Project – 2 Report (Bonus)

Team Members

1. Aseem Baranwal (UFID: 84127415)
2. Debi Prasad Das (UFID: 22650204)

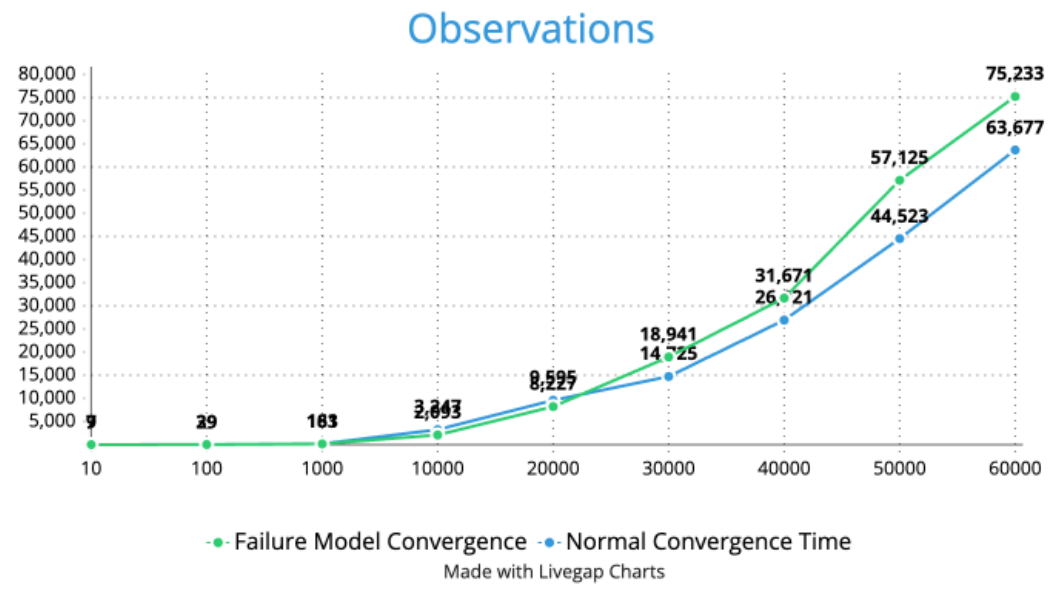
Code Execution Details

We should uncomment the function call **makeRandomNodeInActive** in the **server.erl** file at line 47 and compile the server file again.

Failure Model Implementation

Our implementation of the program has a state parameter named **isAlive** that we have used to handle failures. Every node's status of a process is tracked, whether it is active or not. The server and node system decide whether to include a specific node in the algorithm depending on the value of **isAlive**. To evaluate how failures affect convergence, we constructed one failure model. In our paradigm, a process alerts the server at any time by sending a message. The server would change **isAlive** = **false** at that time and randomly choose any process. So, we randomly cause one node to fail every round.
Although the operation may have been terminated immediately, doing so would have required further message exchanges between the actors to handle the return messages. With nodes arranged in a line topology, we tested our gossip algorithm failure model. Because the line topology converged the fastest of all the topologies for Gossip Algorithm in our prior experiment (see to **Project - 2(Gossip Algorithm).pdf**), we decided to use it.

Results



Nodes	Normal Convergence Time (in ms)	Failure Model Convergence Time (in ms)
10	7	9
100	39	29
1000	183	161
10000	3247	2093
20000	9595	8227
30000	14725	18941
40000	26921	31671
50000	44523	57125
60000	63677	75233

The time it took for the line topology to converge with and without the failure model is depicted in the above graph. For a lesser number of nodes, we saw that the failure model reduced the time required for convergence. In contrast to the pattern seen for typical convergence, the failure model takes longer to converge after 30000 nodes, as can be seen from the graph. This is in accordance with our existing observation that a scenario with lesser number of neighboring nodes converge quickly for Gossip Algorithm. According to our theory, the failure model has less nodes accessible for convergence when there are fewer nodes. As a result, convergence occurs more quickly. On the other hand, more message transactions between the server and the actors are required when the number of nodes rises due to failure checks, which ultimately lengthens the convergence time.