CSE 511 Programming Assignment 2: Replicated Linearizable Key-Value Store

Out: November 8th, 2025   Due: December 4th, 2025

Bhuvan Urgaonkar

1 Overview

In this assignment, you will implement a distributed key-value store, which will replicate data across multiple machines. You will implement two variants of your key-value store, both of which will offer linearizability: (i) a blocking protocol and (ii) the non-blocking ABD protocol covered in class. There will be an experimental evaluation component wherein you will compare certain aspects of the performance of your implementations (i) and (ii). You can use the RPC-based client server architecture developed in project 1, for your ease of implementation. But that's not a project requirement and you are free to choose any communication protocol for your client server interactions.

2 Design and Implementation

Review relevant lecture materials to make sure you understand the memory consistency model called linearizability. Then you need to understand the ABD algorithm, that implements a linearizable read-write shared memory over a distributed asynchronous system. You will be working with majority quorums for reads and writes. We denote the sizes of these quorums as R and W, respectively. The number of replicas is given by N. You need to be very sure about each aspect of the ABD algorithm and understand why is it a non-blocking or wait-free protocol. All these concepts will not only help you in the implementation of ABD algorithm, but also prepare you to better handle the next part of your project.

In addition to ABD, we would also like you to construct an alternate protocol that offers linearizability but may cause a client to be blocked because of the failure or slowness of another client. By design, the ABD algorithm cannot lead to such a scenario. We will simply refer to this alternate protocol as "the blocking protocol." As a hint for this protocol, when doing a get(), a client may issue N "acquire lock" requests and wait for the first R "lock granted" messages from R servers. Following this, the client may send "read" requests to these R servers and then use their timestamps in a way similar to the ABD algorithm to determine the correct value. You are to design the details of this protocol and describe it in your report.

3 Experimental Evaluation

You will be using multiple machines to create a distributed system of N servers. These could be AWS instances or CSE machines. The experimental evaluation will consist of two parts:

- Measure and report the performance impact of increasing N (number of servers) from 1 to 3 to 5 for both implementations. Performance metrics of interest include sustainable throughput (requests/sec) and latency (median and 95th per- centile) at the sustainable throughput. You are to report latency numbers for gets and puts separately. Use two get/put ratios, one with 90% gets and 10% puts and the other with the fractions reversed. The number of clients must be high enough so that throughput saturates - you will determine this empirically.
- Measure and report the impact of a client crashing, on the performance for the rest of the clients, with the two protocols.

4 Submission and Grading

You will submit your well-documented code and a short report containing the experimental evaluation from Section 3. You will also describe the blocking protocol that you developed based on the hint offered above. You will be evaluated by the TA using a combination of:

- The correctness of your code (50%) - we will use a combination of running your code against some test cases and visual inspection (aided by your own documentation). Your submission should completely describe how to compile and run your code. Also list any shortcomings in your code you are aware of and would like us to know.
- The quality of your code (20%) - is your code well-documented and easy to understand? does it follow good programming practices (or, perhaps more importantly, does it use some undesirable practices)?
- The quality and correctness of your experimental evaluation (30%).

5 Some final remarks

- **IMPORTANT: We will be comparing your codes for similarity and all parties involved in copying will receive a 0 grade on the project 2 at the very least and an F grade in the course in the worst case. Outsourcing your code to an external entity is considered cheating. You will also be reported to the college of engineering disciplinary committee.**
- The following are your friends (and your grades' as well): make, gdb, github, good documentation, a decent plotting tool.
- It is very important that you start early. Please seek as much as help as you need from the instructor and the TA.
- Good luck and happy coding!