Design document

Name: Ning Zhang
CWID: A20336916
Department of Computer Science, Illinois Institute of Technology

The DistributedHashTable system is developed by JAVA and the remote communication is based on sockets and multithread. The whole system is entirely distributed in the sense that each peer is also a server which is capable of processing PUT, GET and DELETE requests from multiple clients as Fig1.

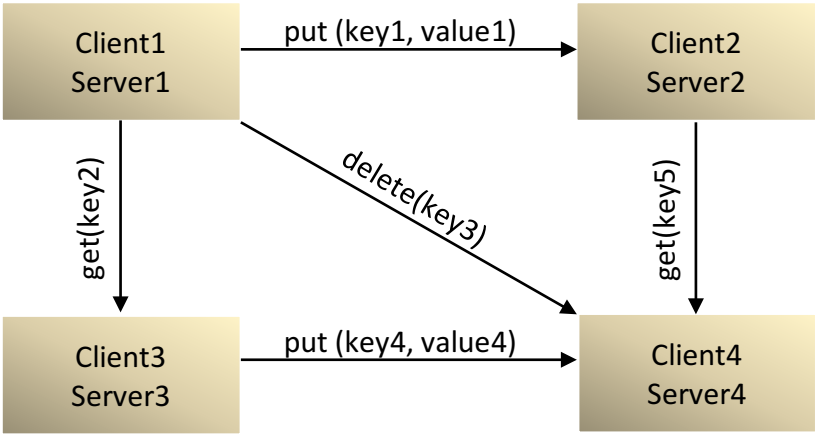

Fig1. DistributedHashTable system

Config File

At the beginning of running this DistributedHashTable system, the system will initialize the servers by reading the config file. The format of this config file is as Fig 2.

config.txt

```
:127.0.0.1:5555
:127.0.0.1:5556
:127.0.0.1:5557
:127.0.0.1:5558
:127.0.0.1:5559
:127.0.0.1:5560
:127.0.0.1:5561
:127.0.0.1:5562
```

Fig2. Format of config file

The config file contains ips and ports of all servers. And every line contains a server's ip and port which are separated by ":" symbol. When every client and server is established, it will read this config file and know all the servers.

Seletion of Servers

Every server keeps a hash table to store the key and value. Before every client send a command, the system will use MD5 algorithm and mod operation to decide which sever the client will send this command to. Firstly, the system will use MD5 algorithm to transfer the string key into the 16-byte hex. Then using 16-byte hex to mod the number of servers for getting which server the client will send the command to as Fig 3.

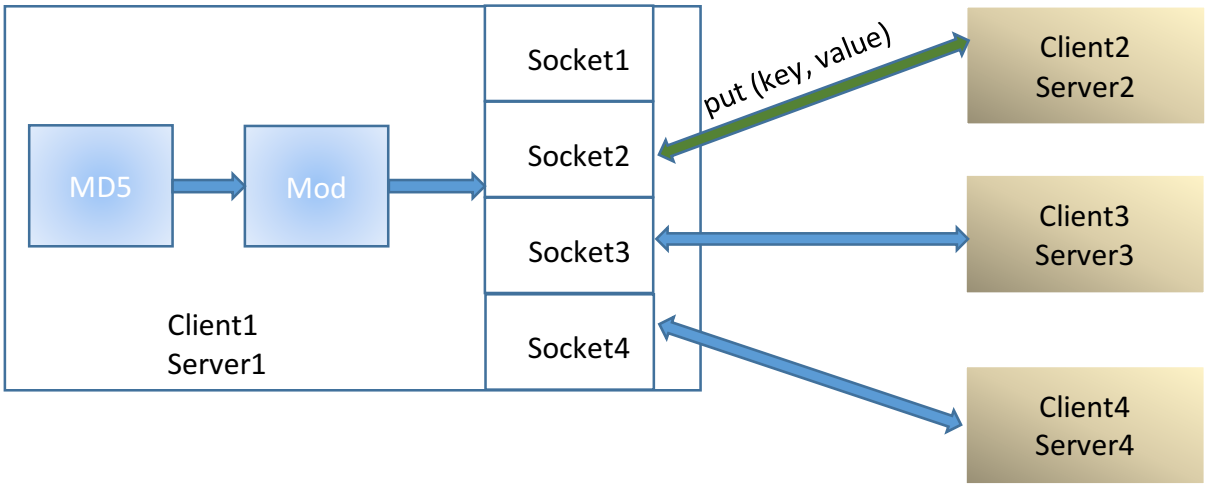

Fig3. The implementation of this DistributedHashTable system

Socket Cache

As the Fig 3 shows, the Client1 selects the Server2 to send the PUT command. And in order to improve the communication efficiency, we use a hash table to store the sockets connected to the servers as the socket cache. By this method, we reduce the cost of establishing and releasing the socket connection.

Synchronization of Threads

Because every server has a hash table and this hash table may be simultaneously written and read by more than two clients, we must handle this synchronization of

these client threads. So we use the concurrenthashtable in JAVA to implement the hash table for handling the synchronization of these client threads.

MD5 algorithm

The MD5 message-digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte)hash value, typically expressed in text format as a 32 digit hexadecimal number. MD5 has been utilized in a wide variety of cryptographic applications, and is also commonly used to verify data integrity. So we use this algorithm and mod operation to assign the keys on average to the servers.

Running in the cloud

This system is designed to function in the cloud environment in that each peer can run independently on different machines. As discussed previously, the communication is entirely based on sockets/TCP/IP. Specifically, we assign each peer a PID based on ip address to ensure its uniqueness and this PID is used for internal system communication. The basic functions including put (key, value), get(key) and delete(key) which are verified in the system.