# Project 3 Bonus Report

**Team members**

1. Prafful Mehrotra UFID: 1099-5311
2. Siddhant Mittal UFID: 6061-8545

**Objective Achieved**

- Introduced of failure model to simulate a simultaneous node failure situation.

**Experiment Setup**

- To introduce failure into our system we used an extra parameter called p (probability of failure). P is the probability which a node fails in our system
- We are going to perform lookups on a the perfectly stabled chord ring followed by failure of some nodes in the ring based on the parameter p. We again perform lookups on the chord ring and then analyze the difference in the average hop time.

**Advantage of the chord implementation done by us**

- We are running stabilization process as a concurrent and periodic process which keeps on setting the correct predecessors and successors for the nodes.
- We have a concurrent and periodic process named fix-fingers which keeps the finger table of the nodes in chord ring updated
- Hence these modules take care updating the ring when some nodes suddenly leave

**Enhancement in chord implementation and introduction of failure model**

- First of all, we will be introducing a new state in each node that is apart from having the predecessors and finger table, we will also store a successor list
- A successor list of some size r stores r successors of any node. Instead of storing only the immediate successor in the first entry of the finger table we also store first r successors in successor list
- There are two types of failure: voluntary failure and in-voluntary failure
- Voluntary failure is a failure when a node voluntary departs from the ring and In-voluntary failure is the case where node dies due to connection failure, transmission failure or infrastructure failure.
- Since In-voluntary failure can occur between lookups and are hard to monitor we will not be covering that case instead will focus only the voluntary departure of nodes
- Voluntary departure of node notifies its predecessor and successor before leaving and transferring its keys to its successor.
- All the failed nodes are stored in the ETS (erlang tool for local storage) of our program.
- We enhance all the modules that is find_successor(), find_closest_predecessing() and fix_finger() to check ETS every-time a call is made. This is similar to timout check used in the chord implementation of the paper's (https://pdos.csail.mit.edu/papers/ton:chord/paper-ton.pdf) 5th section.

**Important hallmarks which need to be highlighted**

- **How is lookup working after simultaneous Node Failure situation?**
  a. When a node leaves the system we notice that we used a notify function hence it has made the predecessor and successor known of its absence
  b. Since such absences of nodes are known by some of the nodes of the ring hence after the ring becomes stable that is stabilization and fix-finger work is done, the lookups are performed correctly and gives an average hope time of the order of O(logN)

- **How is stabilization process updating the successor list?**
  a. We enhanced the stabilization process which now also stabilize the successor list of each node
  b. This is done by a node N by taking successor list of its successor S. Removing last entry of that list and adding S and modified list to N's successor list

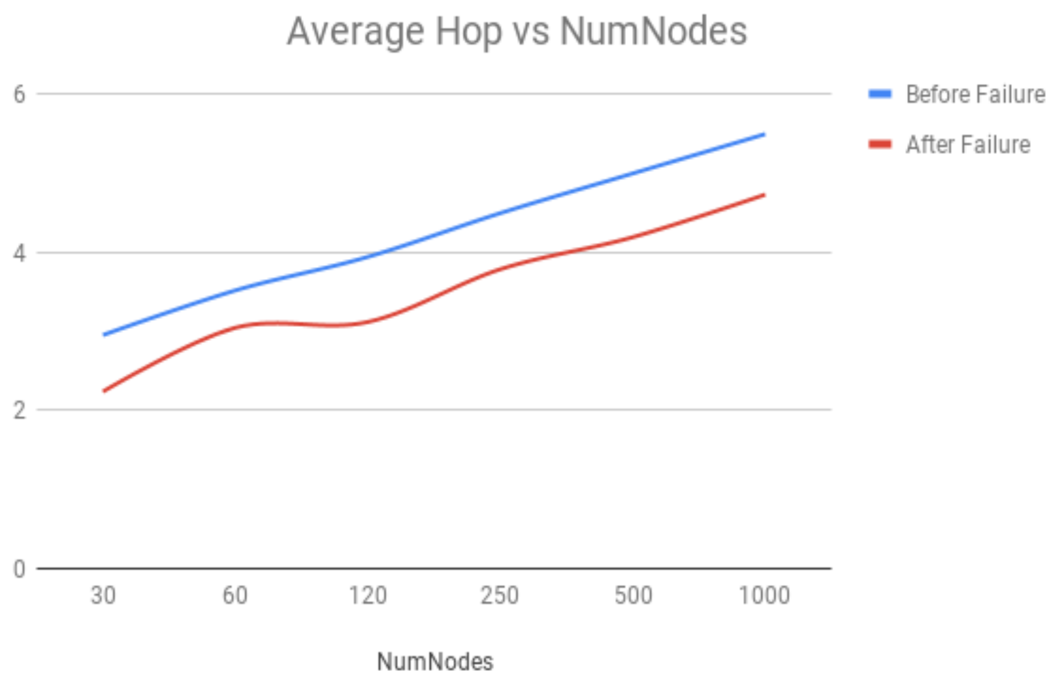
- **How is a lookup working correcting even before the chord ring is stable?**
  a. This is due to the fact that we are using a successor list and also an enhance use of finger table (see the next point for details).
  b. Even if the ring is not stable the lookups won't give wrong keys. They may give a poor average hops value but it is correct with a very high probability
  c. This is because we are maintaining a successor list that is if the entry of the successor is not updated by stabilize() or fix_finger() each node uses its successor list to update it successor.
  d. Our high probability argument in part c. holds because a node has a probability of failure p (the parameter we introduced in our failure model) thus the probability with which a successor list fails is p^r (r being the size of successor list). Hence a decent value of r minimizes the complete successor list failure probability. In our case r is 5 hence if p is taken as 1/5 that is every one node in five fails that means the probability of failure of complete successor list is 1/3125, which is quite small

- **How is a node handling the situation in which it's finger table still contains one or more entry of the failed node?**
  a. Suppose an entry i of the finger table contains a failed node and at the time of lookup (before stable chord ring is achieved) this entry is being used
  b. We enhanced this by checking the ETS to check whether this node is still alive or has failed. If it is failed, we return the i-1th value of the finger table.
  c. This help us as we will eventually hit a node which contains the correct lookup node in its successor list or finger table

**Observation and output**

- We observe that chord lookup correctness is not effected by nodes failure

- We also observe that after achieving a stable chord ring the node lookups takes the same time complexity before node failure that is O(logN). Below graph depicts this



- By this experiment we can have a more realistic analysis that is, in a real world application of chord the stable state of the chord is never achieved and nodes keep joining and leaving the system. Even after this case chord lookup should always hold its correctness argument even if it has to compromise its time complexity of lookups

- Output below shows lookups of random keys on a node before and after simultaneous node failure situation

```
Node key identifier used = 237811748
Lookups perform on keys = [342882692, 16241480, 601607496, 247657952, 261970472, 894179965, 840886032,
  637682866, 471318738, 191658006, 489101610, 560292933, 421751351, 483251616,
  686705479, 1032888151, 40344878, 560090548, 969983836, 277128805, 450470249,
  346280928, 543845388, 699620806, 975947012]
```

Output formant= {key identifier, number of hops}

| Before the failure situation | After the failure situation |
|---|---|
| {357134471, 2} | {357134471, 2} |
| {20136134, 4} | {20136134, 4} |
| {608042333, 3} | {608042333, 2} |
| {250404916, 1} | {250404916, 1} |
| {276782238, 2} | {276782238, 1} |
| {898182960, 3} | {898182960, 4} |
| {860837034, 3} | {860837034, 3} |
| {654914313, 4} | {654914313, 4} |
| {474459828, 4} | {474459828, 3} |
| {201330787, 5} | {217284549, 5} |
| {493480629, 3} | {493480629, 4} |
| {570198283, 3} | {570198283, 3} |
| {431521214, 3} | {431521214, 3} |
| {483720195, 4} | {483720195, 3} |
| {687064283, 4} | {687064283, 3} |
| {1038721808, 4} | {1038721808, 5} |
| {43916114, 4} | {43916114, 4} |
| {570198283, 3} | {570198283, 3} |
| {974388186, 4} | {974388186, 5} |
| {304674160, 1} | {304674160, 1} |
| {454808707, 2} | {454808707, 4} |
| {357134471, 2} | {357134471, 2} |
| {549096016, 2} | {555558980, 2} |
| {699634853, 4} | {699634853, 4} |
| {998803001, 4} | {1001932220, 5} |