CS 425 MP1 : Distributed Grep

Team members: Imani Palmer (ipalmer2), Ramya Nayanaswamy (rpn2)

Programming Language Used: Python

Brief description of logic:

The program implements distributed grep using socket programming and client-server TCP/IP architecture model. The client (client.py) receives the entire grep command as command line argument. This allows expansion of client/server codes to other required Unix commands, if needed, in future. However, for purposes of MP1, only grep has been tested.

Client establishes a socket connection and sends and receives messages through a multiprocessing function "send_query". Server/remote machines (server.py) receive the grep command in its entirety, spawns a child process to execute the received command and returns the result. Server appends pre-coded message to grep results to indicate end of results. Client loops to receive all of the sent message, checking for pre-coded end condition. Thus, client receives results from all remote machine and itself in parallel. The received messages are printed to the user serially. Both client and server codes use Python "logging" framework for debugging/information purposes.

Unit Test:

Various unit tests were developed using "Python Unittest" framework. Tests for various grep options, pattern frequency, results from one, few and all machines are coded and validated with expected results stored in a file for comparison. A sample log was split into 4 chunks of 60MB (we have ~57MB) for testing purposes and average query speed for various patterns were observed, as per specification requirement

Query Speed:

The average query time for a pattern (irrespective of frequent/infrequent) was observed to be 0.025s (clock time) and 0.115s (Wall time). This time involves setup of multiprocessing function, setup of socket, sending command, remote grep time and receiving results. Our conclusion from these results are: the size of grep results is not a dominating factor, as the average time was almost same across various patterns. However, file size is one component contributing to query time.

The grep command was executed in remote machine and only the results were transferred to local/query machine. Transferring the log file to local machine is dependent on various factors like network bandwidth, socket resource availability for lengthy periods of time and security concern of exposing entire log content to network. Additionally, we believe that local machine logic will become both processor intensive and IO intensive if entire log file needs to be fetched and grep were to run locally. Hence, for all patterns we considered executing grep remotely.