Wachirayana Wanprasert sec 3 6088082
Chavanont Sakolpongpairoj sec 3 6088157
Panaya Sirilertworakul sec 3 6088164

ITCS 414: Information Storage and Retrieval

Program Description:

We implemented 3 classes included Index.java, Query.java, and BasicIndex.java.

Index class: This class, it will get termDict and index in the writePosting method. It will create postingList that contains termID, docIDs and docFrequency in each file to create the algorithm of token, merge block and corpus.index. Each creates, it will delete the file in outdir and then create the new postingList.
Query class: this class will create outputQuery and Interset.
BasicIndex class: this class will change postingList to read and write files.

Indexing algorithm
Check the postingList and if TermID and DocID of each file in Posting are greater than 0, TermID and DocID will be added to the postingList. And then it will merge sub-block b1 and b2 together.

Retrieval algorithm
Query class will get the corpus from the indexing algorithm to get the merge result. Then, token it to the single word then compare it with the PostingList in Index class. Using link list to store the docID and then examine the output.

Questions and Answers:

a) We asked you to use each sub-directory as a block and build index for one block at a time. Can you discuss the trade-off of different sizes of blocks? Is there a general strategy when we are working with limited memory but want to minimize indexing time?
Ans:

Index

	Small	Large	Citeseer
Total Files Indexed	6	98998	18824
Memory Used	0.519736 MBs	588.088816 MBs	497.734144 MBs

Wachirayana Wanprasert sec 3 6088082
Chavanont Sakolpongpairoj sec 3 6088157
Panaya Sirilertworakul sec 3 6088164

Time Used	0.108 secs	840.898 secs	643.855 secs
Index Size	2.17437744140625E-4 MBs	55.37303924560547 MBs	64.60990905761719 MBs

<u>Query</u>

	Small	Large	Citeseer
Memory Used	0.0 MBs	-319.255176 MBs	-22.7464 MBs
Time Used	0.017 secs	13.961 secs	1.016 secs

For the large block, it will use more memory but fast to merge. While the small block, it will use less memory but slow to merge. in order to minimize index time, you need to use the size of block to be close to size of memory usage.

b) Is there a part of your indexing program that limits its scalability to larger datasets? Describe all the other parts of the indexing process that you can optimize for indexing time/scalability and retrieval time.

Ans: If the datasets are too large, it will go to the subdirectory. But if it still too large, there will be no enough memory to process.

c) Any more ideas of how to improve indexing or retrieval performance?

Ans: If we can write the appropriate algorithm to organize and sort the words it will be more effective. Also, upgrading the hardware would help.