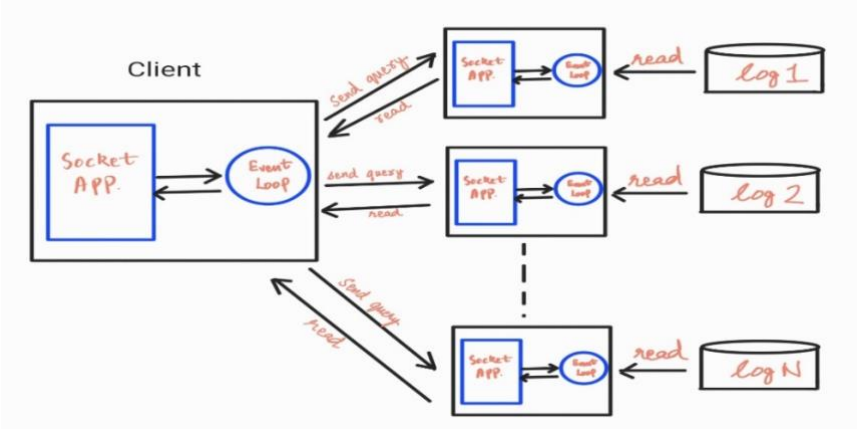
CS 425 – MP1 Report

Group 69 – mgoel7 and bachina3

Architecture and Algorithm

We have gone with a simple distributed architecture where the client sends the search queries to the individual servers. The servers then run the grep commands locally and sends back the outputs to the clients. Since querying and fetching the results are I/O operations, we have utilized the default “asyncio” library in Python. Using “asyncio” we have employed a single-threaded, single-process design which achieves concurrency using an event loop. On the server side, for each connected client, we add a task in the event loop. As soon as a query is received from a connected client, we execute its task from the event loop and return the outputs back to client. On the client end, for each server connection, we create a task in the event loop. We then execute all tasks and wait for every task to complete.

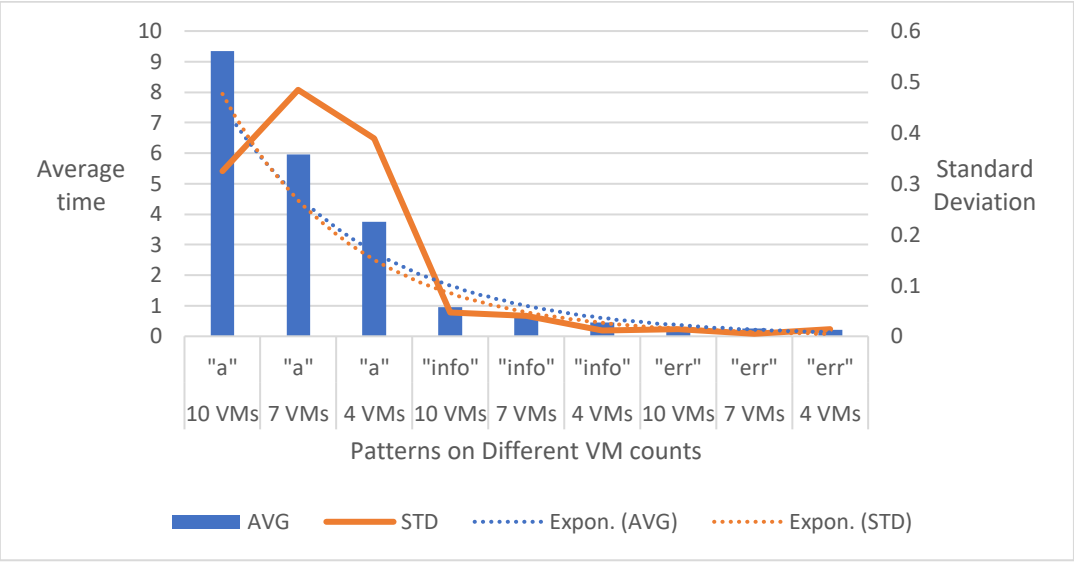


Unit Tests

We have implemented 3 simple unit tests for testing our implemented code. We test our code on 4 VMs, assuming that the VMs are already running the server application. The first test tests match count of frequent patterns present in 3 of the 4 VMs. The second test tests infrequent patterns present in 2 of the 4 VMs. The third test tests output when all servers have crashed where we expect all servers to return 0 line counts.

Performance

We tested our solution by search a frequent pattern (“a”), a somewhat frequent pattern (“info”) and a rare pattern (“err”) on 10, 7 and 4 VMs each. Below is the performance time (in seconds).



Machine count	Pattern	Match count
10 VMs	"a"	2709264
7 VMs	"a"	1899395
4 VMs	"a"	1091212
10 VMs	"info"	270780
7 VMs	"info"	190086
4 VMs	"info"	109274
10 VMs	"err"	40013
7 VMs	"err"	27817
4 VMs	"err"	15893

The results were as expected. We observed that the frequent pattern took the most time and for each pattern, the average time taken scales linearly with the number of connected VMs. We also observed that the standard deviation for all test cases is less than 1 which implies that the running times are consistent. This is also highlighted by the fact that the trendlines for St. Deviation and Average are almost identical. One interesting thing to note is that for “err”, the running time for all VM counts are almost identical which is a slight deviation from the norm. However, this may be attributed to network latency.