CS8803 BDS / CS4365

Homework Assignment 1

(Programming Category)

Student Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Session: cs8803 or CS4365 (circle one)

You are given three types of programming problems in the first homework assignment. You only need to choose one of them as your first homework. Feel free to choose any of your favorite programming language Java, C, Perl, Python, …

Post Date: Before or on Monday of Week 3 (Aug. 31)

Due Date: midnight on Friday of Week 4 (Sept 11)

You are asked install and run an open source search engine software, such as Apache Lucene search engine (<http://lucene.apache.org/core/>), and report your experience. This problem has a number of options. You are expected to perform one of the following options:

**Problem 1. Hand-on Experience with a Web Crawler.**

1. You are asked to read the article on Web Crawler at <https://en.wikipedia.org/wiki/Web_crawler>
2. You have two options:

**Option 1: Experience with an Open Source Crawlers**

1. Download and install a Web Crawler, such as an open source crawler from <https://en.wikipedia.org/wiki/Web_crawler> or PeerCrawl (<http://www.cc.gatech.edu/projects/disl/PeerCrawl/>).
2. Select a seed URL to initialize your Web crawler and you are expected to crawl at least 1000 URLs.
3. Show the design of your Web archive to store your crawled web pages, the keywords (subjects) you have extracted.
4. Plot the crawl speed in terms of the number of keywords you have extracted and the number of URLs you have extracted and the number of URLs you are able to crawl.
5. Discuss your experience and lessons learned. Predict how long your crawler may need to work in order to crawl 10 millions of pages and 1 billion of pages.

Deliverable:

1. Source code and executable with readme.
2. ScreenShots of your Web Crawler’s Command lines or GUIs
3. Crawl Statistics (Crawl speed🡪 #pages/minute, ratio of #URL crawled / #URL to be crawled, etc.) in excel plots or tabular format
4. Discuss your experience and lessons learned.

**Option 2: Write a Web Crawler of your own.**

Feel free to use any open source crawler as the code base. Write your focused crawler, such as crawling only CoC website or crawling only healthcare web pages. You are expected to crawl at least 1000 pages.

Deliverable:

1. Source code and executable with readme.
2. Discuss the design of your crawler: Pros and cons.
3. ScreenShots of your Web Crawler’s Command lines or GUIs
4. Crawl Statistics (Crawl speed🡪 #pages/minute, ratio of #URL crawled / #URL to be crawled, etc.) in excel plots or tabular format
5. Discuss your experience and lessons learned.

**Problem 2. Search Engine Optimization**

1. Read the article on Search Engine Optmtimization: <https://en.wikipedia.org/wiki/Search_engine_optimization>
2. Determine an optimization that you propose to implement, which can effectively improve the performance of a Search Engine.

For example, improving the quality of PageRank algorithm, such as replacing uniform rank distribution with personalized rank distribution, improving the efficiency of PageRank algorithm, such as convergence rate, computation efficiency.

Deliverable:

1. Source code and executable with readme.
2. ScreenShots of your SE Command lines or GUIs
3. Measurements of effectiveness of your optimization methods.

**Problem 3:**  **Hand-on Experience with an Open Source Peer to Peer System**

Download an open source peer to peer software system. You are expected to set up a peer to peer network system with a minimum of 50 peer nodes and each node has a skewed degree distribution. Create a P2P application such as file sharing, and measure the performance of this peer to peer network system in terms of transaction/message throughput and latency.

Here are some example P2P systems:

1. Gnutella: http://sourceforge.net/projects/gtk-gnutella/
2. PeerSim: A P2P Simulator: <http://peersim.sourceforge.net>
3. OpenVoIP: <http://www1.cs.columbia.edu/~salman/peer/>
4. Morpheus: <http://cwenger.github.io/Morpheus/>
5. Chord: <http://www.vromans.org/johan/projects/Chordii/>

Deliverable:

1. ScreenShots of your P2P Command lines or GUIs
2. Measurements of your P2P system in terms of throughput and latency.
3. Discuss scalability, reliability and anonymity of your P2P system.