Raman Sonkhla
rsonkhla@buffalo.edu, 50026724
Buffalo Wings

Simple Distributed Hash Table

1. Components

Content Provider
Provides basic interface between application and underlying DHT. The content provider as of now implements only insert and query function. Other than these no other interface is used to pass information from application layer to chord DHT algorithm. Insert and Query are provided to provide basic store and retrieve.

Communication Module
Provides network interface, for Chord DHT to work properly. It provides network read and write function calls.

Chord Module
Implements actual algorithm of Chord DHT. This component makes the content provider active component and give it ability to constantly listen to network messages and respond to them.

2. Chord Formation

For this a designated node (5554 in our case) is made responsible for configuring nodes so that new node can join Chord. Any new node wanting to join Chord sends "joining request" to 5554 and 5554 does the configuration. For this following steps are followed:-

1. New Node say 5558, sends joining request to 5554
2. This new node is to join as predecessor of any node if its hash lies between the current predecessor and the current node. The new node is to join as successor of any node if its hash lies between the current node and the current successor.
3. Node 5554 performs lookup for new node's port number. If it is able to find that the new node should join as its successor then it calls the node_join procedure below. If it finds that this new node cannot be successor then it sends the lookup request to its own successor and its successor performs the lookup.
4. Node_join (Successor of current node)
   a. (currentNode->next)->prev=NewNode
   b. NewNode->prev= currentNode
   c. NewNode->next= currentNode->next
   d. currentNode->next= NewNode

This is how new nodes join existing chord

3. Lookup

For every insert/query request initiated by some node, that node will first perform a lookup to find final node where insert/query is supposed to happen. Once lookup returns with a valid nodeID, requester node directly sends insert/query request to destination.

```
Lookup(KeyID)

        if(myPrevHash > myHash)
        if(keyID > myPrevHash || keyID <= myHash)
                return myNodeID
        if(myPrev < keyID && myHash >= keyID)
                return myNodeID
        else
        successor.lookup(keyID)
```

Above lookup node is same as the one given in paper, with just one modification to cover boundary condition.

4. Server or Listener Thread

To implement chord DHT, we need to make content provider as active component so that it can listen to network messages sent by other nodes. For this a server or listener thread is setup so that any incoming request like, node joining, previous or successor modification, query or insert request.