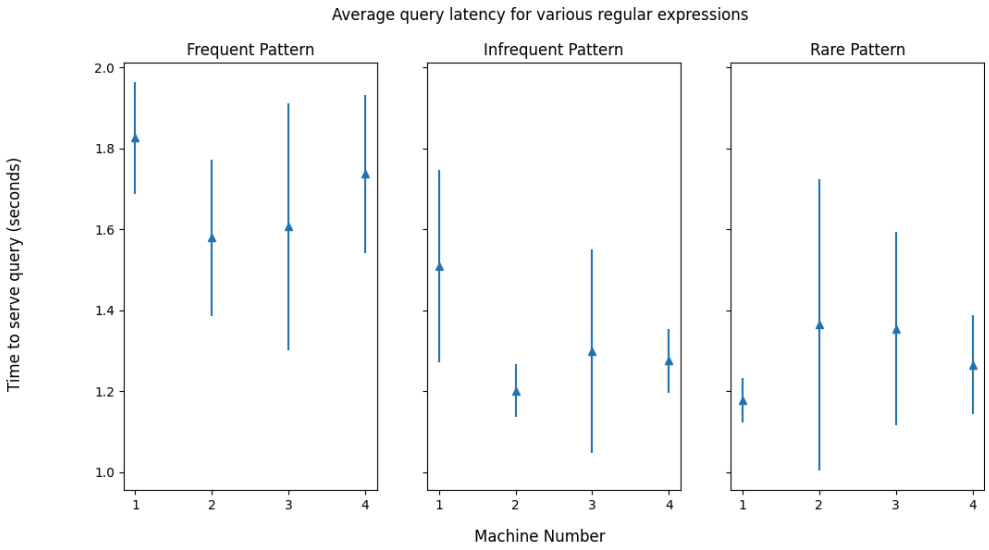
Design/Algorithm:3

Our algorithm is designed around using SSH tunnels to perform a `grep` search over multiple machines. We chose this implementation since this does not require all the machines to run a client.

On any of the machines (which we call the search machine), a user can run the executable and enter a regular expression to search. The search machine then opens SSH tunnels to every other machine, including itself, and executes a `grep` command on the log files of the respective machines. The `grep` outputs are sent back over the SSH tunnels to the search machine, which then counts the number of lines in each `grep` output and the total number of lines cumulatively. The search machine also saves the outputs of the most recent regular expression search on local files.

Unit Tests:
We have written unit tests to validate the correctness of our program. We generate test log files for our unit tests and assert results for frequent, infrequent and non-existent queries on these logs. The tests are written using Golang's testing package.

Average Query Latency:



We report the average query latencies from each machine when there are 60MB log files in 4 machines in the figure. We also explore three different search patterns, each of which occur in different frequencies. The average query latency across all search patterns and machines is 1.432 seconds. The latencies for frequent, infrequent, and rare patterns are 1.687 seconds, 1.321 seconds, and 1.290 seconds respectively. The difference in the time taken for the rare pattern search vs frequent pattern search shows that there must be a minimum time needed to establish an SSH connection and to run the `grep` command. The patterns then differ in the time taken by the `grep` command to search and to then send the output over the SSH tunnel.