HW 1 - Webscience Intro

Jacob Conner

September 20, 2020

Q1

Consider the ”bow-tie” structure of the web in the Broder et al. paper (http://snap.stanford.edu/class/cs224w-readings/broder00bowtie.pdf) that was described in Week 1. Now consider the following links (1) Draw the resulting graph (either sketch on paper or use another tool) and include an image in your report.)

```
A --> B
B --> A
B --> C
C --> D
C --> G
D --> A
D --> H
E --> F
E --> O
F --> G
G --> P
H --> L
J --> N
K --> I
M --> A
N --> L
O --> J
P --> C
```

Listing 1: Links

For the above graph, list the nodes (in alphabetical order) that are each of the following categories:

- IN:
- SCC:
- OUT:
- Tendrils:
  - indicate if the tendril is reachable from IN or can reach OUT
- Tubes:

  - explain how the nodes serve as tubes

- Disconnected:

Answer

All figures must have a caption and must be referenced in the text. Example below.

Figure 1 was created using the *Dia* diagramming tool [1] and shows the relationships between each node.

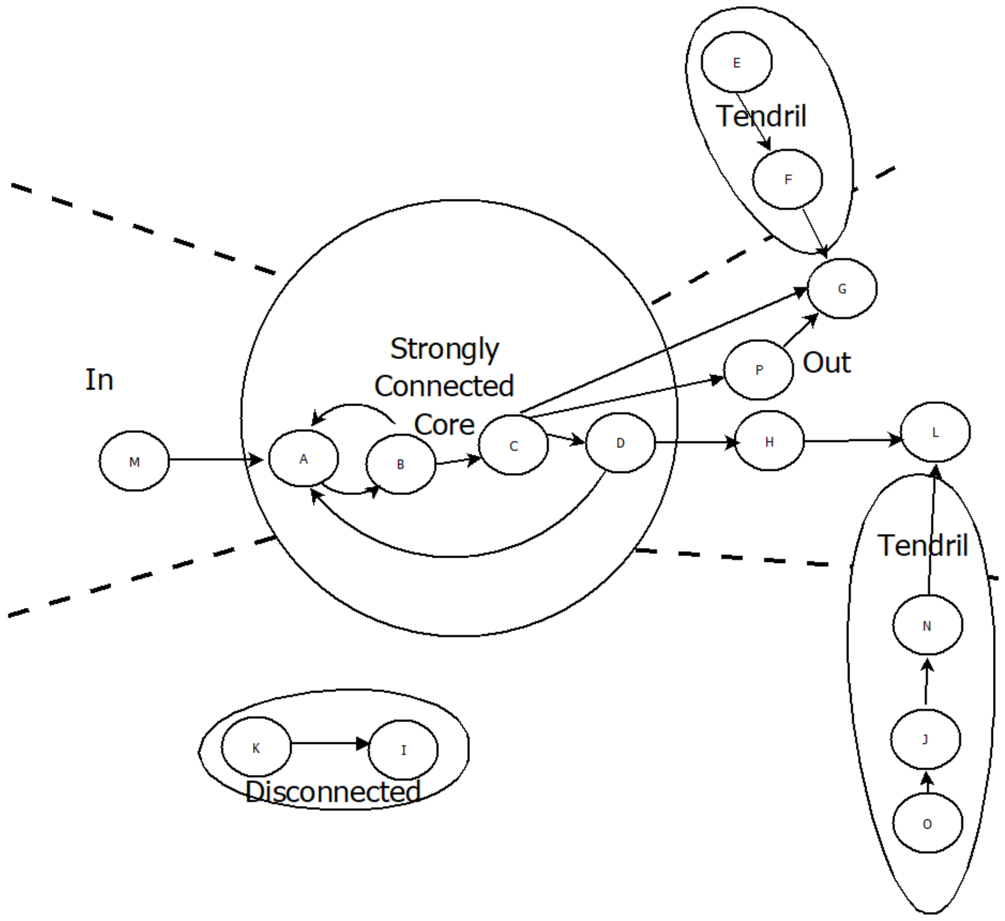


Figure 1: Bow-Tie Graph representing relationship between nodes

- IN: M
- SCC: A, B, C, D

- OUT: G, H, L,
- Tendrils : NONE
  - Reaching outward (FROM IN): NONE
  - : Reaching in (to OUT): E, F, J, N, O
- Tubes: NONE
- Disconnected: K, I

Discussion

The very first step to identify types of nodes is to identify what portions of the graph are strongly condnected. The graph is "strongly connected if there is path beween all pairs of vertices" [**?**]. In this graph there are four strongly connected nodes, A, B, C, and D. **??** lists the strongly connected component nodes (SCC) and the paths beween each of these nodes. Since these four nodes are all strongly connected, they form the Strongly Connected Component in the graph. The OUT nodes are the nodes that only link out from a strongly connected component node or another OUT node. In this case, the OUT nodes consist of nodes G and P which links out from the SCC node C, node H that links from the SCC node D, and node L that links from the OUT node H. Similarly, the IN nodes were identified as the nodes that only link in to another IN node or a SCC node. In this graph there is only one IN node, M, which links in to to the SCC node A. Tendrils are the groups of nodes that either link out from an IN node or link in to an OUT node. In this graph, there were not any nodes linking outward from the one IN node, but there were three nodes, J, N, and O, that reached in to the OUT node L. Tubes are the nodes that that link between IN and OUT nodes without going through the SCC. Since there were no nodes linking from IN to the OUT nodes, there were no tubes. Then disconnected nodes were the nodes that do not connect to a SCC, IN, or OUT noded. In this graph, there were two disconnected nodes, K and I.. Node K can connect to Node I but cannot connect to anything else and Node I serves only as a destination for Node K and does not link to anything else.

Since this graph is a miniature representation of the web, the most interesting observation we can glean from the graph is that there are several factors that will prevent search engines and web crawlers from identifying every website simply by following links. The first limiation to indexing the whole web is that some nodes are disconnected and are not linked to from any other website. Since no websites link to these disconnected ndoes, it is simply not possible to uncover these websites. A second limitation is depending on where you are in the graph, will determine how much of the remainder of the web can be uncovered. If a webcrawler started indexing at the SCC node A, it would be able to find most of the nodes in this graph, but I would never be able to reach node M since it is merely linking to A and A does not link back to M. A webcrawler would also never be able to uncover any of the tendrils since they link to OUT nodes, but the OUT nodes do not link back to them. If by some chance a webcrawler started on an OUT node then the number of websites that could be indexed would be relatively small as only OUT nodes could be indexed at that point. Clearly, through this graph, we are able to see some of the limitations of webcrawlers that follow links from one page to another

Table 1: SCC Links

col	**A**	**B**	**C**	**D**
A	X	A,B	A,B,C	A,B,C,D
B	B,A	X	B,C	B,C,D
C	C,D,A	C,D,A,B	X	C,D
D	D,A	D,A,B	D,A,B,C	X

Q2

Demonstrate that you know how to use curl and are familiar with the available options.

a) In a single curl command, request the URI, show the HTTP response headers, follow any redirects, and change the User-Agent HTTP request field to ”CS432/532”. Show command you used and the result of your execution on the command line. (Either take a screenshot of your terminal or copy/paste into a code segment.)

b) Then make the same request again, but without showing the HTTP response headers and with saving the HTML output to a file. Show the command you used and the result of your execution on the command line. View that file in a browser and take a screenshot.

c) Finally, load the URI directly from your browser and take a screenshot.

Explain the results you get for each of these steps.

Answer

a)

```
curl -i -L -A "CS432/532" http://www.cs.odu.edu/˜mweigle/courses/cs532/
   ua_echo.php
```

Listing 2: Command

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time
    Current
                                 Dload  Upload   Total   Spent    Left
    Speed
100   178  100   178    0     0   2282      0 --:--:-- --:--:--
   --:--:--  2825
```

```
100   114  100   114    0     0    561      0 --:--:-- --:--:--
   --:--:--  1461HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Sat, 29 Aug 2020 15:08:24 GMT
Content-Type: text/html
Content-Length: 178
Connection: keep-alive
Location: https://www.cs.odu.edu/~mweigle/courses/cs532/ua_echo.php

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 29 Aug 2020 15:08:24 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 114
Connection: keep-alive
Vary: Accept-Encoding
X-Powered-By: PHP/5.6.40

<!DOCTYPE html>
<html>
<body>

<br/>USER AGENT ECHO
<br/><br/>
<b>User-Agent:</b> CS432/532<br/>

</body>
</html>
```

Listing 3: curl command to get the GET response and header, redirects and send a User-Agent header propety

The curl command is used to make HTTP GET Requests. the -i or –include flag is used to specify curl to include the headers and the response. The -L or –location flag is used to request the header and response for all redirects. The -A or –user-agent flag specifies the User-Agent property in the header that is sends to the string next to it. IN this case the User-Agent sent is "CS432/5432". Then the last part of the curl command is the URI(s) for the webserver(s) that we are making a HTTP GET request to [2].

b)

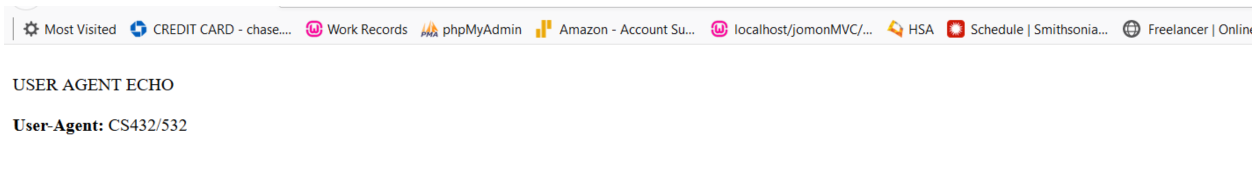
```
curl -o response.html -L -A "CS432/532" http://www.cs.odu.edu/\~mweigle
   /courses/cs532/ua\_echo.php
```

Listing 4: "curl command to get headers only for all redirects and sets the User-Agent header property"

```
<!DOCTYPE html>
<html>
<body>
```

```

<br/>USER AGENT ECHO
<br/><br/>
<b>User-Agent:</b> CS432/532<br/>

</body>
</html>
```

Listing 5: response.html



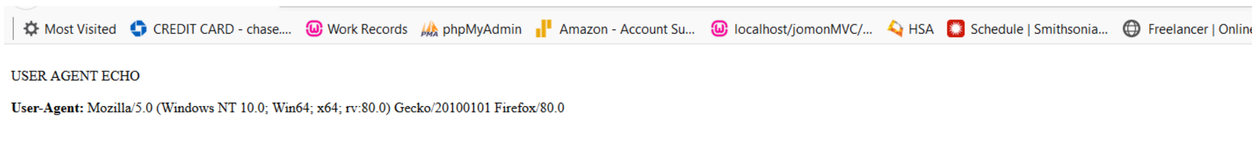
USER AGENT ECHO

User-Agent: CS432/532

Figure 2: Browser representation of the GET request response.html

This command is similar to the previous command in part a. However since we do not need the header information,, so the -i –include flag has been omitted. I have also added the -o output flag to copy the resulting response from the server into the file response.html. 5 shows this output from the GET request. The response.html file is a fairly simple html file with a body tag enclosing one set of bold tags surrounding the words User-Agent, and the value of our User-Agent property (CS432/532) in the GET request header. The browser representation of the response is displayed in 2.

c)



USER AGENT ECHO

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0

Figure 3: Browser representation of GET request to https://www.cs.odu.edu/ mweigle/courses/cs532/ua_echo.php

If a browser navigates to https://www.cs.odu.edu/ñweigle/courses/cs532/ua_echo.php, the browser sends the GET HTTP request in the background and receives the Response and creates a representation of the response which will look similar to 3. It looks very similar to the previous response.html retrieved from the curl command, but in the browser the User-Agent shows a different string. On the computer I am currently using, I am running Windows 10 64-bit and running Firefox, so the browser uses that information to send a string to the ”User-Agent” property in the header of the GET request to help the webserver identify how I am accessing this page. If I were to use a different browser such as Chrome or if I ran Firefox on a linux distribution, I would get a different string. On every GET Request submitted by the browser, this same User-Agent string is submitted.

Discussion

In this section, different methods of create HTTP GET Requests was examined. In the first two questions, the GET Request was created using the curl command and the headers had to be specified in the command line. In this case only one property was sent in the header and that was ”User-Agent”. Then in the last question, a browser was used to issue a GET request for https://www.cs.odu.edu/ mweigle/courses/cs532/ua_echo.php and the browser automatically created a User-Agent string for the header. One interesting observation from this exercise is that a webserver can very easily identify if a GET request is coming from a browser or another application because a browser sends a header with a unique ”User-Agent” string by default and other methods may not.

Q3

Write a Python program to find links to PDFs in a webpage.

Your program must do the following:

- take the URI of a webpage as a command-line argument
- extract all the links from the page
- for each link, use the Content-Type HTTP response header to determine if the link references a PDF file
- for all links that reference a PDF file, print the original URI (found in the source of the original HTML), the final URI (after any redirects), and the number of bytes in the PDF file. (Hint: Content-Length HTTP response header)

Show that the program works on 3 different URIs, one of which must be https://www.cs.odu.edu/ mweigle/courses/cs532/pdfs.html Here is a snippet of the expected operation:

```
% python3 findPDFs.py https://www.cs.odu.edu/~mweigle/courses/cs532/
   pdfs.html

URI: http://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-
   violations.pdf
```

```
Final URI: https://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-
    temporal-violations.pdf
Content Length: 2,184,076 bytes

URI: http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.
    pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-
    annotations.pdf
Content Length: 622,981 bytes
```

Listing 6: Expected Output

Answer

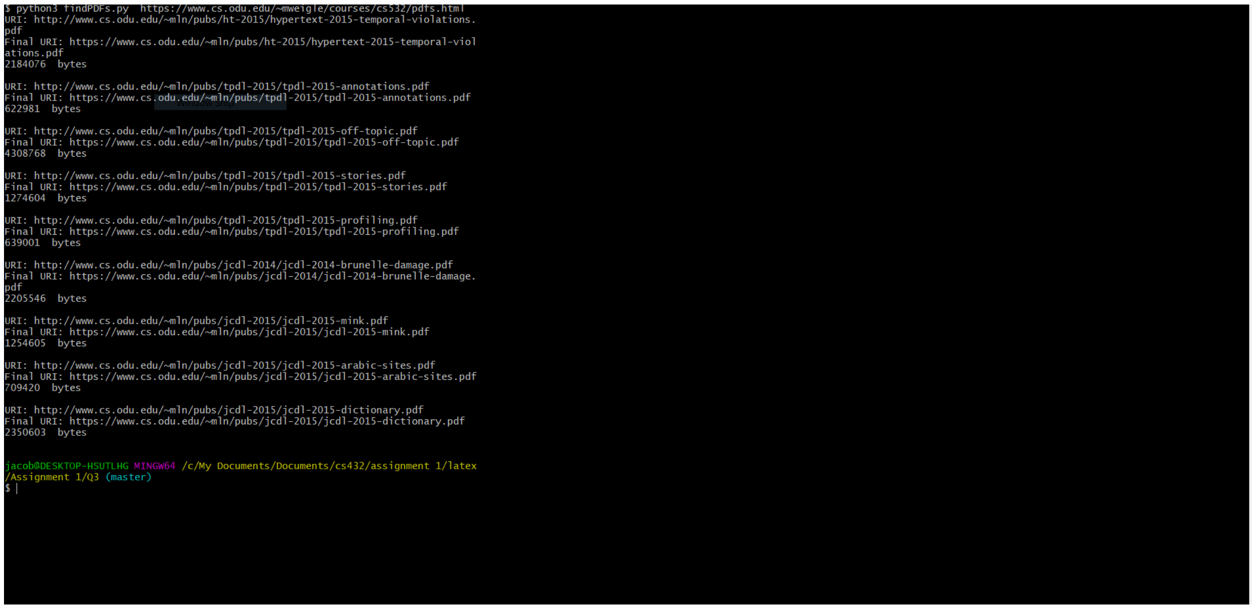
Listing 7 is the imported code I used to scrape pdf files from webpages.

```python
from bs4 import BeautifulSoup #for parsing html
import requests #for getting final uri
import random # used for User Agent Function
import sys #for command line argments
import re # for regex method of validating PDFs
from urllib.parse import urljoin  # for converting relative to absolute
    paths

#code for handling user-agent to deal with some webservers
#GET_UA() code derived from
# https://www.jcchouinard.com/random-user-agent-with-python-and-
    beautifulsoup/
# User Agent Strings obtained using https://www.whatismybrowser.com/
def GET_UA():
    uastrings = ["Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0)
    Gecko/20100101 Firefox/80.0",
                 "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit
    /537.36 (KHTML, like Gecko) Chrome/85.0.4183.83 Safari/537.36",
                 "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit
    /537.36 (KHTML, like Gecko) Chrome/85.0.4183.83 Safari/537.36 Edg
    /85.0.564.44",
                 "Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0;
    Touch; rv:11.0) like Gecko",
                 "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit
    /537.36 (KHTML, like Gecko) Chrome/84.0.4147.135 Safari/537.36 OPR
    /70.0.3728.160",
                ]
    return random.choice(uastrings)

#converts relative links to absolute paths
```

```python
# absolutePath code derived from
# https://stackoverflow.com/questions/476511/resolving-a-relative-url-
    path-to-its-absolute-path
def absolutePath(webpage, url):
    absoluteUrl = url
    if not url.startswith('http://www.'):
        absoluteUrl = requests.compat.urljoin(webpage, url)
    elif not url.startswith('https://www.'):
        absoluteUrl = requests.compat.urljoin(webpage, url)
    else:
        absoluteUrl = url
    return absoluteUrl

# code for handling system arguments
def checkForArguments():
    #code for argument handling
    try:
        webpage = sys.argv[1]
    except Exception as error:
        print("Error: A url is a required argument to find pdf links")
    return webpage

# gets response from http request or returns none
def validateRequest(url):
    headers = {'User-Agent': GET_UA()}
    response = None
    try:
        response = requests.get(url, headers=headers)
    except Exception as error:
        print("Unable to receive response from ", url)
        print("Try another webpage address")
    return response

def checkIfPDFRegEx(url):
    #use regex to check if link ends in .pdf
        m = re.search('.*(\.pdf)$', url)
        # anything matches, then must be a pdf
        if m != None:
            return url
        return None

def printResults(response, link):
    print("URI:", link) #link from anchor tag
    print("Final URI:", response.url) #obtained from response
    #check for content-length
    if 'Content-length' in response.headers:
        print(response.headers['Content-length'], ' bytes')
    else:
        print('File Size Unknown')
    print() # print a new line for aesthetics

def main():
    webpage = checkForArguments()
    if webpage == '': return
    #get headers and text for url
    response = validateRequest(webpage)
    # parse response text using Beautiful soup to get links
    links = BeautifulSoup(response.text, 'html.parser').findAll('a')

    for link in links:
        #check if relative path and convert to absolute path
        url = absolutePath(webpage, link['href'])
        #check if pdf, and get http request
        url = checkIfPDFRegEx(url)
        if(url):
            response = validateRequest(url)
            if (response):
                #To account for bad links
                #Check once more if the final uri is a pdf
                if(checkIfPDFRegEx(response.url)):
                    response = validateRequest(response.url)
                    printResults(response, url)

if __name__ == '__main__':
    main()
```

Listing 7: FindPDFs.py

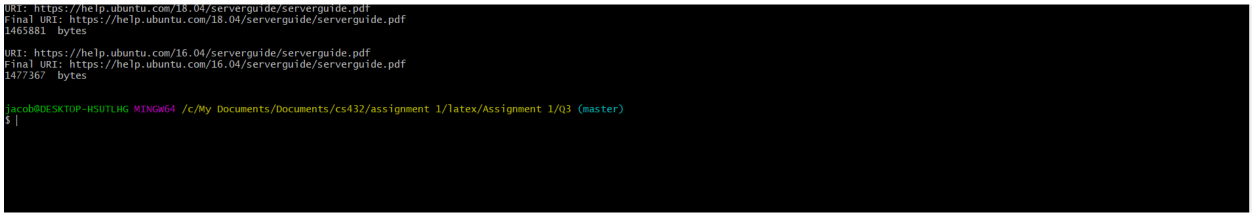
Below I have provided screenshots of various sample output from the findPDFs.py scraper. 4 is the output I received after running the findPDFs.py scraper on https://www.cs.odu.edu/ mweigle/courses/cs532/pdfs.html



```
$ python3 findPDFs.py  https://www.cs.odu.edu/~mweigle/courses/cs532/pdfs.html
URI: http://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-violations.
pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-viol
ations.pdf
2184076  bytes

URI: http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf
622981  bytes

URI: http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf
4308768  bytes

URI: http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf
1274604  bytes

URI: http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf
639001  bytes

URI: http://www.cs.odu.edu/~mln/pubs/jcdl-2014/jcdl-2014-brunelle-damage.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/jcdl-2014/jcdl-2014-brunelle-damage.
pdf
2205546  bytes

URI: http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf
1254605  bytes

URI: http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf
709420  bytes

URI: http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf
Final URI: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf
2350603  bytes


jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/latex
/Assignment 1/Q3 (master)
$ |
```

Figure 4: findPDFs.py results for https://www.cs.odu.edu/ mwei- gle/courses/cs532/pdfs.html

5 is the output I received after running the findPDFs.py scraper on the Ubuntu manual page (https://help.ubuntu.com/)

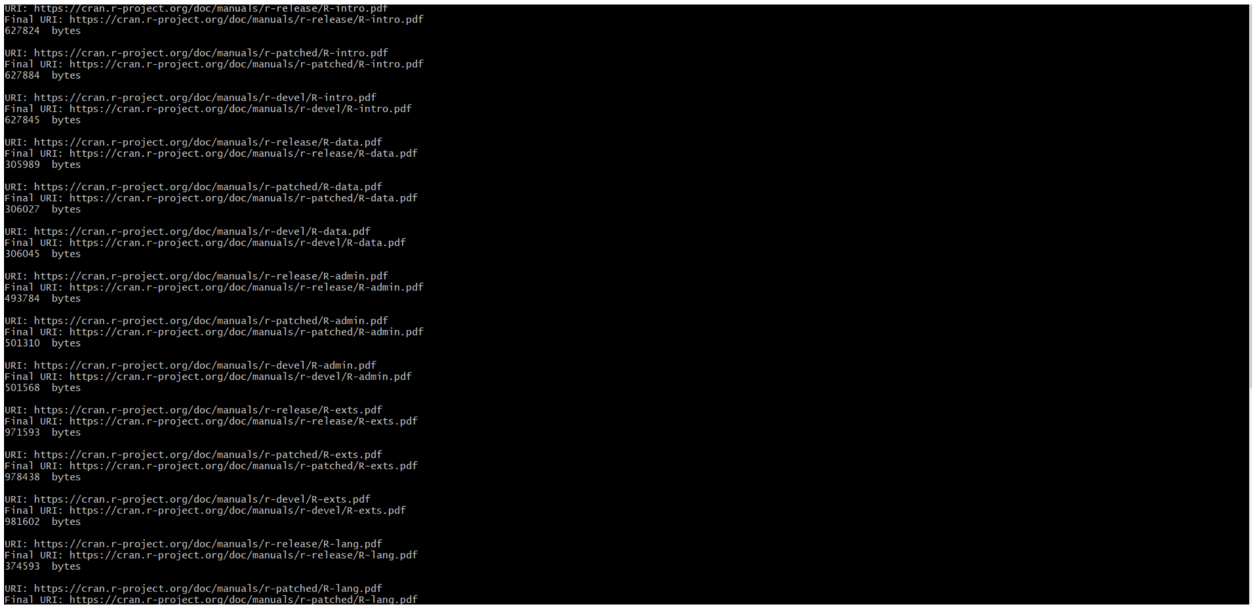


```
URI: https://help.ubuntu.com/18.04/serverguide/serverguide.pdf
Final URI: https://help.ubuntu.com/18.04/serverguide/serverguide.pdf
1465881  bytes

URI: https://help.ubuntu.com/16.04/serverguide/serverguide.pdf
Final URI: https://help.ubuntu.com/16.04/serverguide/serverguide.pdf
1477367  bytes


jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/latex/Assignment 1/Q3 (master)
$ |
```

Figure 5: findPDFs.py results for https://help.ubuntu.com/

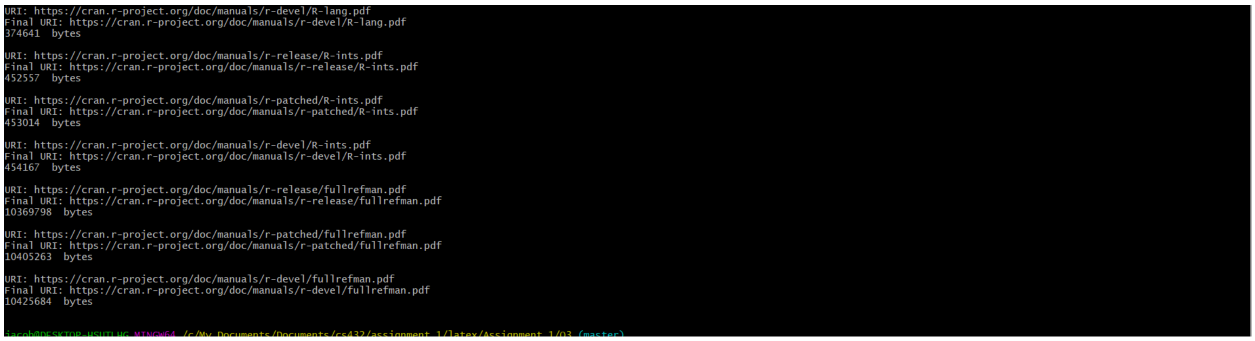
6 and 7 are screenshots of the output I received after running the findPDFs.py scraper on the CRAN-R R programming manuals page (https://cran.r-project.org/manuals.html)

8 and **??** are screenshots of the output I received after running the findPDFs.py scraper on the Society of East Asian Archaeology's 2018 bulletin page (https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology)



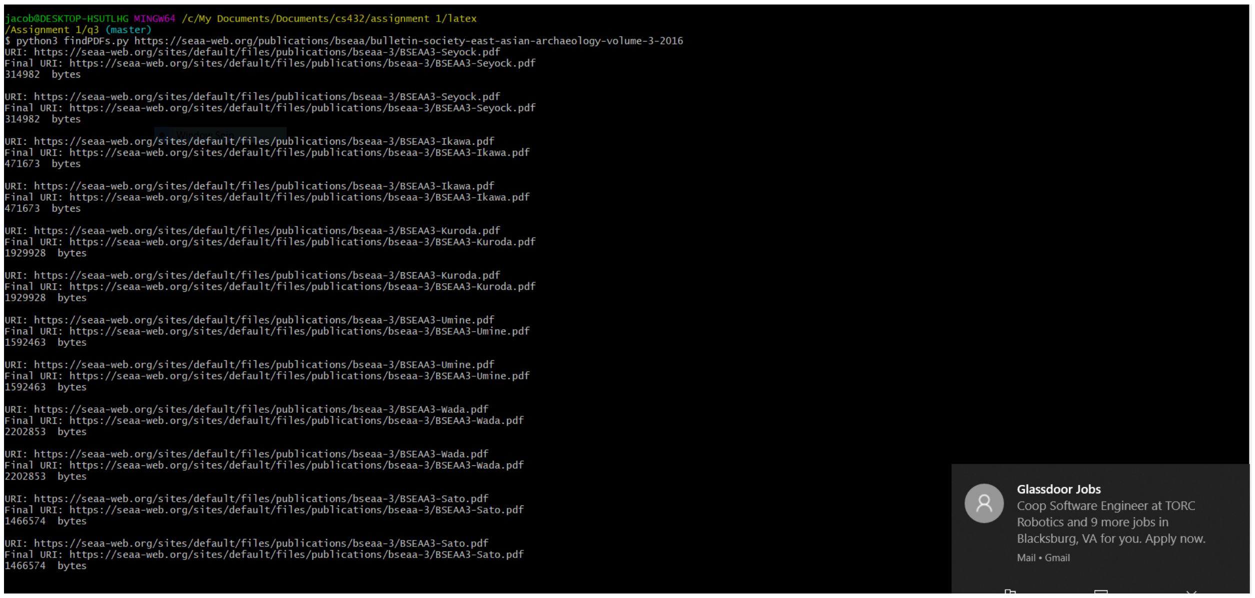
```
URI: https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf
627824  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-intro.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-intro.pdf
627884  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/R-intro.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-intro.pdf
627845  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/R-data.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-data.pdf
305989  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-data.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-data.pdf
306027  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/R-data.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-data.pdf
306045  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/R-admin.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-admin.pdf
493784  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-admin.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-admin.pdf
501310  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/R-admin.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-admin.pdf
501568  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/R-exts.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-exts.pdf
971593  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-exts.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-exts.pdf
978438  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/R-exts.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-exts.pdf
981602  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/R-lang.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-lang.pdf
374593  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-lang.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-lang.pdf
```

Figure 6: findPDFs.py results for https://cran.r-project.org/manuals.html



```
URI: https://cran.r-project.org/doc/manuals/r-devel/R-lang.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-lang.pdf
374641  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/R-ints.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/R-ints.pdf
452557  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/R-ints.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/R-ints.pdf
453014  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/R-ints.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/R-ints.pdf
454167  bytes

URI: https://cran.r-project.org/doc/manuals/r-release/fullrefman.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-release/fullrefman.pdf
10369798  bytes

URI: https://cran.r-project.org/doc/manuals/r-patched/fullrefman.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-patched/fullrefman.pdf
10405263  bytes

URI: https://cran.r-project.org/doc/manuals/r-devel/fullrefman.pdf
Final URI: https://cran.r-project.org/doc/manuals/r-devel/fullrefman.pdf
10425684  bytes


jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/latex/Assignment 1/Q3 (master)
```

Figure 7: findPDFs.py results for https://cran.r-project.org/manuals.html



```
jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/latex
/Assignment 1/q3 (master)
$ python3 findPDFs.py https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology-volume-3-2016
URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Seyock.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Seyock.pdf
314982  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Seyock.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Seyock.pdf
314982  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Ikawa.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Ikawa.pdf
471673  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Ikawa.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Ikawa.pdf
471673  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Kuroda.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Kuroda.pdf
1929928  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Kuroda.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Kuroda.pdf
1929928  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Umine.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Umine.pdf
1592463  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Umine.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Umine.pdf
1592463  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Wada.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Wada.pdf
2202853  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Wada.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Wada.pdf
2202853  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Sato.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Sato.pdf
1466574  bytes

URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Sato.pdf
Final URI: https://seaa-web.org/sites/default/files/publications/bseaa-3/BSEAA3-Sato.pdf
1466574  bytes
```

Glassdoor Jobs
Coop Software Engineer at TORC Robotics and 9 more jobs in Blacksburg, VA for you. Apply now.
Mail • Gmail

Figure 8: findPDFs.py results for https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology

Discussion

The Python code above generates a console program that accepts a url as a command-line argument from the user. If no url is provided, the program exits. A HTTP GET request is made to the URL along with a header with the ”User-Agent” property to allow the code to run on more webservers that may reject GET requests that omit this property. Once a response is received from the webserver is received, the resulting text from the response is parsed using the BeautifulSoup library to retrieve all of the anchor (’a’) tags in the document. Then every anchor’s ’href’ attribute is tested to see if it is an absolute path. If the url is a relative path, i tis converted to an absolute path. After the link is in the correct form, it is checked to see if it is a pdf and another GET Request is made for each URL. The resulting response from the GET request is used to print the URI, Final URI and file size output to the console.

In order to accomplish task, I have created a main function with 6 other functions. The main function begins with retrieving the url from the user input. The user input is handled using the checkForArguments function and it makes use of the sys library’s argv function to get the first command from the user. A conditional is used to check if the the argv variable exists and if not, it exits the program. This function only checks to make sure the url argument is provided and does not evaluate if it is a valid url or not.

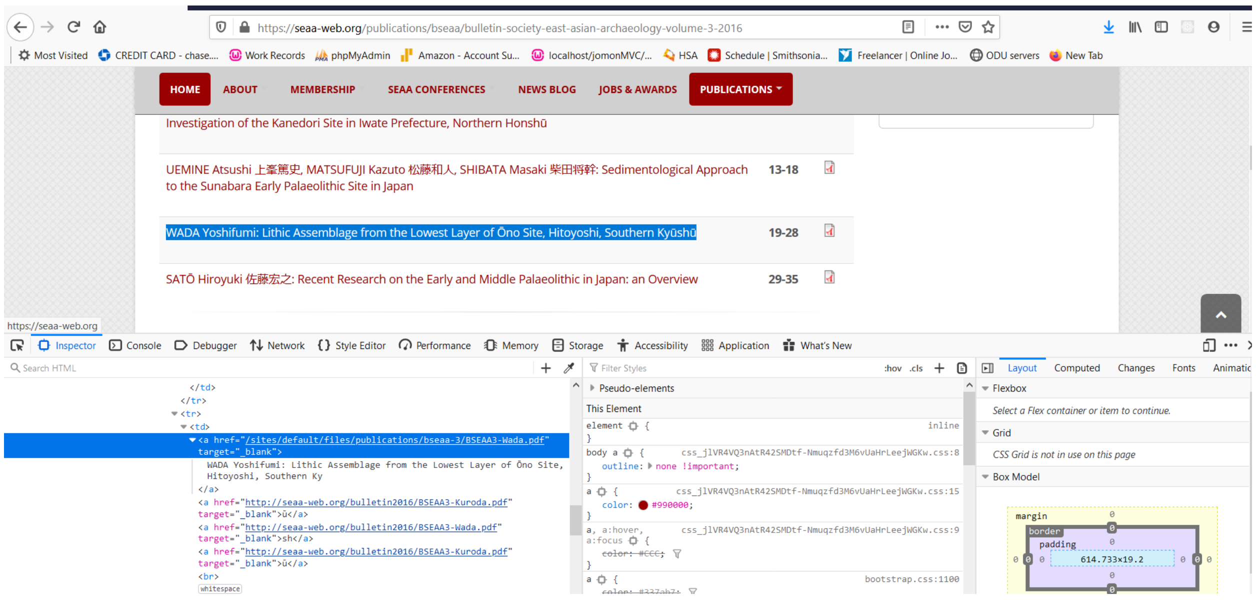
The user provided URL is then validated with the The validateRequest function. Since some webservers do not like webscrapers and will reject their GET requests, the validateRequest function begins with building a header and generates a ”User-Agent” string to mimic a browser’s ”User-Agent” string using a modified version of the GET_UA function.[3]. TheThe GET_UA function is simply a list of ”User-Agent” strings that I created using Firefox, Chrome, Edge, Opera, and Internet Explore n Windows 10. These User-Agent strings were obtained by visiting whatismyip.com [4] but technically, I could have obtained them from the website used in Question 2 as well. The GET_UA function use Python’s built in random library to pick one of these ”User-Agent” strings at random. The validateRequest function uses the resulting ”User-Agent” string to build a header. A try catch block is used to try a HTTP GET request using the requests library. If a response is received, the validateRequest function returns the response and if it fails, an Exception is generated and the function returns None.

After retrieving the response, the main function uses the BeautifulSoup library to parse the HTML to retrieve all anchor tags. There are numerous HTML parsers available including Python’s built in HTMLParser library, but in order to make use of that library, a subclass of HTMLParser has to be created and we must override various methods to get the anchor tags. The BeautifulSoup library is able to do this with a single line that initializes the BeautifulSoup object with theGET request’s response text and then calls the FindAll method to retrieve all anchor tags. Due tothis simplicity, BeautifulSoup was used. BeautifulSoup returns a list-like object that is iterated through using a for loop in order to check each anchor tag’s ’href’ attribute.

Now that I have a list of links, I need to format them so they are ready for the response requests. First, I verify that the url is an absolute path, because many websites like to use relative paths to make the code easier to port between testing and production environments. I have modified an absolutePath function [5] to check if a url starts with ”http://www” or ”https://www” and if not, the function assumes the link found was a relative url. The urljoin method is used from the UrlLib library to convert the relative url to an absolute url.

Next, next the checkIfPDFRegEx function is used to check if the link is a pdf. This function uses regular expressions with the re libary to check if a url ends in ’.pdf’ and if it ends in .pdf, the function assumes the link is a pdf. This will find all pdfs as long as the file has the extension. An alternative approach to checking for PDFs is to make a GET or HEAD request using the requests library to read the ”Content-Type” property from the header. If the property exists, you can check if it is is ”application/pdf” and if so, the response must be a pdf. Unfortunately, the ”Content-Type” property does not always exist in every header and it requires a lot of extra GET or HEAD requests that could lead to a webserver blocking all requests from the webscraper. Since thiis approaach does not always work, I abandoned the Header approach for PDF identification for the regular expression approach. After verifying the url is a pdf link, the checkIfPDFRegEx function calls the validateRequest function to generate a new GET request and returns the response. If a link is not a PDF, then the function returns None.

I added one more check in the main function to make sure a pdf link is valid by retrieving the response from the GET request and verifying the final URI in the header ends in a .pdf by running the checkIfPDFRegEx function on the response’s url property. If the final URI is not a link ending in .pdf, then the link is likely mistyped on the website or links to a location that no longer exists. I ran into this issue while visiting the Society for East Asian Archaeology’s site [6]. 9, shows three links that appaer to be applied at the end of the link to the article by Wada on this page. I examined these links using curl and received output sunukar to 10, which reveals these links ending in .pdf are ultimately linking back to the page that contained those links, which is common on some webservers when typos occur on links. HTML typos like this occur all of the time, so this additional step to validate the final URI from the response was included in this program.

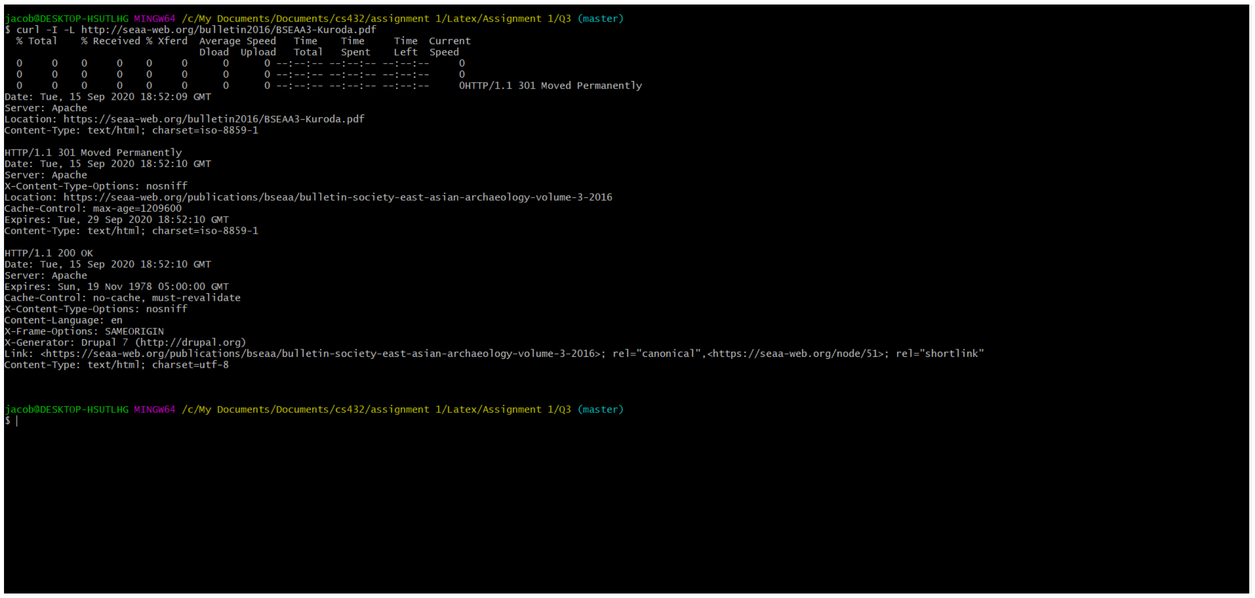

https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology-volume-3-2016

Investigation of the Kanedori Site in Iwate Prefecture, Northern Honshū

UEMINE Atsushi 上峯篤史, MATSUFUJI Kazuto 松藤和人, SHIBATA Masaki 柴田将幹: Sedimentological Approach to the Sunabara Early Palaeolithic Site in Japan	13-18	
WADA Yoshifumi: Lithic Assemblage from the Lowest Layer of Ōno Site, Hitoyoshi, Southern Kyūshū	19-28	
SATŌ Hiroyuki 佐藤宏之: Recent Research on the Early and Middle Palaeolithic in Japan: an Overview	29-35	

```
      </td>
    </tr>
  ▾ <tr>
    ▾ <td>
      ▾ <a href="/sites/default/files/publications/bseaa-3/BSEAA3-Wada.pdf"
        target="_blank">
          WADA Yoshifumi: Lithic Assemblage from the Lowest Layer of Ōno Site,
          Hitoyoshi, Southern Ky
        </a>
        <a href="http://seaa-web.org/bulletin2016/BSEAA3-Kuroda.pdf"
        target="_blank">ū</a>
        <a href="http://seaa-web.org/bulletin2016/BSEAA3-Wada.pdf"
        target="_blank">sh</a>
        <a href="http://seaa-web.org/bulletin2016/BSEAA3-Kuroda.pdf"
        target="_blank">ū</a>
        <br>
        whitespace
```

Figure 9: Html Errors on SEAA website with incorrect pdf links.

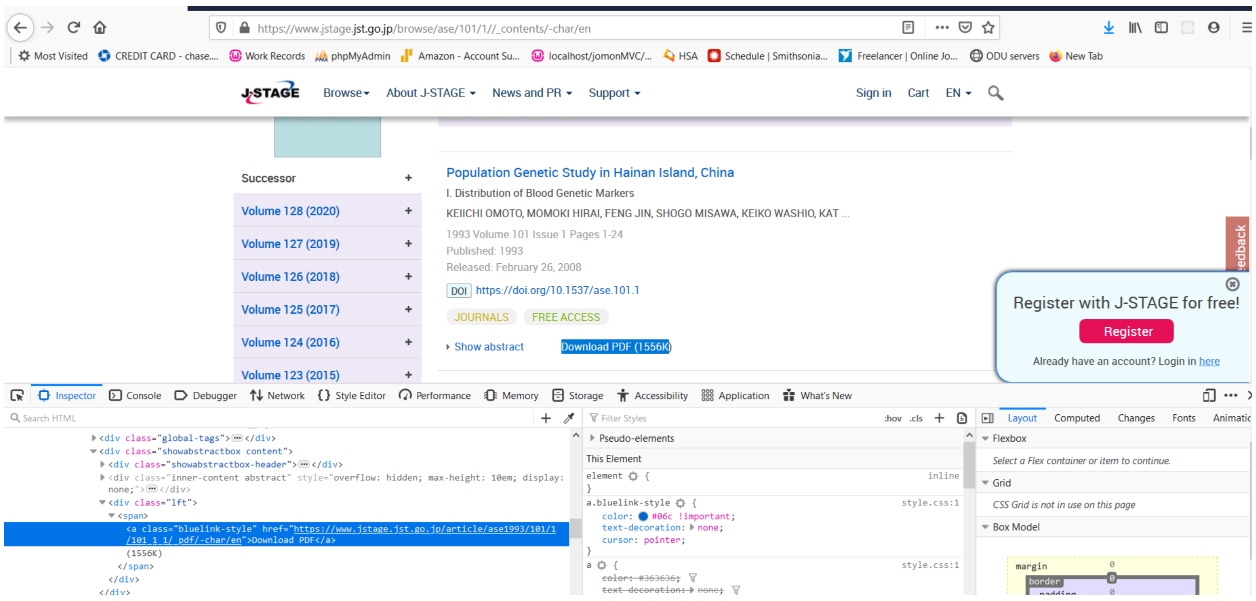


```
jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/Latex/Assignment 1/Q3 (master)
$ curl -I -L http://seaa-web.org/bulletin2016/BSEAA3-Kuroda.pdf
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0HTTP/1.1 301 Moved Permanently
Date: Tue, 15 Sep 2020 18:52:09 GMT
Server: Apache
Location: https://seaa-web.org/bulletin2016/BSEAA3-Kuroda.pdf
Content-Type: text/html; charset=iso-8859-1

HTTP/1.1 301 Moved Permanently
Date: Tue, 15 Sep 2020 18:52:10 GMT
Server: Apache
X-Content-Type-Options: nosniff
Location: https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology-volume-3-2016
Cache-Control: max-age=1209600
Expires: Tue, 29 Sep 2020 18:52:10 GMT
Content-Type: text/html; charset=iso-8859-1

HTTP/1.1 200 OK
Date: Tue, 15 Sep 2020 18:52:10 GMT
Server: Apache
Expires: Sun, 19 Nov 1978 05:00:00 GMT
Cache-Control: no-cache, must-revalidate
X-Content-Type-Options: nosniff
Content-Language: en
X-Frame-Options: SAMEORIGIN
X-Generator: Drupal 7 (http://drupal.org)
Link: <https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology-volume-3-2016>; rel="canonical",<https://seaa-web.org/node/51>; rel="shortlink"
Content-Type: text/html; charset=utf-8


jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/Latex/Assignment 1/Q3 (master)
$ |
```

Figure 10: Html Errors on SEAA website with incorrect pdf links.

The main function ends by calling a the printResult function on all of the responses returned from checkIfPDFRegEx. This function displays the original URI, the final URI (after any redirects) and the size of the pdf file. The URI is created using the original link url (after converting to an absolute path). The final URI is obtained using the GET request response's url property. The requests library by default will set the url to the final redirect unless the GET request is sent with an additional argument of "allow-redirects=False". If allow-redirects is set to false, then GET requests would have to be made on the resulting response.url until the response."status_code" is no longer 300, 301, 302, 303, 307, or 308. This is quite inefficient and lots of GET requests to a server in a short period of time, increases the likleihood of the server blocking the webscraper. Instead, a normal GET request using the requests library can be used and the url property of the response is the final URI after any redirects. If a history of redirects is needed, the history property of the response can be looped through to get all URIs in between [7]. The file size of the PDF is found using the 'Content-length' found in the response's header. Since header properties can be absent on certain sites, I made use of a loop to check if it exists, prior to displing the file size to avoid an exception. Technically everything used in the printResults can be used using a HEAD Request, however some webservers in an attempt to avoid scrapers do not allow HEAD requests, so a GET request was used. Once all links to pdfs have been displayed, the program ends.

This code is a fairly decent general purpose webscraping tool for pdfs on most web servers. However I found a few limitations. Some web servers will generate the "'Unable to receive response from javascript:void(0)". For example I attempted to run the code on Google Scholar search results for Jomon (a prehistoric culture in Japan) [8] and the program generated this exception, but Google Scholar has not been consistent as I have been able to get successful runs on some pages, and other pages will errorneously find no PDFs. In cases like this, Google's server has blocked my webscraper from running the GET Requests because it is able to identify this is a scraper and not a regular browser. On clever webservers like Google, the requests library will not work and a headless browser approach using a library like Selenium would have to be used. Another limitation I came across and was suprised to see was that some web servers will load files without the file extension. For example, JSTAGE, a Japanese Journal database that provides free access to various articles in English and Japanese links to pdf files but the URIS provided do not end in .pdf as seen in 11. The names of the pdf are obscured and only visible through the header in a ”content-disposition” property as seen in 12.These types of links are hard to identify and can't be found using this program. The only way to find these pdfs is to check the content-type in the header if it is available or to tailor your scraper to a single website. Websites with lots of pds tend to follow patterns for CSS formatting and javascript purposes, so it is possible all anchor tags linking to pdfs may be stored under a div with a unique class. While this scraper should find most pdf files on most websites, clearly there are still going to be cases where a specialized webscraper or headless browser solution will have to be used to get all pdf files.

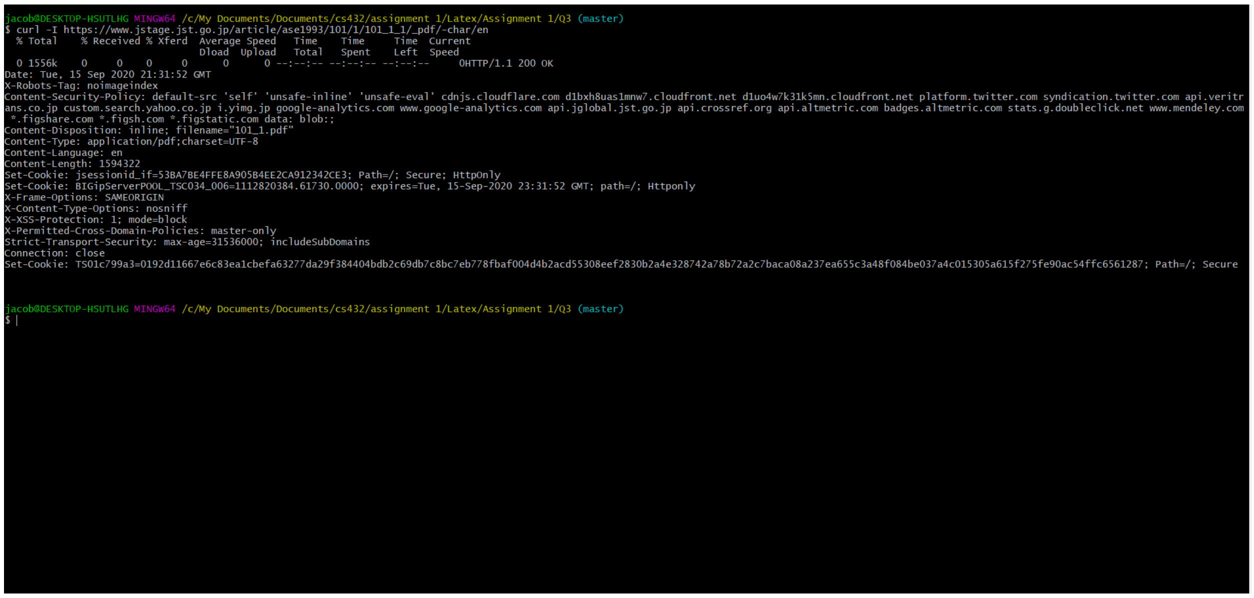


Population Genetic Study in Hainan Island, China
I. Distribution of Blood Genetic Markers
KEIICHI OMOTO, MOMOKI HIRAI, FENG JIN, SHOGO MISAWA, KEIKO WASHIO, KAT ...
1993 Volume 101 Issue 1 Pages 1-24
Published: 1993
Released: February 26, 2008
DOI https://doi.org/10.1537/ase.101.1
JOURNALS FREE ACCESS
Show abstract Download PDF (1556K)

```
<div class="global-tags">⋯</div>
<div class="showabstractbox content">
  <div class="showabstractbox-header">⋯</div>
  <div class="inner-content abstract" style="overflow: hidden; max-height: 10em; display:
  none;">⋯</div>
  <div class="lft">
    <span>
      <a class="bluelink-style" href="https://www.jstage.jst.go.jp/article/ase1993/101/1
      /101_1_1/_pdf/-char/en">Download PDF</a>
      (1556K)
    </span>
  </div>
</div>
```

```
Pseudo-elements
This Element
element {                                  inline
}
a.bluelink-style {                         style.css:1
  color: #06c !important;
  text-decoration: none;
  cursor: pointer;
}
a {                                        style.css:1
  ~~color: #363636;~~
  ~~text-decoration: none;~~
```

Figure 11: PDF links that do not end in .pdf on the JSTAGE database



```
jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/Latex/Assignment 1/Q3 (master)
$ curl -I https://www.jstage.jst.go.jp/article/ase1993/101/1/101_1_1/_pdf/-char/en
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0 1556k    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0HTTP/1.1 200 OK
Date: Tue, 15 Sep 2020 21:31:52 GMT
X-Robots-Tag: noimageindex
Content-Security-Policy: default-src 'self' 'unsafe-inline' 'unsafe-eval' cdnjs.cloudflare.com d1bxh8uas1mnw7.cloudfront.net d1uo4w7k31k5mn.cloudfront.net platform.twitter.com syndication.twitter.com api.veritr
ans.co.jp custom.search.yahoo.co.jp i.yimg.jp google-analytics.com www.google-analytics.com api.jglobal.jst.go.jp api.crossref.org api.altmetric.com badges.altmetric.com stats.g.doubleclick.net www.mendeley.com
 *.figshare.com *.figsh.com *.figstatic.com data: blob:;
Content-Disposition: inline; filename="101_1.pdf"
Content-Type: application/pdf;charset=UTF-8
Content-Language: en
Content-Length: 1594322
Set-Cookie: jsessionid_if=53BA7BE4FFE8A905B4EE2CA912342CE3; Path=/; Secure; HttpOnly
Set-Cookie: BIGipServerPOOL_TSC034_006=1112820384.61730.0000; expires=Tue, 15-Sep-2020 23:31:52 GMT; path=/; Httponly
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Permitted-Cross-Domain-Policies: master-only
Strict-Transport-Security: max-age=31536000; includeSubDomains
Connection: close
Set-Cookie: TS01c799a3=0192d11667e6c83ea1cbefa63277da29f384404bdb2c69db7c8bc7eb778fbaf004d4b2acd55308eef2830b2a4e328742a78b72a2c7baca08a237ea655c3a48f084be037a4c015305a615f275fe90ac54ffc6561287; Path=/; Secure


jacob@DESKTOP-HSUTLHG MINGW64 /c/My Documents/Documents/cs432/assignment 1/Latex/Assignment 1/Q3 (master)
$ |
```

Figure 12: Headers for a pdf on JSTAGE with filename stored in content-disposition property

References

[1] The Dia Developers. Dia diagram editor. `http://dia-installer.de/index.html.en`. Accessed on 2020-9-07.

[2] curl. curl.1 the man page. `https://curl.haxx.se/docs/manpage.html`. Accessed on 2020-9-07.

[3] JR Oakes. Randomize user-agent with python and beautifulsoup (by jr oakes). `https://www.jcchouinard.com/random-user-agent-with-python-and-beautifulsoup/`, February 2020. Accessed on 2020-9-07.

[4] Whatismyip.com user agent info. `https://www.whatismyip.com/user-agent-info/`. Accessed on 2020-9-07.

[5] StackOverflow. Resolving a relative url path to its absolute path. `https://stackoverflow.com/questions/476511/resolving-a-relative-url-path-to-its-absolute-path`. Accessed on 2020-9-07.

[6] Bulletin of the society for east asian archaeology volume 3 (2016). `https://seaa-web.org/publications/bseaa/bulletin-society-east-asian-archaeology-volume-3-2016`. Accessed on 2020-9-07.

[7] StackOverflow. Python requests library redirect new url. https://stackoverflow.com/questions/20475552/python-requests-library-redirect-new-url. Accessed on 2020-9-07.

[8] Google. Google scholar. https://scholar.google.com/scholar?hl=en&as_sdt=0%2C47&q=jomon&btnG=. Accessed on 2020-9-07.