HyperKV

Shikhar, 23CS30049

Project Overview

HyperKV is a fast, in-memory key-value store written in C, following the RESP. It supports various data types—strings, hashes, lists, and sets—with a wide range of commands for each. The project includes both server and client components, and a utility script for quick setup.

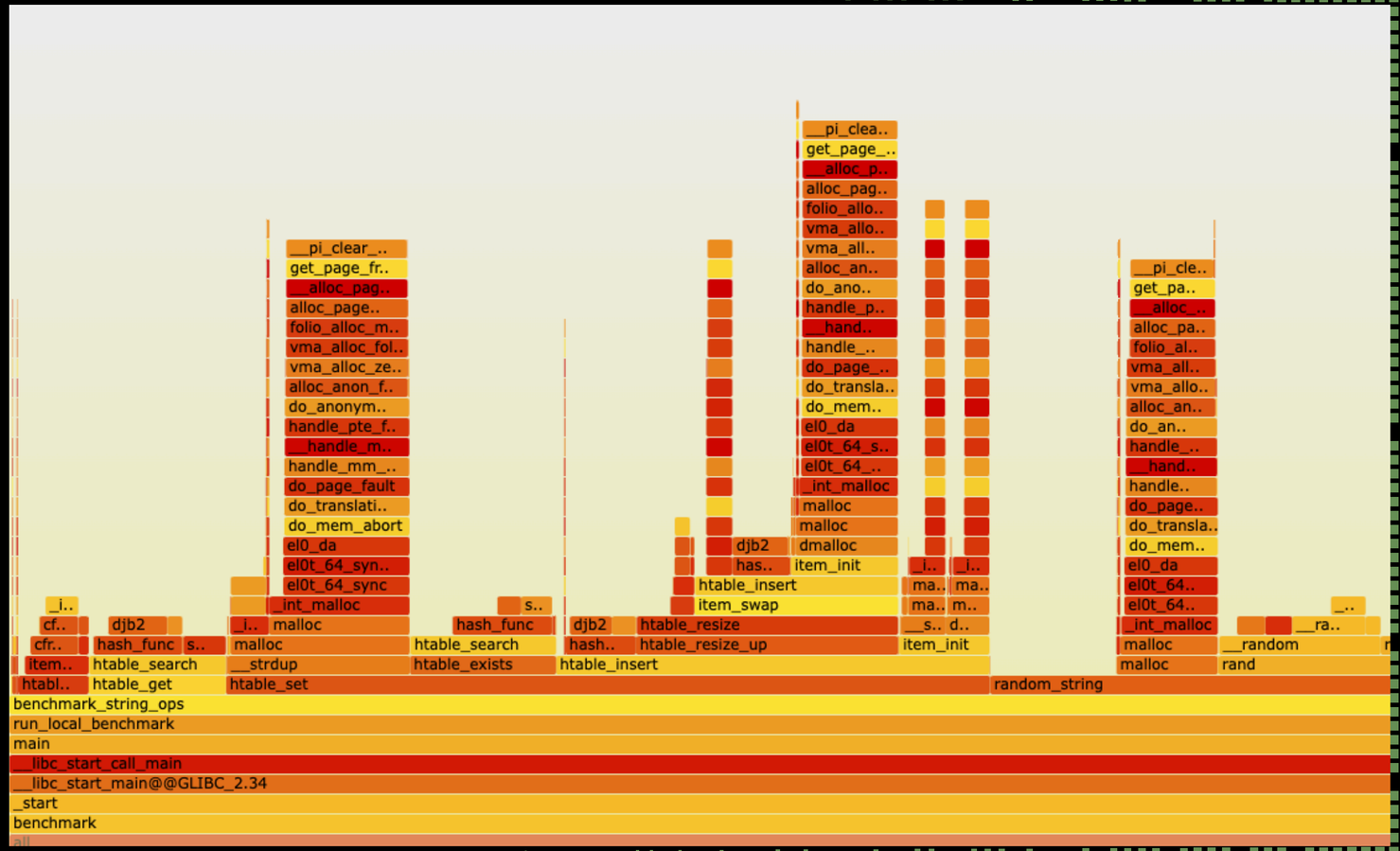
Designed for performance and simplicity, HyperKV offers benchmarking tools to compare its speed with Redis across different operations. It’s easy to build, test, and extend, making it ideal for learning database internals or building lightweight caching systems.

Technologies Used

HyperKV is written completely in C.

I have used Make for streamlining the build process.

I have used Flamegraphs to analyze the performance bottlenecks. On the right is a Flamegraph for a benchmark running one million GET/SET operations.



Tests

- Writing thorough tests in C, especially for edge cases, and debugging low-level issues without high-level tooling was meticulous and time-consuming. In the end, I was able to do exhaustive unit testing using the *miniunit* library

CI/CD

- I have written a Github Action which lints the code, builds the project and runs all tests. The Action is ran whenever a commit is pushed on the master branch or there is a Pull Request on the master branch.

Challenges Faced

- Writing an in-memory store in pure C required careful handling of memory allocation and deallocation to avoid leaks or segmentation faults.
- Implementing custom, high-performance data structures (like hash tables, linked lists, or sets) without built-in libraries was complex and error-prone.