Program Design

Our implementation of distributed grep includes a server session which runs on every machines and a client which send query and handling results returned by server session. The client can be run at any of the 10 machines and once it got the request, it will creates 10 threads to talk to 10 server sessions which looks for matches. After received the results from server sessions, the client saves the result to individual files and outputs metrics to screen. Our implementation is based on python with XMLRPC.

Unit Test

In our unit test, we have two major components: to generate log files at every machine with rules and to assert the query result from the distributed grep with various cases. To generate log files at each machine, we generate machine.i.log where i is the machine number containing unique line for the machine, frequent lines appearing in all machines, somewhat frequent lines appearing in all machines, and some lines only if their machine number is even. Once we have the generated log files ready, we call the distributed grep we implemented and compare the returned result with the expected values. We used python's unit test to kick off the unit test flow and to assert the returned result and the expected values.

Performance Evaluation

As required in the document, we conduct 5 performance tests and take the average of the query latency when 4 machines each store 72MB (> 60MB) log files. We tried both frequent pattern ("GET") and infrequent pattern.

Case	1	2	3	4	5	Average
"GET" (freq)	19.036s	19.113s	19.084s	18.769s	18.722s	**18.945s**
"legend" (infreq)	0.108s	0.096s	0.108s	0.104s	0.093s	**0.102s**