MP1 Report

Design

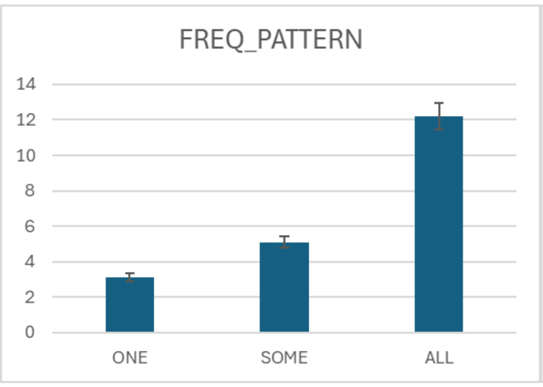
We use a client-server implementation where the client takes the grep command and sends it as is to each server, using different worker threads. Server listens for client messages and grep its own log file, returning the results to client. Grep results are stored in output/output.i.log. The program assumes that log files have the format machine.i.log where i is the server number. Grepping a crashed server will pass silently.

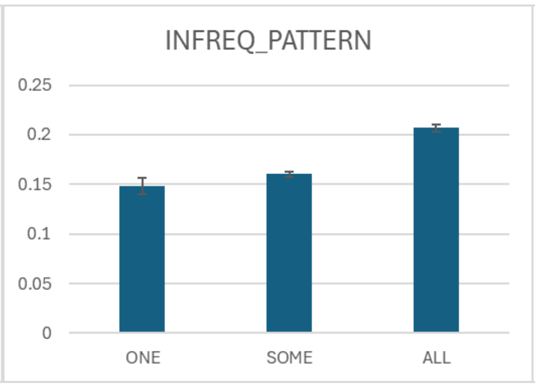
Unit Tests:

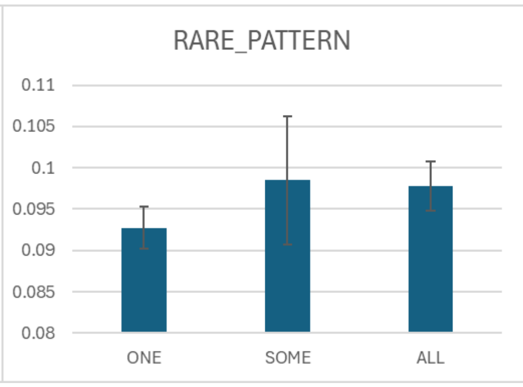
We randomly generate log files and, if flag is set, randomly place a fixed number of predefined patterns into the files. File is bumped to 60mb by more generated lines. Tested cases listed in homework specs by comparing results to unix grep results.

Average Query Latency

We grep 4 machines each containing the unit test log file we generated. Same as unit test, we tested 9 cases where each is a datapoint averaged across 5 trials, and calculates standard deviation.







Y axis is time in seconds.
A frequent pattern occurs 10^5 times which takes roughly ⅛ of the logs. Infrequent patterns occur 10000 times in each log and rare patterns occur 1000 times.
ALL means the patterns appear in all log files, SOME means they appear in some files and ONE means only one file has the patterns.
In this plot, the average query time across five trials for each pattern is depicted with error bars representing the standard deviation, ensuring that the results reflect both the average performance and the variability across runs.

Conclusion

The runtime of each query thread when grepping frequent patterns would be dominated by server grep and sending results back to the client. Runtimes are proportional to the number of files containing patterns. This implies that our implementation is not parallel, presumably blocked by the client waiting to receive response and the server grep processing its results. Infrequent patterns are instead dominated by result aggregation (writing log files etc.) so runtime does not scale much with the amount of data. Rare pattern runtimes are mostly just establishing connections, therefore it does not scale with the amount of data at all and exhibits high variance. Currently the program focuses on correction, but in the future non-blocking ways of receiving results from the server could be looked into.