# COP5615: Dist Oper Sys Princ, Fall 2018

Project 3

Date - 10/22/2018

---

## Team Members

Sourav Dutta (9863 3443)

----------------------------

## Coding environment

Erlang/OTP 21

IEx 1.7.2

**Linux Ubuntu 18.04** machine with 6 cores

-------------------------------------

## Part 1

## Instructions

The code is in directory project3
Command format: mix run lib/project3.exs <number of nodes> <number of requests>

e.g.

```
cd project3
clear && mix clean && mix compile
mix run lib/project3.exs 1000 4
```

**Parameters:**

1. **Number of nodes:** This denotes the size of the network and accepts an integer value. For simplicity the size of the network is converted to the nearest power of 2.

2. **Number of requests:** This is the number of key lookup requests each node in the chord makes. The requests are done per second for each node.

**Common details**

1. Maximum number of nodes tested is 10000 nodes for both the sections.
2. The last line of the output is the average hop count. The average hop count is well below log(m) where m is the number of bits needed to store the key/id of the nodes.

## Bonus

## Instructions

The code is in directory projet3-bonus
Command format: mix run lib/project3.exs <number of nodes> <number of requests>

e.g.

```
cd project3-bonus
clear && mix clean && mix compile
mix run lib/project3.exs 1000 4
```

Temporary failure model is implemented for the bonus problem. In this case, when the finger table is iterated to get the successor for a given key, the successor node might be temporarily unavailable with a probability p. This can happen because of network failures.

The probability of failure can be changed in function should_fail in file lib/utility.ex

This failure causes the network to skip that node and continue the search. The search will now take longer time to complete as it will have to search the finger table again or go through so other node in the network.

**Observations:**

The chord network was tested with failure rates 0%, 10%, 20%, 30%, and 50%. Seeing the below timings for different failure rates the number of hops does not rise as quickly as the error rate. Even when the finger table lookup fails 50% of the time, the average hop count is less than double than that of a perfect scenario. Chord topology for a distributed hash table seems very resilient to temporary failures and it handles then with very little increase in hops.

**Without any failure (project3 directory):**

```
mix run lib/project3.exs 10 5
{"Nodes", 16, "Average hop count", 1.925}

mix run lib/project3.exs 100 5
{"Nodes", 128, "Average hop count", 3.4640625}

mix run lib/project3.exs 1000 5
{"Nodes", 1024, "Average hop count", 5.01484375}

mix run lib/project3.exs 10000 5
{"Nodes", 16384, "Average hop count", 7.00250244140625}
```

**10% failure (project3-bonus directory for the below tests):**

```
mix run lib/project3.exs 10 5
{"Nodes", 16, "Average hop count", 2.225}

mix run lib/project3.exs 100 5
{"Nodes", 128, "Average hop count", 3.825}

mix run lib/project3.exs 1000 5
{"Nodes", 1024, "Average hop count", 5.3087890625}

mix run lib/project3.exs 10000 5
{"Nodes", 16384, "Average hop count", 7.43941650390625}
```

**20% failure**

```
mix run lib/project3.exs 10 5
{"Nodes", 16, "Average hop count", 2.1625}

mix run lib/project3.exs 100 5
{"Nodes", 128, "Average hop count", 4.09375}

mix run lib/project3.exs 1000 5
{"Nodes", 1024, "Average hop count", 5.7009765625}

mix run lib/project3.exs 10000 5
{"Nodes", 16384, "Average hop count", 8.01036376953125}
```

**30% failure**

```
mix run lib/project3.exs 10 5
{"Nodes", 16, "Average hop count", 2.5375}

mix run lib/project3.exs 100 5
{"Nodes", 128, "Average hop count", 4.4421875}

mix run lib/project3.exs 1000 5
{"Nodes", 1024, "Average hop count", 6.2552734375}

mix run lib/project3.exs 10000 5
{"Nodes", 16384, "Average hop count", 8.77510986328125}
```

**50% failure**

```
mix run lib/project3.exs 10 5
{"Nodes", 16, "Average hop count", 3.4}

mix run lib/project3.exs 100 5
{"Nodes", 128, "Average hop count", 5.7546875}

mix run lib/project3.exs 1000 5
{"Nodes", 1024, "Average hop count", 8.140234375}

mix run lib/project3.exs 10000 5
{"Nodes", 16384, "Average hop count", 11.2670654296875}
```