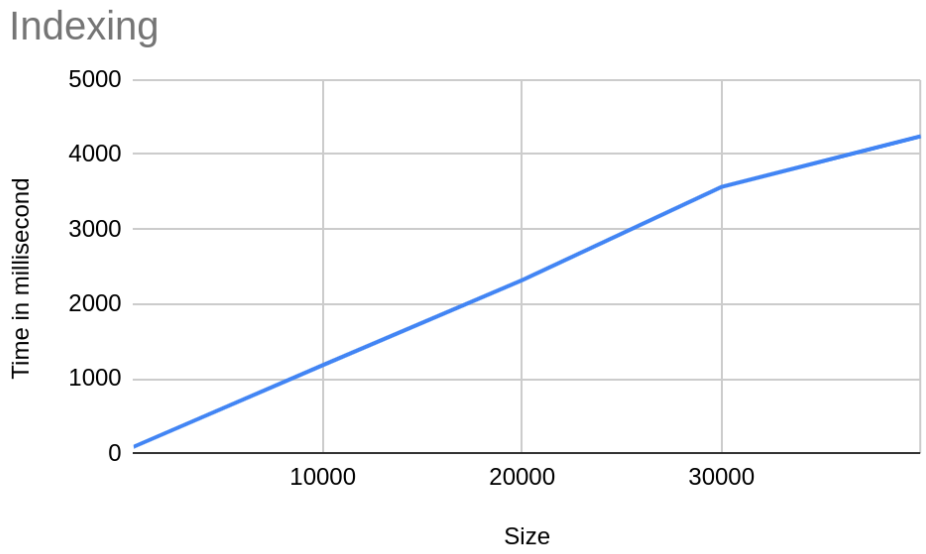
CSC263 Assignment 2

- Vladimir Maksimovski
- Nikola Danevski

All of the information about our testings are saved in this Google Sheet. The Google Sheet is also attached to this submission, as a .xlsx file. If you are logged in with your University of Rochester email, you should be able to access it.

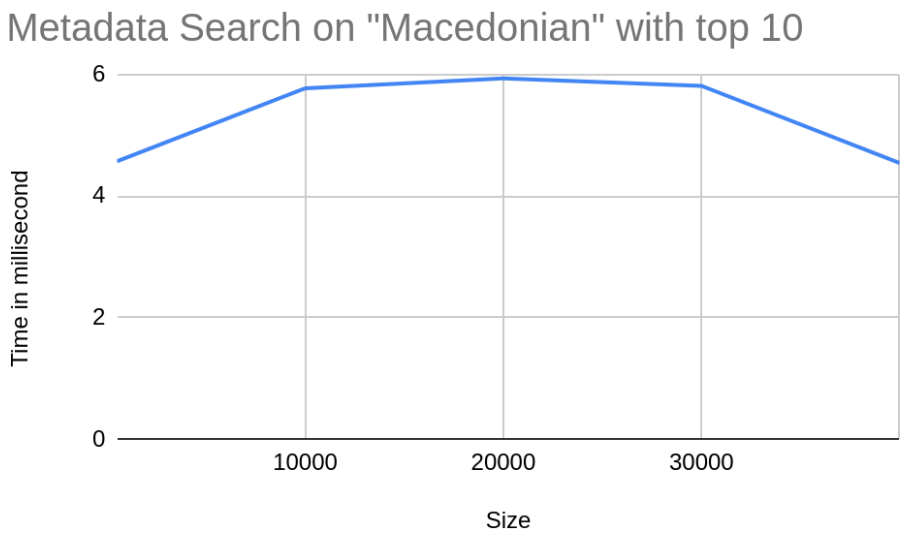
To compile the programs, run “make”. The executables generated will be “metadata_indexer”, “metadata_search”, “keyword_search”.

1. Indexing time versus data size



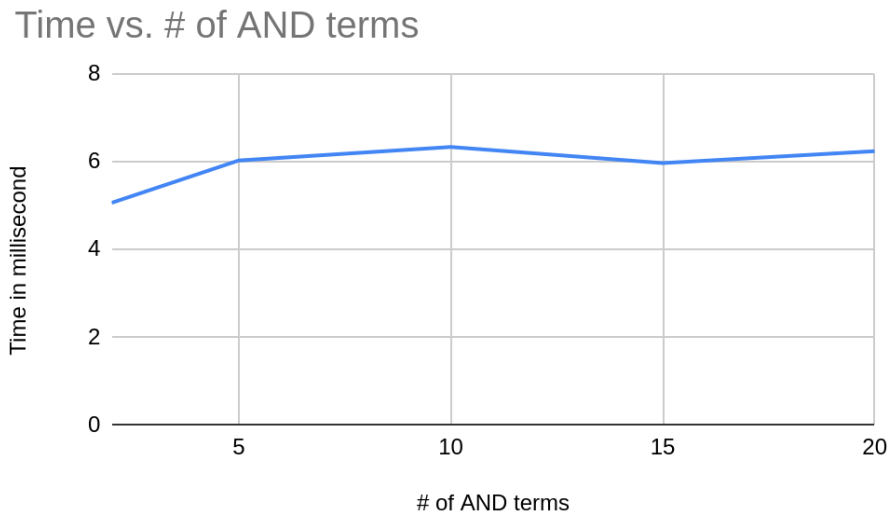
We can see that the indexing time is almost linear with respect to the size of the corpus which is in line with expectations.

2. Search time versus data size



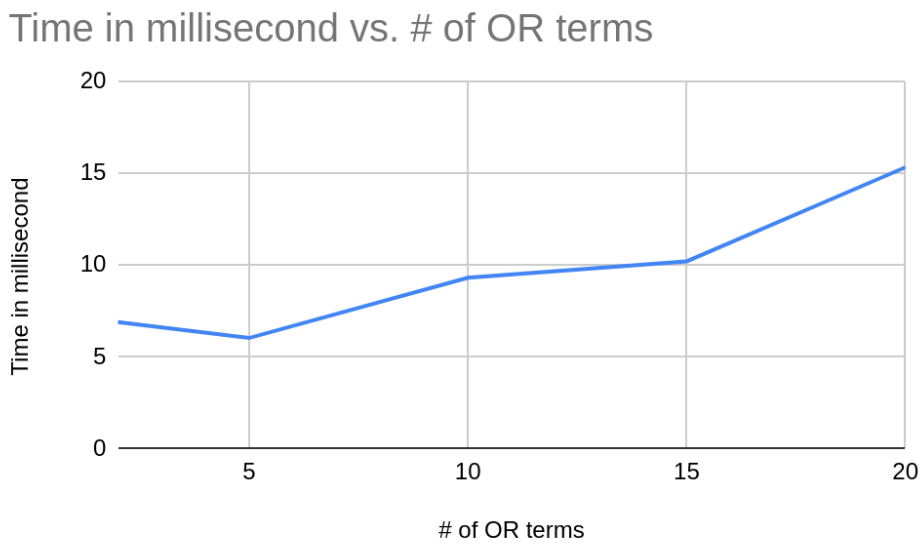
The search time with one word for the query does not increase as the size of the corpus increases. This is a little bit surprising, but the fact that it is only one keyword for the query definitely has a big effect on the resulting time.

3. Search time versus number of AND-keywords in the query



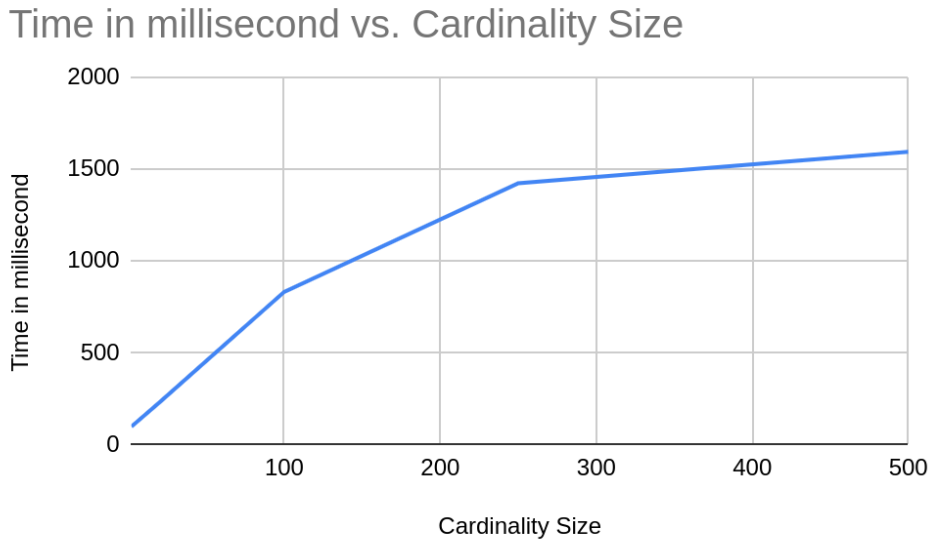
The testing was done such that each of the groups contained the words in the previous group. This result shows that the search time does not vary much and increases just a little bit as the number of AND-keywords in the query increases. This is so because in fact the words were chosen to be of one specific document which contained all of them, and eventually that was the only document that contained ALL of the words, so somewhere around 10 terms, it did not make a difference if we added more terms or not.

4. Search time versus number of OR-keywords in the query



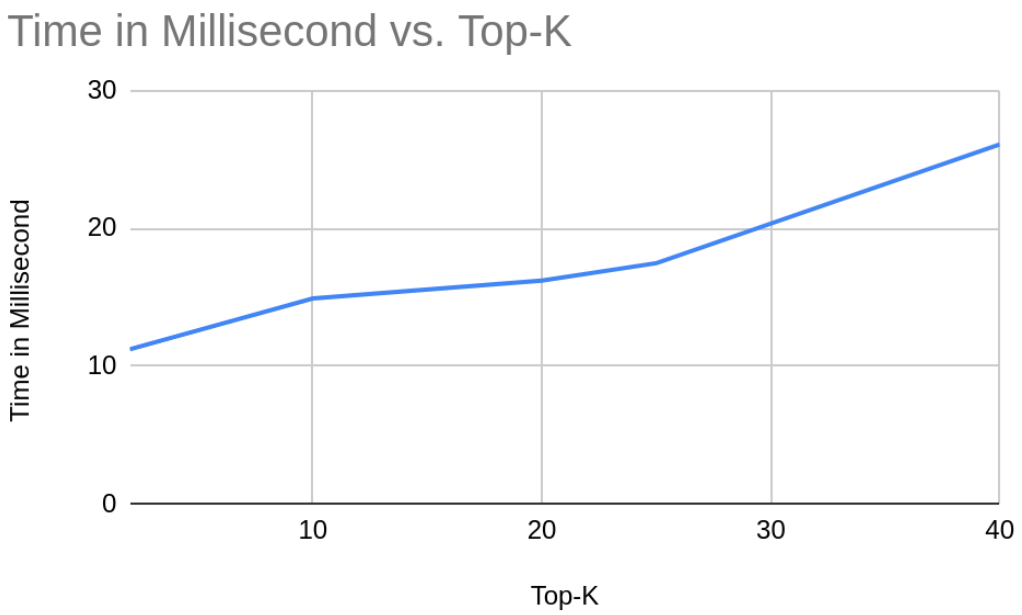
Contrary to the previous case, here we do have a significant increase in time because we consider OR-terms. As we increase the number of OR-terms, more documents will satisfy the filter criteria, so the number of documents that need to be ranked for the search results increases. So performance is quicker when fewer documents need to be considered in the ranking.

5. Plot the top-K search time (y-axis) versus the cardinality of the query set. Consider K = 10.



In this test, we again have a significant time increase by increasing the cardinality size. We always take the first ten as the assignment requires, but since the cardinality of the query set changes, the time taken to compare the set to the document differs. Additionally we do this 30 times (we repeat the process as requested) and this makes a difference as well.

6. [Optional] Plot the top-K search time (y-axis) versus K for a query set. Consider the following K values: 2, 10, 20, 25, 40.



We see that as K increases, the time taken also increases. Our thinking is that the more document descriptions printed to the terminal, the more printf will have to empty buffers, resulting in overhead from system calls.