

MP1 Report

Group61: Gan Yao(ganyao2), Linxi Li(linxili2)

Design

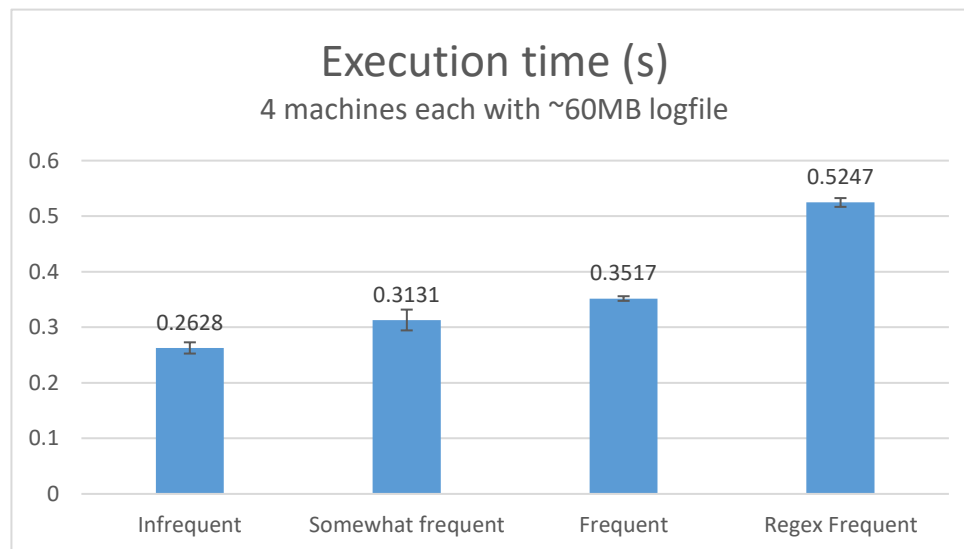
Our MP1 was designed based on the web socket client-server model and was implemented mainly via the websockets library in Python.

The design followed the following paradigm: each and every VM has server running on it; Once we execute the client program on any one of VMs, it takes the inputs from the user, and send this input, which is a Linux command, to all the servers; Servers executes these commands on the local terminal via the subprocess library in Python, directly send these results to the client. The user will then see the results of the command.

Our code is fault tolerant, if one of the VM fails, or one of the grep command returns an error or cannot return, the code will pass the server and move on to the other server. Considering the performance, we called asyncio library in python to parallelly send the messages to all VMs. The server was designed as once started unless close the ssh terminal or keyboard interrupted, it will not stop.

Unit test

Since our unit tests were completed before the release of demo log file, we generated our own log files, each about 50000 lines, 3MB. We tested queries for rare/somewhat frequent/frequent patterns in one/some/all files. So in all 9 test cases. Our system successfully passed all tests.



Performance

We measured queries execution time for infrequent(~2000 line in all files), somewhat frequent(~55000 lines), frequent(~550000 lines), frequent(~800000 lines) regex patterns across 4 machines(vm1.log to vm4.log). Each data point consists of 10 trials. We use the 'real time' produced by linux command 'time', which is the whole execution time of client program. As shown in plot, the delay increases as the pattern frequency grows. The increase in delay caused by frequency growth is more significant when frequency is low. And regex has huge impact on execution time.