

COEN272 Web Search and Information Retrieval

Project 1 (20%)

Web Crawling
Content Processing
Zipf's Law

Part 1: Create your own Web Crawler using Java

1. Take as input a comma-separated specification file called **specification.csv**, containing (in this specific order):
 - A single seed URL, e.g. `http://www.apple.com/mac/`
 - Maximum number of pages to crawl in total
 - Domain restriction, e.g. only within `www.apple.com`. This should be allowed to be left blank.

E.g. `http://www.apple.com/mac/ , 1000 , www.apple.com/`
2. Crawl the Web, starting with the seed URL
3. For each crawled URL
 - Download the complete textual page content (including all html tags, but not images) into a folder called **repository**
 - Add the page URL, title, and additional statistics to a file called **report.html** (see 4 below for details)
 - Crawl all outlinks
4. At the end of the crawl, your program should have:
 - Downloaded the content from all crawled pages into the **repository** folder
 - Generated **one report.html** file (layout is up to you), which shows the following for the crawled pages:
 - A clickable title leading to the live URL
 - A link to the downloaded page in the repository folder
 - Page statistics: HTTP status code, number of outlinks, number of images

Web Crawler (cont.)

- Must include politeness policies (e.g. respect `robots.txt` files)
- **Bonus:** Make the crawler multi-threaded

Part 2: Create a Content Processor in Java

- Process the downloaded content in the **repository** folder to **remove noise** as best as possible, ideally only leaving the main content text
- Minimum requirements:
 - Remove main navigation bars
 - Remove ads
- Removal technique should be site-agnostic, i.e. it should not be a technique that only works on one particular website

Note: You can use any technique of your choice, but it must be implemented in Java.

Part 3: Analyze content

- Perform word frequency analyses on the processed content for 3 different crawls (e.g. with different seeds, restrictions, etc.). For each analysis:
 1. Calculate word frequencies and ranks using all textual content (feel free to use any tool/language)
 2. Plot word frequencies and word ranks (feel free to use any tool/language)
 3. Check if crawled content follows Zipf's law (i.e. compare with typical Zipf's distribution)

Submission

1. Code - All code required to run
 - a) the crawler
 - b) the content processor

2. Report (in PDF - 5 page max) outlining:
 - a) The main components of the programs
 - b) Any design/development/architectural choices. Must include at a minimum:
 - i. Discussion on how noise reduction was performed
 - ii. Discussion on how noise reduction performance was evaluated
 - iii. Discussion on how well the noise reduction technique worked based on the chosen evaluation
 - iv. Any challenges faced during the development of the crawler and content processor
 - c) Word frequency/rank plots for 3 different crawls
 - d) Discussions whether the 3 word distributions follow Zipf's law or not
 - e) An appendix containing details on these 3 crawls (e.g. seeds, domains), as well as a list of the 100 most frequent words for each crawl

Note: Screenshots can be added to an appendix, and do not count towards page limit

Submission deadline

- Apr 27, 2016, 5.10pm