Phase 3 Report: CMSC 646 Akarsh Kashamshetty

Preprocessor: I have used the same preprocessor in this phase from phase 2. I have not much changed the tokenizer.

Algorithm: To create the dictionary file and the posting file I have used two dictionaries respectively. The dictionary file has the key as the word and value is a list of number of documents the word is present in and its first occurrence in the postings file. As we have already the first value of the list while we are calculating the term weight of the word using the idf, now to create a postings file I used another dictionary where the key is word, and the value is again a list of document id and weight. So, while calculating the weight I used posting file to store each word and its document along with its weight in document. So, when I finish calculating the weights of all the words in files my postings file has all the words with document id and weight. So, I later wrote those values into the txt file. Now my dictionary file needs to be updated with the first occurrence so using postings file in nested loop I calculated the position of each word first occurrence and later wrote all the key and values into the file.

Efficiency:

The following is the output screenshot when executed on all files(503).



Following is the graph of the timings as a function if number of documents indexed.

Chart, line chart

Description automatically generated

In regard to the files size of the input and output (dictionary and postings), the size of the output files is increasing in respects to the input files but after some point the increase in output file size is not significant. Following is the graph of the size efficiencies. On the x axis size

Chart, line chart

Description automatically generated

is plotted in kilo bytes.

**Command** : To run the program : python3 phase3.py files/ output/