COP5615 – Fall 2018
Project 3 – Chord

Alin Dobra

October 3, 2018

- **Due Date:** October 22, Midnight
- One submission per group
- Submit using eLearning
- **What to include:**
  - `README` file including group members, other requirements specified below
  - `project3.tgz` the code for the project
  - `project3-bonus.tgz` the code for the bonus part, if any

1 Problem definition

We talked extensively in class about the overlay networks and how they can be used to provide services. The goal of this project is to implement in Scala using the actor model the Chord protocol and a simple object access service to prove its usefulness.

The specification of the Chord protocol can be found in the paper *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications* by Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, Hari Balakrishnan. `https://pdos.csail.mit.edu/papers/ton:chord/paper-ton.pdf`. You can also refer to the Wikipedia page: `https://en.wikipedia.org/wiki/Chord_(peer-to-peer)`

The paper above, in section 2.3 contains a specification of the Chord API and of the API to be implemented by the application.

2 Requirements

You have to implement the network join and routing as described in the Chord paper (Section 4) and encode the simple application that associates a key (same as the ids used in Chord) with a string. You can change the message type sent and the specific activity as long as you implement it using a similar API to the one described in the paper.

Input: The input provided (as command line to your `project3.scala`) will be of the form:

```
project3.scala numNodes  numRequests
```

Where `numNodes` is the number of peers to be created in the peer to peer system and `numRequests` the number of requests each peer has to make. When all peers performed that many requests, the program can exit. Each peer should send a request/second.

Output: Print the average number of hops (node connections) that have to be traversed to deliever a message.

Actor modeling: In this project you have to use exclusively the actor facility in Elixir (i.e. `GenServer`) (**projects that do not use multiple actors or use any other form of parallelism will receive no credit**). You should have one actor for each of the peers modeled.

README file In the README file you have to include the following material:

- Team members
- What is working
- What is the largest network you managed to deal with

3 BONUS

In the above assignment, there is no failure at all. For a 20% bonus, implement node and failure models (a node dies, a connection dies temporarily or permanently). Dealing with failures is described in Section 5 of hte paper. Write a report describing how you tested that the systems is resilient and your findings.