Cmpe 493 Introduction to Information Retrieval, Spring 2018
Assignment 1 - A Simple Document Retrieval System for Boolean Queries, Due: 08/03/2018 (Thursday), 17:00

In this assignment you will implement a document retrieval system for simple boolean queries using the positional inverted indexing scheme. You will use the Reuters-21578 data set, which can be download from Moodle (reuters21578.zip). Reuters-21578 contains 21578 news stories from Reuters newswire classified under one or more of 118 categories. There are 22 SGML files, each containing 1000 news articles, except the last file, which contains 578 articles.

You should perform the following steps:

1. Pre-processing the Data Set : The text of a news story is enclosed under the <TEXT> tag. You should use the <TITLE> and the <BODY> fields to extract the text of a news story. Implement your own tokenizer to get the tokens from the news texts and perform normalization operations including case-folding, stopword removal, and stemming. Please use the stopword list on Moodle (stopwords.txt) for stopword removal and the Porter Stemmer (http://tartarus.org/martin/PorterStemmer/) for obtaining the stem of each token. Note that you should perform the same preprocessing steps for the queries as well. (Assume that the files in the reuters21578.zip file are in the folder named as "Dataset".)

2. Building the Positional Inverted Index: Instead of taking each SGML file as a document unit, you should index each news article as a separate document and use the $NEWID$ field as document IDs. Then, you need to generate two files, namely the dictionary and the inverted index. The first file should contain the stems of the words and their ids as a hashmap. The second file should contain the positional indexes of each stem based on their ids in the first file. When you implement the boolean query processor, you should use **only** these two files, not the original Reuters-21578 dataset. Note that the inverted index construction and query processor should be designed as two separate modules. That is, the query processor should NOT construct the inverted index each it is run. It should just use the dictionary and inverted index files generated by the indexing module.

3. Implementing a Boolean query processor: You should implement the postings merge algorithm for conjunctive, phrase, and proximity queries. Assume that each query consists of either a conjunction, a phrase, or a proximity of terms. You do NOT need to handle disjunctive queries, wildcard queries, or queries containing the NOT operator or parenthesis. That is, the queries will be of the following three types:
   (i) w_1 AND w_2 AND w_3... AND w_n
   (ii) $w_1\ w_2\ w_3$... w_n
   (iii) $w_1\ /k_1\ w_2\ /k_2\ w_3\ /k_3$... $/k_{n-1}\ w_n$
   where each w_i is a single-word keyword and k_{i-1} is a constant number which implies that w_i is within k_{i-1} words of w_{i-1}. That is, a matching document will have w_{i-1} in the text followed by at most k_{i-1} words, followed by w_i. For example, $w_1\ /0\ w_2$ is equivalent to the phrase query $w_1\ w_2$. Note that the order of the query terms is important for the proximity queries: $w_{i-1}\ k_{i-1}\ w_i$ is NOT the same query as $w_i\ k_{i-1}\ w_{i-1}$.

The query processor should take as input a query preceded by its query type information (1 for conjunctive query, 2 for phrase query, and 3 for proximity query). Here are some example input queries:

- 1 oil AND price
- 2 oil price
- 3 oil /3 price

The query processor should return the IDs of the matching documents sorted in ascending order.

You may use any programming language of your choice. However, we should be able to run your program by following the instructions in your readme file and you should not use any third party libraries except for stemming.

Submission: You should submit a *".zip"* file named as YourNameSurname.zip containing the following files using the Moodle system:

1. Report:
   (i) Describe the steps you have performed for data preprocessing and provide answers for the following questions.

   (a) How many tokens does the corpus contain before stopword removal and stemming?
   (b) How many tokens does the corpus contain after stopword removal and stemming?
   (c) How many terms (unique tokens) are there before stopword removal, stemming, and case-folding?
   (d) How many terms (unique tokens) are there after stopword removal, stemming, and case-folding?
   (e) List the top 20 most frequent terms before stopword removal, stemming, and case-folding?
   (f) List the top 20 most frequent terms after stopword removal, stemming, and case-folding?

   (ii) Describe the data structures that you used for representing the positional index (dictionary and postings).

   (iii) Provide a screenshot of running your system for a conjunctive query.

   (iv) Provide a screenshot of running your system for a phrase query.

   (iv) Provide a screenshot of running your system for a proximity query.

2. Source code and executable: Commented source code and executables of your document retrieval system.

3. Readme: Detailed readme describing how to run your program.

Contact: For questions/comments you can contact Abdullatif Köksal (abdullatifkoksal@gmail.com).
Late Submission: You are allowed a total of 5 late days on homeworks with no late penalties applied. You can use these 5 days as you wish. For example, you can submit the first homework 2 days late, then the second homework 3 days late. In that case you will have to submit the remaining homeworks on time. After using these 5 extra days, 10 points will be deducted for each late day.