GIGAPAXOS/RSM APPROACH

Joshua Gu

ID: 34466505

To achieve crash fault tolerant and totally ordered consistency across replicated Cassandra applications, I used the gigapaxos approach where I implemented the 3 functions execute, checkpoint, and restore. To ensure each of these three functions were atomic, I wrapped them in synchronized(this) for thread safety. In the execute function, the application would send the String request to the Cassandra database by the keyspace. My implementation assumes there is just 1 table in the keyspace, which is as far as the tests go as well. After the request is successful, the execute function returns true. Any consensus and crash recovery is handled by Gigapaxos, including the recovery of executions that alive replicas processed.

To implement checkpointing and restoring application state to a checkpoint, in the checkpoint function, the application queries the database for all rows in the table. The rows are stored in a hashmap of Strings as keys and values. The key is the combination of the primary key name and the value of the primary key. The value is the concatenation of all key value pairs associated with the primary key. The checkpoint function finishes by returning the JSON string representation of the hashmap.

When restore is called, the JSON string is recovered and converted back into the hashmap. We then iterate over all key value pairs and call the UPDATE and SET request to set all rows in the database to the version we have in the hashmap. With this setup, all tests pass.