COP5615 – Fall 2021
Project 3 – Chord: P2P System and Simulation

# Team Members

- Mandar Palkar (UFID: 2140-6740)
- Siddhi Wadgaonkar (UFID: 9544-2212)

# What is working?

The system is designed to implement the Scalable Key Lookup using finger tables as described in the paper.

Chord Ring Formation and Stabilization

- As per the *Join* and *FindSuccessor* algorithms mentioned in the paper, each node asks a peer already in the ring to find a successor for it so that it can know its position in the Chord ring.
- Each node maintains a finger table of size m and periodically fixes its finger table. It also periodically starts the *Stabilize* and *Fix_fingers* methods at a fixed interval. This is achieved using Schedulers in our code.
- Once all nodes are added to the ring i.e. when everyone knows their successors, the ring is said to be stabilized. The maximum number of nodes for which we tested the Ring formation was 4,000 nodes.

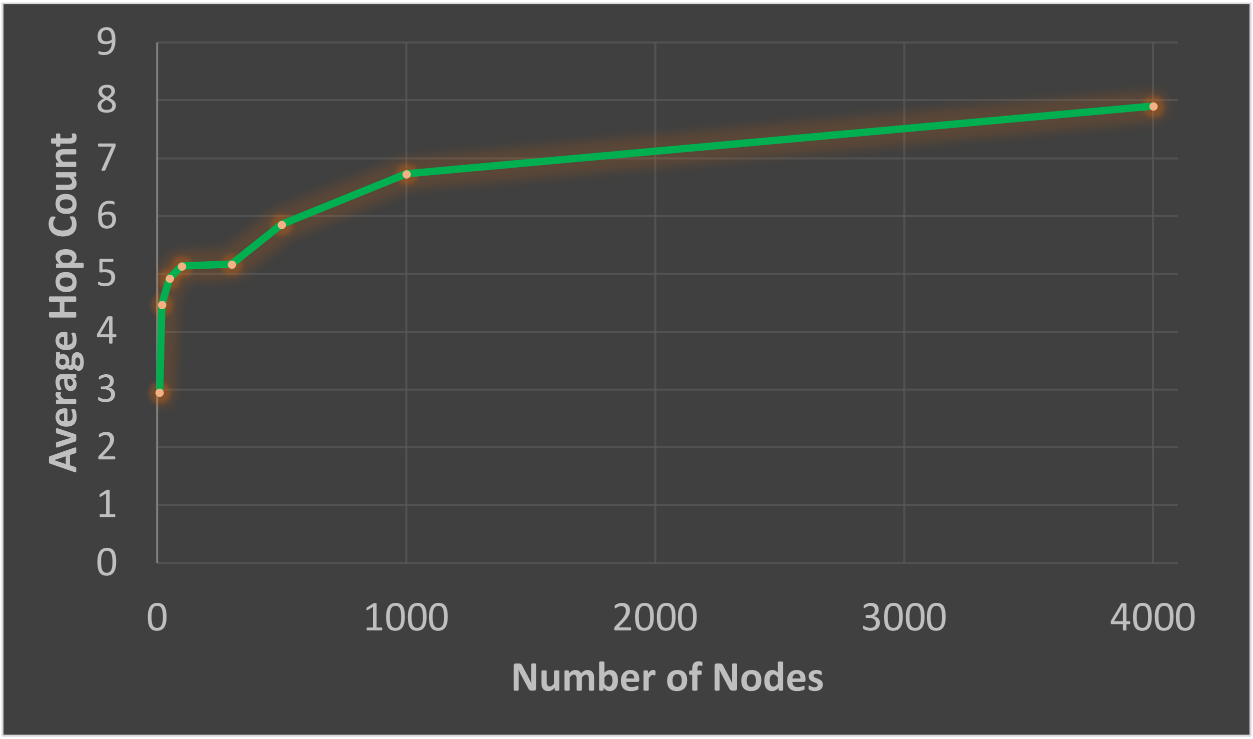
Requests Lookup

- Once all nodes are added to the ring, the Main Actor triggers the first node to start the key lookup operations. Since we do not have any global array that maintains references of all actors, we use the same Start Lookup function of the actor to pass on the Start Lookup command to each node's successor. It means that, once the first node is triggered to Start Lookup, it triggers its successor to do the same thing and similarly, the Lookup is started on all nodes.
- Once a position for the key is found, the node communicates it to the Printer Actor. This actor is responsible for collecting all the hop counts of all requests and calculate the average once all requests are completed.

# Largest Network

Maximum Chord Ring Formation = 4,000 nodes
Maximum Requests = 5 * 4000 = 20,000 requests

# Number of Nodes vs Average Hop Count



# To run

```
$ dotnet run <numNodes> <numRequests>
```

```
Example:
```

```
$ dotnet run 4000 5
```