Faculty of Arts and Sciences
Department of Computer Science
CMPS 365 - Information Retrieval & Web Search
Assignment 2

1 Exercise 1

For your collection, create a text-file version of an inverted index. Remove stopwords, normalize and stem the text. You can use any off-the-shelf stemming libraries out there such as the Porter stemmer or the Snowball one. Your inverted index should contain a line for each unique term in your collection, which contains the term, the document frequency of that term (i.e., how many documents it appears in) and the posting list of that term (i.,e, the ids of documents that contain that term). Keep track of the mapping between document ids and the actual filenames of documents in your collection on your computer so you could retrieve an actual document given a document id.

From your inverted index produce a number of basic statistics: 1) the number of distinct words, 2) the number of documents, and 3) the average number of (distinct) words in a document. What is the size of your inverted index relative to the size of the collection (in bytes)? Also, produce a histogram of the document frequencies of the different terms in your index. Come up with at least one more parameter that you think might be interesting. Report all these statistics and figures in one slide.

2 Exercise 2

1. Using your inverted index, produce 26 lists: the list of all documents containing a word starting with a, the list of all documents containing a word starting with b, and so on until z. Either produce a single file (with lines consisting of a letter, followed by a TAB, followed by document ids), or 26 files (named after the letter and with each line merely holding a doc id), as you prefer.

2. Research intersection algorithms and try to find/come up with one that is more efficient than the linear one outlined in the videos. Describe the algorithm in a second slide.

3. Implement the two algorithms, the one you have seen in the videos and the more efficient one you found or came up with.

4. Use the two algorithms to intersect all 325 pairs of (not-the-same) lists from (1). For each pair check that both algorithms give the same result. Make sure that both algorithms deal with duplicates in the input lists in the same way, or make sure that you do not have duplicates in your input lists. For each algorithm, measure the total running time (summed over all list pairs) in relation to the total number of list elements (sum of the lengths of each list pair summed over all list pairs), and report the two rates (elements per second) in a third slide.

5. Report the rate for reading the 26 lists from disk (again elements per second), as well as the programming language you used and some basic specifications of the machine on which you ran your experiment in that same third slide.

3 Exercise 3

Generate a sample of five Boolean queries for your collection. The queries should combine query terms using the AND, OR and NOT operators. Try to come up with a diverse set of queries in terms of the number of keywords and how you combine them using logical operators. Use your inverted index from exercise 1 to process the queries and retrieve all the documents that satisfy each of your five queries. To be able to do this, you would need to extend one of the intersection algorithms you have implemented in exercise 2 to handle arbitrary Boolean queries involving AND, OR and NOT. Make sure to verify that your algorithm is working correctly by checking that some sample documents retrieved by your algorithm for each query do actually satisfy the query. In a fourth slide, list the five queries you used, describe how you extended the intersection algorithm to handle OR and NOT and report the average runtime of your algorithm over the five queries.