Sayan Sisodiya (sayans3)
Vraj Patel (vrajsp2)

Design

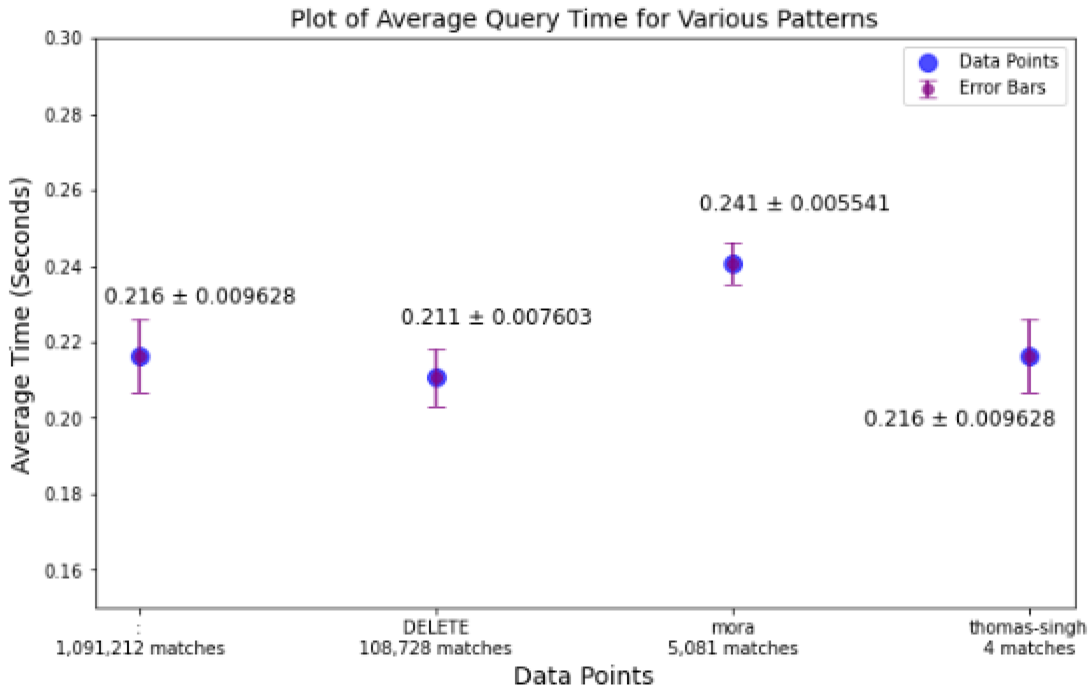
The design of our distributed grep querier is that each VM has its own client and server goroutine. The client goroutine is the one that sends out the grep query to all machines; more specifically, a client goroutine will query all the server goroutines of online servers concurrently with information on the grep query to run, then that same client will collect all the responses and print them to the terminal.The server goroutine on each VM accepts connections from client goroutines; it spawns a new goroutine for each client to handle multiple connections at once. In addition, before the client sends out requests to all servers, it pings all 10 servers in parallel to check which ones are online and only queries the online servers with the grep command to run.

Unit Tests

The unit tests we wrote include tests of frequent, medium, and rare patterns across all 10 machines. Other tests test patterns only present on one, some, or all machines. We also have a test for regex patterns. All tests run on log files that are randomly generated.

Metrics

Data was collected using the "real" result of the "time" command in bash. Only VMs 1-4 were running for these tests, so only logs 1-4 were queried. We used patterns ":" (1,091,212 matches), "DELETE" (108,728 matches), "mora" (5,081 matches) and "thomas-singh" (4 matches).



Based on the measurement results, we do not see any real correlation between the frequency of the pattern and the time it takes for our distributed grep to run; the difference between the slowest and fastest runs across all patterns was only 0.063 seconds. This makes sense since the grep command still has to parse the entire log file, so the frequency of the pattern doesn't affect the speed of grep. Also, these tests were run with the -c flag, so the output wasn't printed to the terminal. This would have affected the average times, since longer outputs take longer to print.