CS425 MP1 Group 44 Distributed Grep

Shichu Zhu(szhu28), Fan Shi(fanshi2)

Overview and System Design

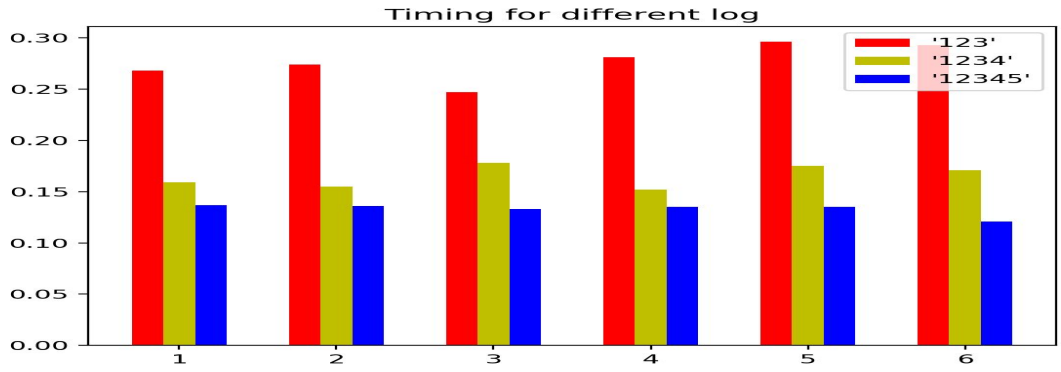
We use Google's gRPC package in Go to implement the Server-Client model. The services are defined in protocol buffer which we can used to generate portable programming interface for remote procedure calls (RPC).

Our distributed system is highly structured and can be easily scaled for new services. We defined three services in protocol buffer and implemented corresponding server function. The three services are System configuration, Grep command and Remote server close. Additionally, we implement VM_setup module enabling us to deploy code change, spawn/terminate server process on each VM with more convenience. These additional functionalities will be greatly beneficial in developing or debugging future MPs.

For each functionality, we create one client module that compiles to a executable command in GOPATH. The distributed grep is named as 'dgrep'. We also implement a library for general use. In this way, we can easily add new client side features.

Measurement

We use the given testing file to measure our distributed system. We connect to 4 VMs and grep three patterns ("12", "123", "12345") each for 6 times. We use the grepping line number as the metric of frequency. For this measurement, we use vm-10 as client, vm-1 ~ vm-4 are server. Y axis is the running time in seconds.



"123" : (49609 lines total, average time: 0.276s , stddev: 0.018)
"1234" : (49609 lines total, average time: 0.165s , stddev: 0.011)
"12345": (236 lines total, average time: 0.132s , stddev: 0.006)

Unit Test

We test the distributed grep by comparing the sorted matching line numbers in all files. The output and generated by the 'grep' and 'dgrep' binary command, then stored in a file, sorted and then are compared using the 'diff' command. We test the output on both the given dataset and the log files generated by the VMs themselves.

In conclusion, we are able to build, connect, grep and close ten servers each in one command. All the services are separated so that we can easily expand the distributed system and add more functionality in the future.