Technical Impression

Weiran Guo

This project is different from project 3. As paxos is implemented largely different from 2-phase protocol. As a result, I tried to redesign the entire project at the beginning. The paxos is hard to understand especially when I read the paper and materials, it is less intuitive. It takes me a while to understand and design the project. To make sure our server can be run on any machines, and clients can access to any servers except from my version, I removed RMI functionality and redesigned the TCP version of KV-store. So at the beginning, all servers will know the address and port of other servers, so they can communicate in the Paxos algorithm. For clients, I added the switch server, sleep option, so we can mimic the real-world scenario that sometime the server might fail or disappear. The timeout is added each time we do not receive a reply from our request.

The algorithm mainly has two phases. First, a client connects to one of server, and send a request to server. If the request is GET which will not change the database, then Paxos will not start. Paxos starts when a put or delete request is sent. For each server, we have a listener, which is a while loop keep listening to new clients and requests, for put and delete command, we will forward the command to the proposer, and proposer is running on a thread concurrently, and checking if new message is arriving in the queue. Once the message arrive, it will pop the command and try to send prepare message to the acceptor, after majority approves, it will send accept message to the acceptor, also wait for majority to approve. Then it will change the kv database. For all acceptors, it will change its database after check that accept message is valid. Also, we can make server randomly fail to mimic the real world situation.