CSE 5337/7337: Information Retrieval and Web Search

Spring 2017

Project 1: Web crawler (100 points)

Due: March 22 8 am

1. Identify the key properties of a web crawler. Describe in detail how each of these properties is implemented in your code. [20 points]

The key properties of a web crawler are:

* Scalability: This property was not given that much thought in this assignment, given that it was designed to run in a single thread. However, it could be implemented by making the vocabulary dictionary (called vocabulary\_dict in the code) and the URL frontier (called my\_url\_dict in the code) shared across the multiple threads, and starting each thread with a different URL.
* Politeness: The crawler reads the robot.txt file and respects the directories that should be left un-crawled. This is implemented by using the robotparser library, opening the robots.txt file at the known location, and comparing each new URL with the contents of said file with the built-in function “can\_fetch”. Also, if it finds a URL that refers to a document outside the professor's directory, it is not crawled. This is implemented by comparing the URL with a regular expression that describes only documents in that directory. This crawler does not have a delay between requests implemented (given that it is out of scope), but it could be implemented by adding said delay on the urllib.urlopen function. Given that all URLs refer to the same host, that delay can be global across all URLs.
* Robustness: This crawler is able to identify broken links by comparing its title with this webserver's configured broken link title, 404 Not Found. It might not work for different web servers or different configuration, but it does work with the test data. Robustness is also implemented with duplicate detection, that is implementing by calculating the hash of every visited document. Every time a new crawleable URL is found, before crawling it, its hash is calculated and compared to all existing hashes of crawled documents, and not crawling the document if the hash is the same as any of them.
* Freshness: This property of a web crawler is not implemented on the code, as it is considered out of scope. However, it could be easily implemented by storing the dictionary and the URL list data structure (that contains hashes for each document) in non-volatile storage. That data structure would then be read at the start of execution and then the crawling begins. Once a URL has been found, its hash is calculated and compared to the previous hash for that same document. If they are not identical, it means the document has changed. All entries in the dictionary with that document ID would then be deleted so the document can be crawled again.

1. Use your crawler to list the URL of all pages in the test data and report all out-going links of the test data. [10 points]  
   **CSE7337**: display the contents of the <TITLE> tag

All URLs:

URL: https://lyle.smu.edu/~fmoore/misc/porter\_stemmer\_example.html, Title: Porter Stemmer Online, Type: CrawleableDocument

URL: https://lyle.smu.edu/~fmoore/SMU-CSE-LOGO.gif, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/misc/urlexample1.htm, Title: URL Example - relative vs absolute, Type: CrawleableDocument

URL: https://9ol.es/porter\_js\_demo.html, Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore/CSE7337\_syllabus.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/example-2017-03-02.xlsx, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/dontgohere/badfile1.html, Title: NONE, Type: ForbiddenRobots

URL: https://lucene.apache.org/core/, Title: NONE, Type: OutgoingLink

URL: https://en.wikipedia.org/wiki/Stop\_words, Title: NONE, Type: OutgoingLink

URL: https://smu.instructure.com, Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore/misc/tokenizing\_exercise.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/page113.xlsx, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/index.htm, Title: Freeman Moore - SMU Spring 2017, Type: Duplicate

URL: https://lyle.smu.edu/~fmoore/misc/algorithm-1-7.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://tartarus.org/~martin/PorterStemmer/, Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/count\_letters\_duplicate.txt, Title: NOT FOUND!, Type: CrawleableDocument

URL: https://lyle.smu.edu/~fmoore/misc/exam1.html", Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/levenshtein.html" target="\_blank , Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/count\_letters.txt, Title: NOT FOUND!, Type: Duplicate

URL: https://lyle.smu.edu/~fmoore/misc/excel-icon.jpg, Title: NONE, Type: ImageLink

URL: https://en.wikipedia.org/wiki/Tf\*idf, Title: NONE, Type: OutgoingLink

URL: https://en.wikipedia.org/wiki/Document\_classification, Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore/misc/excel-examples.html, Title: CSE 5/7337 excel demo for March 21, Type: CrawleableDocument

URL: https://lyle.smu.edu/~fmoore/misc/mercator-article.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/mailto:fmoore@lyle.smu.edu, Title: NONE, Type: OtherDocumentFormat

URL: https://search.carrot2.org/stable/search, Title: NONE, Type: OutgoingLink

URL: https://www.smu.edu/EnrollmentServices/Registrar/Enrollment/FinalExamSchedule/Spring2017, Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore/this\_aint\_gonna\_work.htm, Title: NONE, Type: BrokenLink

URL: https://lyle.smu.edu/~fmoore/misc/jaccard\_example.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/misc/google-and-or-example.jpg, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/CSE5337\_syllabus.pdf, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/, Title: Freeman Moore - SMU Spring 2017, Type: OtherDocumentFormat

URL: https://www.gedpage.com/soundex.html" target="\_blank , Title: NONE, Type: OutgoingLink

URL: https://lyle.smu.edu/~fmoore/index\_duplicate.htm, Title: Freeman Moore - SMU Spring 2017, Type: Duplicate

URL: https://lyle.smu.edu/~fmoore/misc/page113-ex2.xlsx, Title: NONE, Type: OtherDocumentFormat

URL: https://lyle.smu.edu/~fmoore/schedule.htm, Title: SMU CSE 5337/7337 Spring 2017 Schedule, Type: CrawleableDocument

URL: https://lyle.smu.edu/~fmoore/does\_not\_exist.htm, Title: NONE, Type: BrokenLink

URL: https://lyle.smu.edu/~fmoore/misc/permutermindex-example.jpg, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/misc/word-morphing-puzzle.pdf, Title: NONE, Type: OtherDocumentFormat

Outgoing links:

URL: https://9ol.es/porter\_js\_demo.html, Title: NONE, Type: OutgoingLink

URL: https://lucene.apache.org/core/, Title: NONE, Type: OutgoingLink

URL: https://www.gedpage.com/soundex.html, Title: NONE, Type: OutgoingLink

URL: https://www.smu.edu/EnrollmentServices/Registrar/Enrollment/FinalExamSchedule/Spring2017, Title: NONE, Type: OutgoingLink

URL: https://search.carrot2.org/stable/search, Title: NONE, Type: OutgoingLink

URL: https://en.wikipedia.org/wiki/Stop\_words, Title: NONE, Type: OutgoingLink

URL: https://smu.instructure.com, Title: NONE, Type: OutgoingLink

URL: https://en.wikipedia.org/wiki/Tf\*idf, Title: NONE, Type: OutgoingLink

URL: https://en.wikipedia.org/wiki/Document\_classification, Title: NONE, Type: OutgoingLink

1. Implement duplicate detection, and report if any URLs refer to already seen content. [10 points]

For this question, <https://lyle.smu.edu/~fmoore/> and <https://lyle.smu.edu/~fmoore/index.htm> are not considered as the same URL. The fact that both of these URLs direct the user to the same document depends only on the web server configuration. If the crawler is designed to work with any server, the assumption that they always direct to the same document cannot be made.

URLs that refer to already seen content:

URL: https://lyle.smu.edu/~fmoore/index.htm, Title: Freeman Moore - SMU Spring 2017, Type: Duplicate

URL: https://lyle.smu.edu/~fmoore/misc/count\_letters\_duplicate.txt, Title: NOT FOUND!, Type: Duplicate

URL: https://lyle.smu.edu/~fmoore/index\_duplicate.htm, Title: Freeman Moore - SMU Spring 2017, Type: Duplicate

1. Use your crawler to list all broken links within the test data. [10 points]

URL: https://lyle.smu.edu/~fmoore/this\_aint\_gonna\_work.htm, Title: NONE, Type: BrokenLink

URL: https://lyle.smu.edu/~fmoore/does\_not\_exist.htm, Title: NONE, Type: BrokenLink

1. How many graphic files are included in the test data? [10 points]

There are 4 different graphic files:

URL: https://lyle.smu.edu/~fmoore/SMU-CSE-LOGO.gif, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/misc/google-and-or-example.jpg, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/misc/permutermindex-example.jpg, Title: NONE, Type: ImageLink

URL: https://lyle.smu.edu/~fmoore/misc/excel-icon.jpg, Title: NONE, Type: ImageLink

There are 6 total occurences of these files.

1. Have your crawler save the words from each page of type (.txt, .htm, .html). Make sure that you do not save HTML markup. Explain your definition of “word”. In this process, give each page a unique document ID. [25 points]  
   **CSE7337**: implement stemming

My definition of a word consists of a set of characters delimited by a space or a slash. That word is then stemmed using an implementation of Porter's algorithm, while removing punctuation at the end of it (the kind of punctuation removed is the one specified by python's string.punctuation list, !"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~). Examples of words under this definition include “1”, “5:50”, “cse”, “hello”, “chpt”, etc.

1. Report the 20 most common words with its document frequency. [15 points]  
   **CSE7337**: words or stemmed words?

chpt (14 times)

2 (11 times)

hmwk (10 times)

2017 (9 times)

freeman (9 times)

moor (9 times)

cse (9 times)

mar (9 times)

mai (9 times)

exampl (9 times)

apr (8 times)

feb (8 times)

site (8 times)

1 (8 times)

4 (7 times)

spring (7 times)

due (7 times)

includ (7 times)

cours (7 times)

assign (7 times)

These are the stemmed words. I am stemming every word before checking whether it already exists in my dictionary or not. This way, “assigning” and “assignment”, for example, are grouped, and are considered as a single term when storing the document they were found on. This is the whole point of stemming, so that when a user searches for one of these words, a document containing the other can also be matched.

(END)