DOS PROJECT-3

CHORD PROTOCOL IN ELIXIR

Ratna Prakirana (UFID : 3663-9969)
Eesh Kant Joshi (UFID : 1010-1069)

Brief Description:

The aim of the project is to implement Chord Protocol in Elixir with simple object access service to prove its usefulness along with the resilience of the network with respect to failure nodes.

Analysis:

- The code is able to generate nodes and create the finger tables of the nodes joining the network.
- The code is able to update finger tables for all the other nodes in the network that are affected by the failure nodes.
- The code is able to correctly identify the node where our keys can be found from any node.
- The code is able to perform updating finger table of each node by calling Fix Finger Table function periodically as part of bonus section by stabilizing the network periodically.
- The code is able to maintain a successor list of size m (2^m = max possible number of nodes). Successor list maintains next m successors of a particular node.
- The code is able to reflect change in finger tables of all the remaining nodes who had any of the failing nodes in their finger table.
- In this implementation 15 percent of total nodes are failed after initial network is established.
- As part of node failure we established that this system is resilient because it is successfully reflecting the finger tables of the remaining nodes in the network, and find all the keys correctly.
- The algorithm performs efficiently and the average number of hops is always below the upper bound (log(n)) by a fair margin, where n is the number of nodes.

Interesting findings:

- The average number of hops are always less than log(n) to the base 2.
- Whenever all the requests are successfully completed, we print the average hop counts to deliver the message to the respected nodes.
- Overall, it takes greater number of hops to deliver a message to its destination node in case of node fails in the network.

- As the number of node failures in a network increases, the average number of hops also increase.
- The failure tolerance for the current system is about 15% of the total number of nodes in the network. For higher number of node failures, the chances of some messages getting delivered reduces greatly.
- The code runs for various numbers of nodes and requests. For the graphic visualization we have taken the number of requests to be 3.

No of Nodes	No. of Failure Nodes	Avg. no. of Hops with failure(15%)	Avg. No. of Hops without failure	Max bound on Hops
100	15	4.49	3.68	6.64
500	75	6.37	5.91	8.96
2000	300	7.71	7.19	10.96
5000	750	10.83	9.11	12.28
8000	1200	11.34	10.84	12.96
10000	1500	12.07	11.36	13.28