CSE305 Project: Parallel crawling

Gleb Pogudin, gleb.pogudin@polytechnique.edu

Crawling refers to the process of going to a webpage, following all the links from it, following all the links in the found pages, etc. This process is used, for example, by search engines to index the content of the Internet. Or it can be applied to a particular multi-page website to get its content. The goal of this project will be to crawl a given website (e.g., Wikipedia) to create its "index", a list of pages. Furthermore, it is interesting to store some additional information (like ranking, i.e., how far is the given url from the starting one; or the number of links going to this url). To avoid immediate blow-up of the result, the code should allow filtering urls (e.g., browse only wikipedia pages, not external ones).

The resulting code will consist of two major parts:

1. Multithreaded crawler, where each thread takes the next webpage to visit, downloads it, and extracts all the links from it. You may want to use LIBCURL[1] for this.
2. A data structure storing already found webpages and allowing the multiple crawling threads to add new elements. For a team of one, the SetList would be sufficient. For a team of two or more, a concurrent hash-table should be implemented (see, for example, Chapter 13 in the Herlify book[2]; StripedHashSet or something more invoved it required).

Furthermore, it is expected that you:

- Analyze the behavior of your implementation depending on the number of threads involved and propose a heuristic to choose such a number. Note that unlikely the algorithms in the course, the main resource in this case may be not CPU/memory but the internet bandwidth.
- Run the code on Wikipedia or comparable website(s) to generate its index, and analyze which parts of the code takes the largest portions of time.

[1] https://curl.se/libcurl/
[2] https://dl.acm.org/doi/book/10.5555/2385452