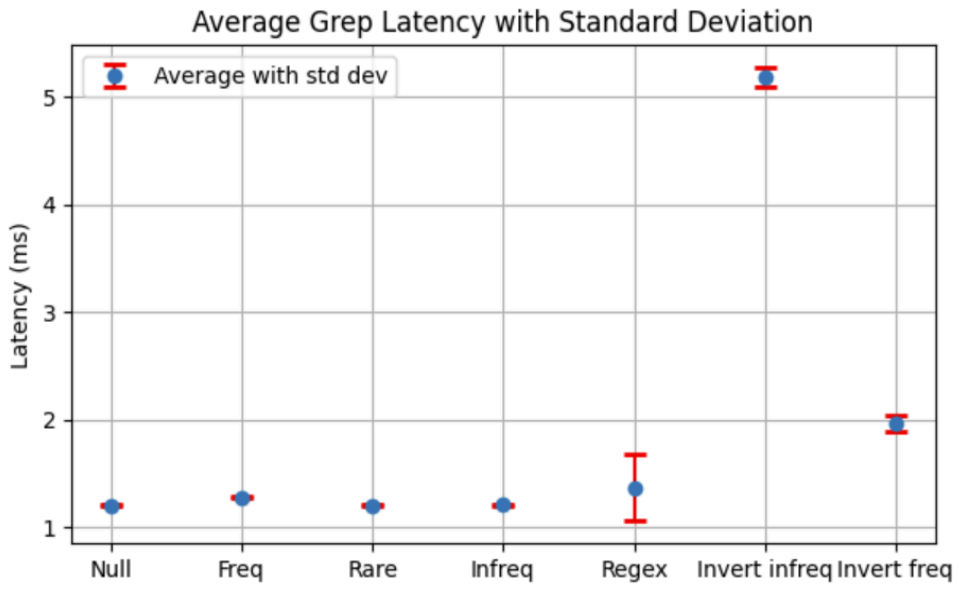
Distributed Systems (CS 425) - Fall 2024 - MP 1 Report

Akhil Sundaram(akhils7), Anurag Choudhary (anuragc3)

Algorithm: Implemented a socket-based system in Go using goroutines for concurrency. The system consists of: a Parser and a Listener. The client sends grep commands to multiple listeners, each executes the command and returns the result. Each socket connection is handled by a separate goroutine, and runs parallely. Once a listener returns the grep output, it is first written to a temporary file and then to the final file. Access to the final file is synchronized using a mutex to avoid race conditions. We also calculate the grep counts and store this information for final output printing.

Unit Tests Coverage: Frequent ("PUT"), Rare ("huffman"), Somewhat frequent patterns("homepage"), Regex "-E pad*", Invert Pattern ("-v hostname, -v GET")

	Null Matches (baseline)	Freq Pattern (PUT)	Rare Pattern	Infreq Pattern	Regex	Invert infreq pattern	Invert freq pattern
Avg latency	1.2081	1.2832	1.2088	1.2124	1.3725	5.183	1.9684
SD	0.0025	0.0083	0.0056	0.00346	0.30965	0.088	0.07417



Findings - We have run a Null match to find how much time our program takes without including the grep operation(since no matches are found). Once that is done, we find the avg latency and SD for each of the following cases. We can see some interesting data for the last 3 data points. Regex data is skewed as one trial took 0.6s over the other samples. Invert takes a lot of time for infreq and freq both (more for former), as it has to check all lines to see which doesn't contain the pattern specified.