Book Finder System on a DHT

Shrivardhan S. Bharadwaj(5478571)

Himanshu Kharkwal(5488345)

University of Minnesota

INDEX

1. Purpose

In this programming assignment, we are implementing a Book Finder System using a distributed hash table (DHT) based on the Chord protocol. Using this system, the client can set and get book's information (title and genre for simplicity) using the DHT.

2. Method

2.1 System Design

We implemented three components:

1. Client : The client is responsible for setting book titles and genres to the system as well as getting a genre from the system with a book title. It does it in the following way:
    A. The client asks super-node for a random host-node.
    B. The client asks host-node from the above step to set book titles and genres to the system as well as get a genre from the system with a book title.
2. Super-Node: The super-node receives jobs from client as well as host-nodes.
    A. Super-node receives a job from host-node to add itself into the DHT network. Super-node maintains a list of host-nodes along with their IP address and Port number and randomly assigns a host-node to the node requesting to join the network.
    B. Client asks for a host-node to the super-node to assign it some jobs. The super-node returns it a random host-node.
3. Host-Node : The host-nodes execute tasks such as:
    A. It asks super-node to assign it another host-node to join into the DHT network as well as to update its finger table.
    B. It sets the genres of book titles into the DHT network according to their hash values(SHA-1). It also returns these genres when client query the network for some particular book title.

2.2 List of Files

1. config : User should provide the Super-Node IP address, Super-Node port number, Host-Node port number and DHT entries(value of m).
2. Client.java :
    I. Asks the user to input directory which contains all the files to be computed.
    II. Requests super-node to give a host-node
    III. Requests host-node to run set and get jobs.
3. SuperNodeServer.java
4. HostServer.java
5. HostClientHandler.java : receive get and set jobs from a client, and sends them to HostNodeHandler.
6. SuperNodeClientHandler.java : Returns a random host-node to the client to perform get and set operations.
7. SuperNodeHostHandler.java : Maintains a list of host-nodes along with their IP address and Port number. It also implements Join and Post join functions.
8. HostNodeHandler.java :
    I. It implements Chord algorithm.
    II. Asks Super-node for a random host-node to join the DHT network.
    III. Receives set or get jobs from the client and adds the genres of book titles according to their hash values(uses SHA-1 hashing function).
6. Make : Compiles all the java files
7. Run : Starts the framework
8. HostService.thrift
9. HostClientService.thrift
10. SuperNodeClientService.thrift
11. SuperNodeHostService.thrift

Note : Any number of nodes can be added to the network without prior declaration. We have attempted the transfer of keys but have commented out that part because in discussions it was mentioned that we only need to implement the set and get operations once DHT network has been formed completely.

2.3 Logs generated

1. Updated finger tables of each node in the DHT network are printed after the post join of a new node.
2. During get and set operations the system prompts the user whether they want routing information or not. If “yes” it prints out hash key of the book title, all the list of nodes visited and the node where the book title was stored.

3. How to run

1. SSH to client and server machines:

   $ ssh <client>

   $ ssh <server>
2. Download and extract the tar to some location in client

   $ tar -xzf <tarfile>
3. Edit config file and add Super-Node IP address, Super-Node port number, Host-Node port number and DHT entries(value of m).

   $ nano config

   > SuperNodeIp:csel-kh4250-06

   > SuperNodePort:8010

   > HostNodePort:9082

   > DHTEntries:5
4. Execute Make file at client for compiling all thrift files and Java Code

   $ sh Make

5. Execute Run file at each machine to start the framework. Note: Start Super-node and host-nodes to from a DHT network before starting the client node to add information into the network.

   In Super-node machine:

   $ sh Run SuperNodeServer

   In Host-node machines:

   $ sh Run HostServer

   In client machine:

   $ sh Run Client

   This will open a menu and prompt the user for the following inputs:

   > Menu

   > 1. Put Keys from File

   > 2. Put keys individually

   > 3. Get keys

   > 4. Quit

4. Test cases

Negative cases:

1. Case : No data present to enter into the network.

   Output : Prints “No data present” and open up the start menu again.
2. Case : User asks for the genre of book title which is not present in the network.

   Output : Prints “Book title is not present in the network.” and opens up the start menu again.
3. Case : If no host-node is present in the DHT network and the user tries to run get or set jobs.

   Output : Prints “No nodes are present in the network.” and opens up the start menu again.

Positive cases:

4. Case : Book Titles to enter from a file

Output : Reads the file and enters all the Book titles into the network and prints. “Book Titles successfully entered into the network” and open up the start menu again.

5. Case : Only one book title to enter

   Output : Enters the book title into the network and prints. “Book Title successfully entered into the network” and open up the start menu again.

6. Case : User asks for the genre of book title present in the network.

   Output : Prints the genre of the Book title and open up the start menu again.

6. References

https://itss.d.umn.edu/services/file-storage

https://www.geeksforgeeks.org/java/

http://www-users.cselabs.umn.edu/classes/Spring-2019/csci5105/pas/pa2/pa2.pdf

https://thrift.apache.org/tutorial/java

https://pdos.csail.mit.edu/papers/chord:sigcomm01/chord_sigcomm.pdf