Ephraim Boyance-Stampley

Dr. Seales

CS 585/685

02/13/2022

HW3: crawl assignment analysis

My initial thought when designing the scraper was to figure out the website structure. This is because however those parameters for each page were set up would be how I could deduce how to crawl to the next page while also scrapping for the data that I desired. Every website is a bit different but the engr.uky.edu domain kept its pattern straightforward and consistent.

The tools I used were PyCharm, Python 3, & Scrapy. PyCharm is a Python IDE that takes an already easy to comprehend scripting language in Python and makes it easier. Python 3 is a scripting language that emphasizes simplicity while being powerful and efficient. And finally, Scrapy is a web scrapping Python library that doubles as a web crawler using its urljoin() function.

There were positives and negatives when implementing this scrapper. I'll be positive, to begin with. I think Scrapy as a python package is very powerful and useful once you hop over the learning curve. It is so flexible that I feel confident that I could use it to scrape any domain that I would like given the time to figure out the website structure. Another positive is that Scrapy keeps its own stats about each spider which helps with debugging but also allowed me to answer the question of the time of operation quickly. Finally, the last compliment I'll mention is that the website documentation might be the best I've seen from any Python package ever.

Now to go into the negatives. Scrapy was quite intimidating at first. From the outside looking in, the set-up is not very clear, but they do a great job with documenting the functionality in a way that allows you to understand the variability with the functionality. The parsing algorithm is also a bit difficult to grasp but it has less to do with Scrapy and more to do with engr.uky.edu domain. To mound this hurdle, you must heavily lean on the flexibility of the Scrapy parser to dissect the structure of the webpage. The engr.uky.edu domain has various links that need to be filtered, so you don't end up falling out of the domain. The last challenge I'll mention is figuring out how to properly format output to get the answers I wanted. The OCD computer scientist in me wanted to code in an output that gave me pretty answers but trying to get answers was not a walk in the park. I was lucky enough that the IDE that I used gave line numbers on .txt files, so I was able to figure this out using print statements and strategically filtered .txt files. t

Overall, I really enjoyed this experience and feel like I learned a lot about something that has intimidated me for so long. I've always been afraid of how my brain my react to working with big data like this, but I found the process fun and rewarding. Next time I crawl, I'll put more emphasis on planning how I'll output my data just to save myself the headache.

Below you'll find the answers to the questions you had about the engr.uky.edu domain. You also see a graph and chart laying out their file extension data.

How much time does it take to do the complete crawl of *engr.uky.edu*? 950.5807 seconds
How many pages did you crawl? 698909
How many unique URLs in the domain did your crawl encounter? 531779

UKY File Extension	Count	Percentage
.asp	12	0.16%
.com	112	1.50%
.doc	12	0.16%
.htm	47	0.63%
.ico	5303	70.91%
.ics	2	0.03%
.jpg	50	0.67%
.jsp	4	0.05%
.mp3	4	0.05%
.pdf	1670	22.33%
.php	238	3.18%
.png	7	0.09%
.ppt	2	0.03%
.xls	4	0.05%
.xml	4	0.05%
.xsn	8	0.11%