CPS 842 Assignment 1 Report

Daniyil Kotov, 500877422

Algorithms

There are two main algorithms used in my program. One is parsing the CACM collection and along the way populating the Dictionary and Postings data structures. Second one is for retrieving relevant to a user's term query documents.

The parsing takes place line by line. Each line is checked for it being a field identifier (.I, .T., .W, etc). If so, the algorithm identifies which field it just hit and starts parsing next lines one by one until it hits the next field identifier. When hitting the next field identifier, the algorithm sends the parsed data to be split into a list and each element stemmed (if enabled). Then the algorithm iterates over each element in the list, checking if it is a stopword (if enabled), after what using some basic logic (checking if term already in Dictionary or the doc id for this term has already been appended to the Postings list, etc) either creates a new entry in Dictionary/Postings or increases its TF/DF, appends position).

The retrieval is a simple algorithm that essentially references entries in Dictionary and Postings and prints out their values. The summary search algorithm takes the terms position index in the unparsed, unedited .W section of the CACM collection (that has been stored during the CACM parsing stage). Using that index, it retrieves the search term. The algorithm then finds the same term's position, but in the processed .W section (with stemming/stopwords-removal if enabled). Then using some basic logic it iterates over the left and right sides of the term in this list, finds if there are 5 terms on each side to grab for the summary. If one side is lacking terms, the algorithm tries to grab more on the other side. If that is unsuccessful as well, it just takes as many as possible.

Data Structures

For Dictionary and Postings, I use the Python Dictionary, which is basically a built-in hash map. Dictionaries offer very fast retrieval and insertion, which allows for good and stable performance, considering the amount of terms and entries added for each of the terms. More specifically, many checks like:

```
if term in dictionary:
    ...
```

will result in O(1) complexity complexity.

Example runs

```
[ ~/m/R/4/C/A/src python3 invert.py
Do you want to enable the stopwords removal? (y/n) y
Stopwords removal enabled
Do you want to enable stemming? (y/n) y
Stemming enabled
 ~/m/R/4/C/A/src
```

Invert

```
Please input a term to search for: manifest
----------------------
Extracted term: manifest
Document Frequency: 3
*
Doc ID: 92
Doc Title: A Checklist of Intelligence for Programming Systems
Term Frequency: 1
Term Position(s): [22]
Summary: ...sophistication of various programming systems.  A particular manifestation is the jungle of assorted devices for reproducing...
*
Doc ID: 1751
Doc Title: The Working Set Model for Program Behavior
Term Frequency: 1
Term Position(s): [83]
Summary: ...memories. "Process" and "working set" are shown to be manifestations of the same ongoing computational activity; then...
*
Doc ID: 3170
Doc Title: On the Proof of Correctness of a Calendar Program
Term Frequency: 1
Term Position(s): [41]
Summary: ...derivation and proof of correctness of the program are sketched. The specification is easy to understand, and its correctness is manifest to humans....
----------------------
Search time was: 0.009131193161010742 seconds
----------------------
Please input a term to search for:
```

Look for term ‘manifest’

```
----------------------
Please input a term to search for: malfunction
----------------------
Extracted term: malfunct
Document Frequency: 4
*
Doc ID: 695
Doc Title: Use of the Disk File on Stretch
Term Frequency: 1
Term Position(s): [139]
Summary: ...disk are considered for both recovery from computer malfunction and for mathematical or physical developments during the calculation....
*
Doc ID: 1712
Doc Title: Recovery of Disk Contents After System Failure
Term Frequency: 1
Term Position(s): [16]
Summary: ...method is discussed by which, after a system malfunction, the contents of disk files can be restored...
*
Doc ID: 2558
Doc Title: Protection in Programming Languages
Term Frequency: 1
Term Position(s): [16]
Summary: ...used to protect one subprogram from another's malfunctioning are described. Function-producing functions and various type-tagging...
*
Doc ID: 2868
Doc Title: Reflections on an Operating System Design
Term Frequency: 1
Term Position(s): [75]
Summary: ...provided by earlier ones and protecting itself from the malfunctions of later ones. There were serious...
----------------------
Search time was: 0.010041236877441406 seconds
----------------------
Please input a term to search for:
```

Look for term ‘malfunction’

```
Please input a term to search for: ZZEND
Average search time was: 0.009440064430236816
 ~/m/R/4/C/A/src
```

ZZEND

How to Run

The program was written and tested through Python 3.8. To run the program, check your python version. After that also install *nltk* pip package. The nltk package is used to implement the Porter Stemmer algorithm. Having installed that, just run python3 invert.py or python3 test.py run either of the programs through the terminal.