CS104 Project Report : Web Crawler

ATHARVA BENDALE : 22B0901

14 June 2023

Introduction

I am Atharva Bendale, a student of CSE department, IITB. This is the Web Crawler I made for my CS104 project. A web crawler is a computer program that's used to search and automatically index website content and other information over the internet. These programs, or bots, are most commonly used to create entries for a search engine index. My program recurses over any specified link for a specified depth and, stores the links obtained in an output file.

Libraries

The libraries and dependencies used in this project are:

- **beautifulSoup** : Beautiful Soup is a Python package for parsing HTML and XML documents, we use this to filter the links in **src** and **href** tags.
- **requests** : We use it to get or send '*http*' requests to the site and get information.
- **argparse** : Argparse is a library used for processing command line inputs. We use it for handling the command line arguments.
- **numpy and matplotlib** : For better data understanding and visualisation[1]
- **sys** : We use this for displaying dynamic updates regarding the running program.
- **warnings** : We use it for ignoring unnnecesary warnings during HTML parsing.
- **os** : We use this for removing previous, not required output .png files.

Usage

To use this program, head to the terminal and type "*python3 web_crawler.py -u* $< site-name >$ *-t* $< threshold >$ *-o* $< output-file-name >$ *-f* $< Y/n >$ *-p* $< Y/n >$"

- -u: for the URL, If not given then must print an error on the command line.
- -t: Accepts an int value for the threshold of recursiveness, must be greater than 0, will give an error for an invalid threshold
- -o: Name of the output file, if not provided then by default print on the command line.
- -f: (Y/n), if given an argument **Y**, it calculates the sizes for all files encountered, give no input or **n** for not using this feature
- -p: (Y/n), if given an argument **Y**, it dynamically displays the progress in the running program.

Code Structure

Function of base code[2]

Using the base code from *https://www.geeksforgeeks.org/python-program-to-recursively-scrape-all-the-urls-of-the-website/*, we can scrape all urls from the given site and it prints them all without any segregation. Using such an unorganised data is a very back-breaking and tedious job. The displayed figure is taken from [1]

```
http://example.webscraping.com///places/default/user/register?_next=/places/default/index
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register/places/default/user/register
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register/places/default/user/register/places/default/user/re
gister
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register/places/default/user/register/places/default/user/re
gister/places/default/user/register
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register/places/default/user/register/places/default/user/re
gister/places/default/user/register/places/default/user/register
http://example.webscraping.com///places/default/user/register?_next=/places/default/index/places/default/user/regi
ster/places/default/user/register/places/default/user/register/places/default/user/register/places/default/user/re
gister/places/default/user/register/places/default/user/register/places/default/user/register
```

Figure 1: Output of a basic Web crawler

Modified code

I have modified the basic code to mmet the basic requirements listed by the problem statement which are:

- Crawling the provided domain link to extract links from the **src** and **href** tags and further recurse this process on all these newly obtained links.
- The program will print or store all the links in text format according to the input give by the user to the -o tag (prints if no input given). All the links will be stored or printed according to their file types into categories (HTML, CSS, JS, JPG, PNG, OTHERS, also the total number of files found at that particular depth will be displayed.

Working

The program uses BeautifulSoup library to parse HTML files for links corresponding to **src** and **href** tags. It then modifies the relative links and transforms them into complete links, it then recursively crawls through all the links which contain the given original link domain.

```
Total number of files found at 1 recursion: 286 (Internal : 211, External : 75)

Internal HTML (2) :
http://www.w3schools.com//codegame/index.html : 2556
http://www.w3schools.com//spaces/index.html : 220

Internal CSS (1) :
http://www.w3schools.com//lib/w3schools32.css : 13653

Internal JS (6) :
http://www.w3schools.com//lib/uic.js?v=1.0.5 : 25076
http://www.w3schools.com//lib/my-learning.js?v=1.0.20 : 4521
http://www.w3schools.com//lib/user-session.js?v=1.0.28 : 8552
http://www.w3schools.com//lib/w3codecolor.js : 6792
http://www.w3schools.com//lib/w3schools_footer.js?update=20230706 : 13127
http://www.w3schools.com//lib/common-deps.js?v=1.0.2 : 2594

Internal JPG and JPEG (1) :
http://www.w3schools.com/w3css_templates.jpg : 61707

Internal PNG (5) :
http://www.w3schools.com//images/colorpicker2000.png : 8621
http://www.w3schools.com//images/w3lynx_200.png : 13570
http://www.w3schools.com/how-spaces-works3.png : 21231
http://www.w3schools.com//images/colorpicker.png : 13141
http://www.w3schools.com/myl-green-off.png : 41045

Internal Others (196) :
http://www.w3schools.com//aws/aws_exercises.php : 176396
http://www.w3schools.com//sql/sql_examples.asp : 223097
http://www.w3schools.com//java/java_examples.asp : 203391
http://www.w3schools.com//sass/default.php : 22023
http://www.w3schools.com/w3css/default.asp : 176239
http://www.w3schools.com//typescript/index.php : 22406
http://www.w3schools.com//js/js_exercises.asp : 188208
http://www.w3schools.com//js/js_examples.asp : 234180
http://www.w3schools.com//whatis/default.asp : 177754
http://www.w3schools.com//howto/tryhow_js_slideshow_ifr.htm : 1310
http://www.w3schools.com//bootstrap/bootstrap_exercises.asp : 22663
http://www.w3schools.com//go/go_exercises.php : 169561
```

Figure 2: Output of the program for www.w3schools.com for 1 recursion depth stored in a text file.

Extra Features

The above features were the basic requirement of the problem statement, but I have also added some extra features which better classifies the data and aid in better data visualization. These extra features are:

- Classifying the links into external and internal links, internal links are links which have same domains to that of the provided link, rest all are considered external
- The program finds the file size of each files encountered and prints or stores them if the user passes **Y** as an input to the **-f** tag. This feature is a bit computationally expensive but can greatly help in better understanding.
- There is also a feature which uses Numpy and Matplotlib to plot the **number of different file types** and **total size of different file types** into horizontal bar graphs for better data visualization:

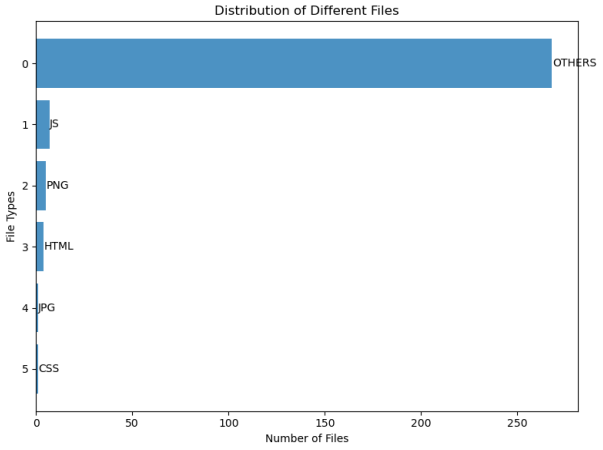

Figure 3: Plot of total number of different file types

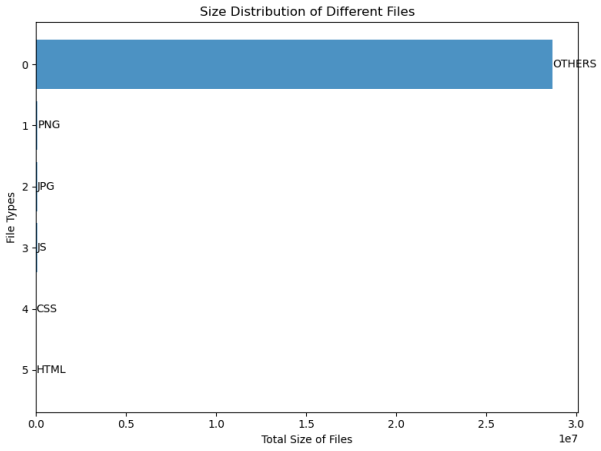

Figure 4: Plot of size distribution for different file types

- This program if used on a large website with many links for larger depths can take a lot of time, so I have introduced a feature by which the user can keep track of the processes happening in real time. The user can activate this feature by passing **Y** as an argument to **-p** tag. This feature display the number of links analysed till now and also update the user about the filesize finding process (if the user has enabled it)

References

[1] URL: `https://www.geeksforgeeks.org/draw-a-horizontal-bar-chart-with-matplotlib/`.

[2] URL: `https://www.geeksforgeeks.org/python-program-to-recursively-scrape-all-the-urls-of-the-website/`.