Text Technologies – Assignment 1 – Report

s0837795

I. Overview

The enclosed file tts1.py is an implementation of a simple web crawler that explores the inf.ed.ac.uk domain from a seed URL in accordance to the domain's robots.txt file and collects statistics about the links it visited or could not visit.

II. How does it work?

The crawler uses a priority heap queue and the tuples of pages and their priority – the page with the largest number as its filename is assumed to have the highest priority. The crawler keeps track of priorities by subtracting each page's priority from the maximum possible priority and placing it on a heap queue and then using heapq.heappop() to pop the item with the smallest number associated with it. As per the specification of the assignment, the priorities are determined from the names of the pages (as opposed to e.g., number of links point to a page, in more realistic examples). This determines the order in which the crawler traverses the graph.

In order to keep track of visited pages, the crawler uses a queue and computes the checksum of each page. The algorithm is very simple – if the crawler hasn't visited a page previously or the page's checksum has changed (e.g. it has been updated), the crawler will visit it.

The crawler will also only visit a page if it is not disallowed by a directive in the domain's robots.txt file. In addition to that, the crawler takes note of server request limiting directives Crawl-delay and Request-rate. As these two directives are not part of the original Robots Exclusion Standard, my interpretation is that requests to the server will be limited by the amount of time specified in Crawl-delay and Request-rate, whichever is greater.

The code is organised in two main classes – Crawler and Scraper, where the Crawler class keeps track of the metrics and the graph traversal process, whilst Scraper parses web pages and looks for anchor tags via the regular expression: `<a\s*\S*\s*href=[\'|"](.*?)[\'|"].*?>`. The Scraper class also handles failure to follow a link – either due to 404 errors or malformed URL errors.

III. Packages used:

a. urllib2 – handles networking, opening urls, etc.
b. time – to see how long the crawler runs and to time delays between server requests
c. re – for regular expressions to extract URLs, and parse the Delay-crawl, Request-rate directives in robots.txt
d. robotparser – to parse robots.txt and see if the crawler is allowed to fetch a link or not
e. hashlib – to compute checksums (sha224) and see if a page has been updated
f. heapq – priority queue