CSE 5337/7337: Information Retrieval and Web Search
Project 1: Web crawler (100 points) – Individual effort
Due: March 20, 2018, 11:59 pm

Deliverables:

1. Complete code in a compressed archive (zip, tgz, etc)
2. A readme file with complete description of used software, installation, compilation and execution instructions.
3. A document with the results for the questions below.

Task:

Develop a specialized Web crawler.

You are encouraged to find and reuse an existing crawler and/or libraries. If so, provide references.

Test your crawler only on the data in:
http://lyle.smu.edu/~fmoore
Make sure that your crawler is not allowed to get out of this "*website*". Yes, there is a robots.txt file that must be used. Note that it is in a non-standard location which standard library routines won't handle, so this may present a challenge for you. The contents of the website will change.

The required input to your program is N, the limit on the number of pages to retrieve and a list of stop words (of your choosing) to exclude.

You can assume that there are no errors in the input files. Your code should be robust under errors in the Web pages you're searching. If an error is encountered, feel free, if necessary, just to skip the page where it is encountered.

Efficiency: Don't be ridiculously inefficient. There's no need to deliver turbo-charged algorithms or implementations. You don't need to worry about memory constraints; if your program runs out of space and dies on encountering a large file, that's OK. You do not have to use multiple threads; a single processor application is OK.

1. Implement your crawler according to requirements. Describe your major data structures. Identify the key properties of a web crawler. Describe in detail how each of these properties is implemented in your code. [25 points]
2. Use your crawler to list the URL of all pages in the test data and report all out-going links of the test data (i.e. items you must not crawl). [10 points]
CSE7337: display the contents of the <TITLE> tag

3. Implement exact duplicate detection, and report if any URLs refer to already seen content. [10 points]
4. Use your crawler to list all broken links within the test data. [10 points]
5. List the URLs of graphic (gif, jpg, jpeg, png) files are included in the test data. [10 points]
6. Your crawler must save the words from each page of type (.txt, .htm, .html, .php). Make sure that you do not save HTML markup. A word is a string of non-space characters, beginning with an alphabetic character. It may contain special characters, but the last character of a word is either alphabetic or numeric. Perform case insensitive matching. In this process, give each page a unique document ID. The output of this step will be a term-document frequency matrix. Your program may generate the data to be further processed in a spreadsheet (Excel or equivalent). [25 points]
   CSE7337: implement stemming
7. Report the 20 most common words with its document frequency. [10 points]
   CSE7337: words or stemmed words?

(END)