ENGR 102 – Programming Practice

Practice Session (Week 12)

In this practice session, you will develop a simple command-line-based search engine for a digital library of academic papers. Your search engine will allow performing keyword searches, and search results will be ranked in a particular way.

Dataset:

As part of the practice session materials, you are provided a paper dataset from physics domain. This dataset contains metadata (i.e., title, authors, abstract, year, etc.) for each paper in a separate file (one per paper) which is named as *paperid.abs*, and located under *metadata* directory. An example metadata file content is shown below:

```
\\
Paper: hep-th/9201020
From: STELLE@PHYS.TAMU.EDU
Date: Sat, 11 Jan 1992 17:06:09 CST   (30kb)

Title: The Superparticle and the Lorentz Group
Authors: A.S. Galperin and K.S. Stelle
Comments: 33 pages
Journal-ref: Nucl.Phys. B368 (1992) 248-280
\\
  We present a unified group-theoretical framework for superparticle theories.
This explains the origin of the ``twistor-like'' variables that have been used
in trading the superparticle's $\kappa$-symmetry for worldline supersymmetry.
We show that these twistor-like variables naturally parametrise the coset space
${\cal G}/{\cal H}$, where $\cal G$ is the Lorentz group $SO^\uparrow(1,d-1)$
and $\cal H$ is its maximal subgroup. This space is a compact manifold, the
sphere $S^{d-2}$. Our group-theoretical construction gives the proper
covariantisation of a fixed light-cone frame and clarifies the relation between
target-space and worldline supersymmetries.
\\
```

Task Description:

- ✓ Your search engine first needs to be initialized by parsing and loading the paper data from the above described files.
- ✓ In order to minimize the searching time, you should store word locations in a Python dictionary, called wordlocation where key and value will be as follows.
  - key: a word w, value: a dictionary where key is a paper id, and value is the list of locations that w appear in that paper. That is, your dictionary will be organized as follows:

    wordlocation[*a_word*] = {paperId1: [loc1, loc2, …, locN], paperId2: [loc1, loc2, …, locM], …, paperIdZ: [loc1, loc2, …, locK]}

    Locations are indexes of the words. You may assume that the title and the abstract of a paper are combined into one piece of text. Hence, you do not need to differentiate between title and abstract. The first word in the title will have location 0, and the last word in the abstract will have location (len(title) + len(abstract) - 1).
  - You may use *separatewords* method in mysearchengine.py module to divide paper title and abstract into the individual words.
  - You may use os.listdir() method to get the list of paper files under metadata directory during the initialization phase. The following tutorial has an example:
    - http://www.tutorialspoint.com/python/os_listdir.htm
- ✓ Your program will not have a GUI. Instead, it will have a command-line-based interface as shown in Figure 1. The above described dictionary should be built at the beginning when your program is started. While the dictionary is being built, your program should print "Building the index…" on the screen. Once the index is built, your program will ask for search terms to the user. The user enters a number of search terms, and hits enter, then your program lists matching papers, and their

scores ordered by their scores which will be computed as described next. Then, your program should ask for another keyword query. This should continue until the user enters 'q' as search term.

- ✓ You are going to rank search results by using a frequency-based scoring measure. More specifically, let's say that a user has searched with three keywords, "word1 word2 word3". Assume that a paper p contains 3 occurrences of word1, 2 occurrences of word2, and 5 occurrences of word3. Then, the content-based score of p is 3 x 2 x 5 = 30. You need to call the normalization function from mysearchengine.py to normalize the scores at the end (higher is better).

```
Building the index...

Please enter your search terms: atom

No matching papers found!

Please enter your search terms: model

1. 1.000000   A Simple Model Inducing QCD

2. 0.363636   Fermionic Determinant of the Massive Schwinger Model

3. 0.272727   Exact Finite Size Results on the Ising Model in 2D Curved Space

4. 0.272727   Combinatorial Solution of the Two-Matrix Model

5. 0.272727   Effective action of gauged WZW model and exact string solutions

6. 0.181818   Two dimensional black-hole as a topological coset model of c=1 string  theory

7. 0.181818   The Spatial Dynamics in Kazakov--Migdal Model

8. 0.181818   On the Mass of Two Dimensional Quantum Black Hole

9. 0.090909   Topological 2-Dimensional Quantum Mechanics

10. 0.090909  2D Black Holes and 2D Gravity

Please enter your search terms: q
```

Figure 1: A sample run of your program