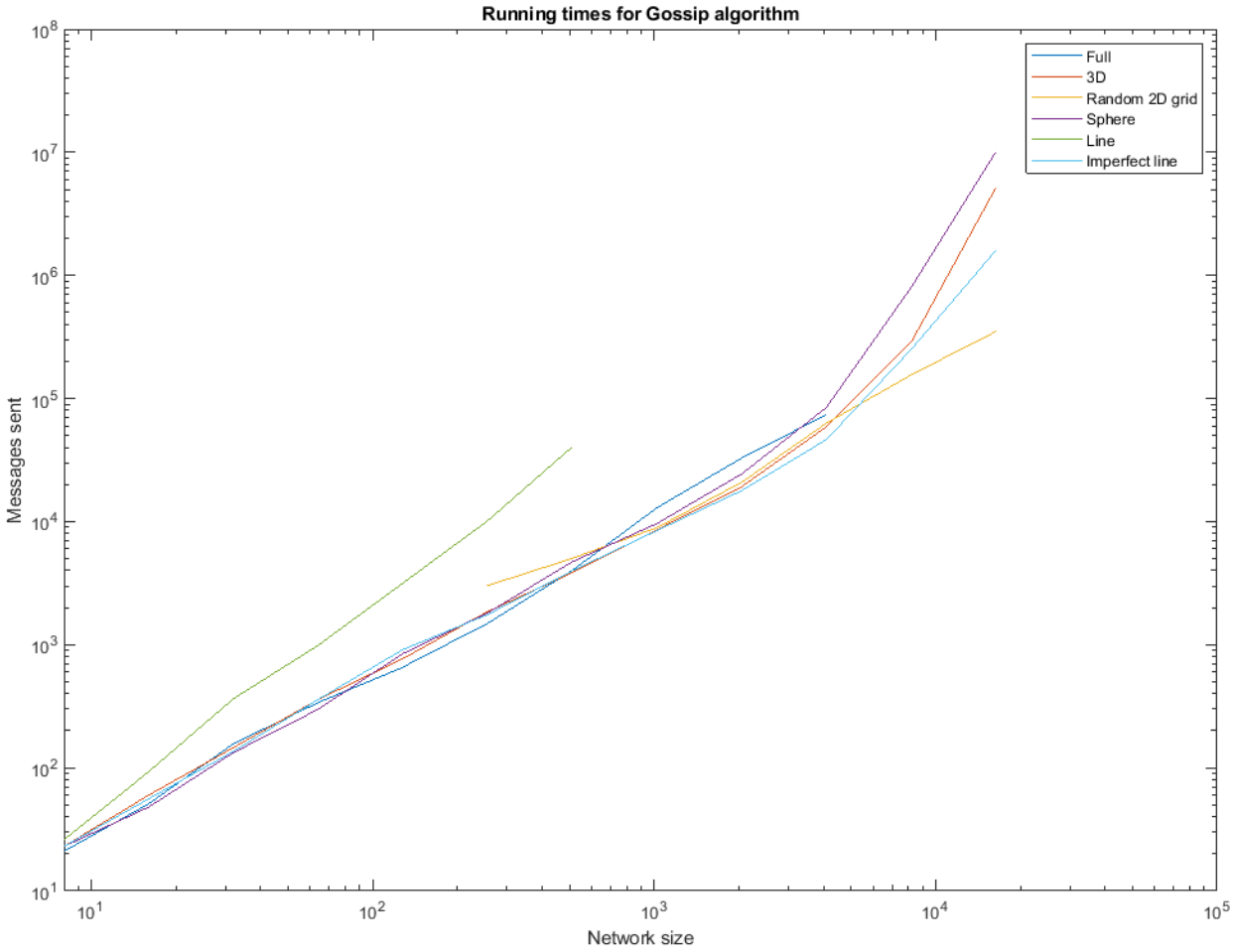

Figure 1: Graph showing the convergence time for different sized networks for each topology using the *Gossip* algorithm. This plot uses logarithmic scales on both the X and Y axis.

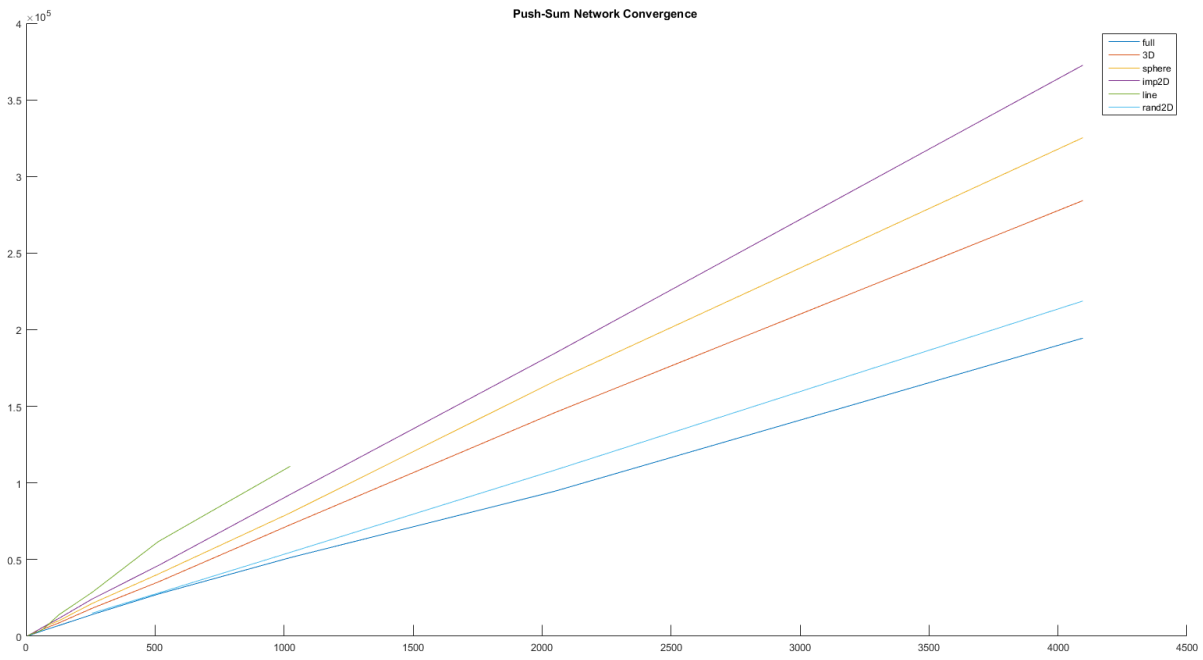

Figure 2: Graph showing the convergence time for different sized networks for each topology using the *Push-Sum* algorithm.

2.1 Fully connected network

The fully connected network defines a network where all nodes are interconnected.

This topology typically performed the best at all network sizes for our simulations of the *Gossip* algorithm, although we believe it should have performed even better. At large network sizes (> 2000) the architecture of our system could not handle the load and severely impacted performance, often leading to a running time orders of magnitude larger than other topologies. We identified a couple problems:

First, each node must maintain a complete list of neighbors to choose from, and sending a message requires an $O(n)$ search through that list for a random neighbor, so the running time grows rapidly because there are n nodes .

Secondly, because the full network achieves convergence so quickly, there is a huge volume of messages sent to our network observer to notify of convergence. Processing all of these messages takes significant time, especially when the system is also using resources to continue handling gossip traffic. In the time it takes the observer to process all the convergence notifications, the gossip nodes have continued to send more messages. We suspect this results in an inflated message count at all network sizes.

With the *Push-sum* algorithm, these problems were somewhat mitigated because the

nodes did not converge as rapidly and overwhelm the network observer. It still performed better than all other topologies.

2.2 3D Grid

The 3D grid represents a three-dimensional space where nodes are interconnected in an orthogonal grid, i.e. a cube.

Unlike the Full network, the number of neighbors is limited to a maximum of 6 per node, which reduced the loading problems described above. Furthermore, we observed a moderate performance for both the *Push-Sum* and *Gossip* algorithm.

2.3 Random 2D Grid

A 2D grid is a topology where all the nodes are randomized in a two-dimensional space, and neighbors are connected within a certain proximity. We used a proximity of 0.1 in a 1x1 space, so on average each node is neighboring approximately 3.14 percent of the total space.

A notable difference with the performance of the 2D grid is that its convergence time scales well with the size of the network. When there is a large number of nodes present in the network, the nodes are densely positioned in relation to one another, improving the convergence time significantly. Also, since each node is neighbors with only about 3-4 percent of the total nodes, it does not experience the loading effects described above.

On the other hand, it is not feasible to initialize the 2D grid for a small number of nodes. Due to the randomization of nodes positions on the grid, nodes may form sparse clusters where there are no connections. Even if the network is connected, there may be bottlenecks which significantly delay propagation.

2.4 Torus/Sphere

The torus structure can be compared to a spherical cylinder without any edge nodes.

When running the simulation for the torus topology we observed a moderate performance for both the *Push-Sum* and *Gossip* algorithm.

2.5 Line

A simple line topology presents a set of nodes connected with each other in a line.

The line topology was the worst performing topology in the case of *Gossip* and *Push-sum*. In theory this result is not surprising. Since each node only has one or two connections, it consequently affects the ability for the network to efficiently propagate a message. This makes the message spread in a linear fashion, where each nodes is waiting for its for messages to arrive–until the final node is reached and convergence happens.

As one can observe from the graph, we were not able to run *Push-Sum* on the line topology for more than 1024 nodes. This is most likely the result of a node terminating in the middle of the line, dividing the network in two clusters–never to converge.

2.6 Imperfect Line

Similarly as the Line topology, an Imperfect Line also defines nodes connected in a line, but some nodes have an additional randomized connection with another node in the network.

The imperfect topology performed very well on all network sizes for *Gossip*, but not as well for *Push-Sum*. This makes intuitive sense, as *Gossip* convergence simply measures how quickly all nodes are reached, but *Push-Sum* depends upon mixing data from all across the network, and this topology is not as well-connected as the others and therefore does not mix as efficiently.

Another important observation is that this topology seemed to experience very little loading compared to more well-connected networks. This is perhaps due to the limited number of neighbors, which slowed down the rate of convergence and prevented message queues from getting overloaded.