Distributed Operating System Principles (COP5615)

Project-3 Bonus

Group Members

JAYACHANDRA GUNTURU **SATYA NAGA AKHILESH IRRINKI**

What is working?

- Implemented Chord protocol - built a ring of nodes, applied SHA-1 hashing algorithm to generate unique m-bit identifier on each node and key and upon key search we are able to find the location of the key in O(log N) time where N represents the number of nodes.
- Implemented Failure model – Dealing with the stabilization of chord to find the location of all the keys when a node fails or dies in the chord.

Implementation:

- Using F# and Actor model by AKKA actor library, we have implemented Chord Protocol to efficiently find a key.
- Given the number of nodes and number of requests we generate unique m-bit identifiers for each node, place all the nodes in the chord, update finger tables for all the nodes, and lookup for all the keys in the chord.
- For the implementation of the failure model, we are removing a node from the chord network to simulate the failure of a node in real-time. shrink
- After the node is removed, we are updating the successors and predecessors about the failure of the node and thereby updating its immediate neighbors so that the ring is not disturbed.
- Then, we are updating the finger tables of all the nodes to stabilize the chord network.
- Once the chord is stabilized after the failure of a node the lookup for all the keys is performed as before.
- The rest of the process is the same i.e., keeping track of hop count and terminating when all the requests are processed printing the average number of hops taken for the nodes to find the key.

Execution:

- The .Zip file consists of F# script file named bonus.fsx
- Command for executing the file:

dotnet fsi bonus.fsx NumberOfNodes NumberOfRequests

How we have tested:

- We removed a node from the network and looked up all the keys.
- Then compared the average count of hops it took for finding the keys between the failure model and the non-failure model.
- There was no significant difference in the performance and the lookup for keys were successful proving that the chord stabilization was successful.

Number of Nodes	Number of requests	Non-failure Model	Failure Model
1000	100	24.826430	24.710320
5000	100	33.017076	33.012854
10000	100	37.418259	37.408892

Observation:

- We have observed that if more and more nodes fail/leave in the network, the system hangs because all the nodes are not able to find the keys and lookup infinitely for keys.
- As more and more nodes fail in the chord, updating the finger tables must be done in the order in which the nodes leave. It is an example of a write-write conflict.
- We can overcome the write-write conflict by only updating the finger tables when all the failure nodes have been identified.