Old Dominion University

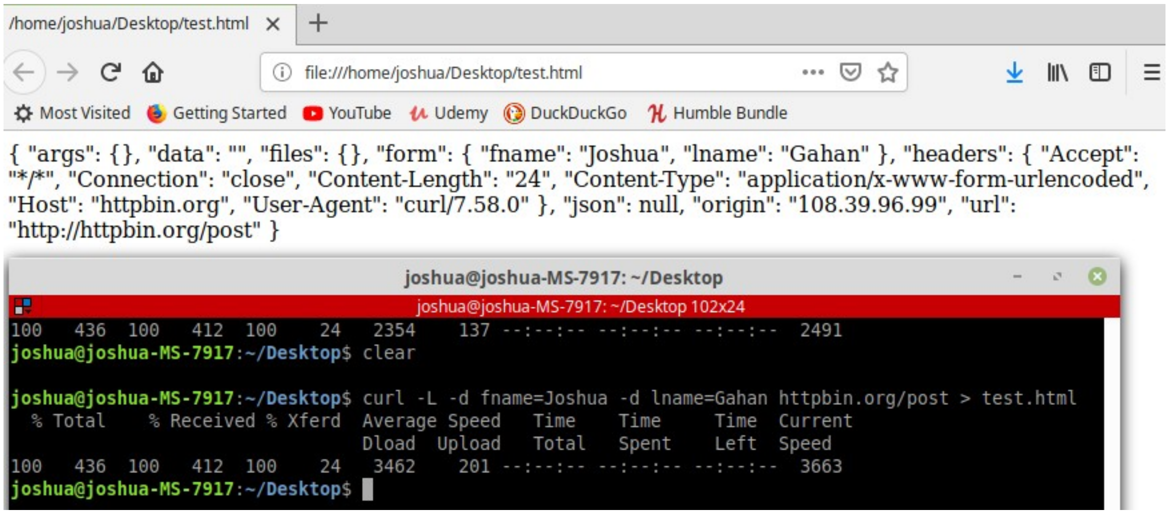
Assignment one

Joshua Gahan

January 29, 2019

1 Curl Post

In this section we demonstrate knowledge of the curl command by using it to post data to httpbin.org/post


/home/joshua/Desktop/test.html
file:///home/joshua/Desktop/test.html
Most Visited
Getting Started
YouTube
Udemy
DuckDuckGo
Humble Bundle
joshua@joshua-MS-7917: ~/Desktop
joshua@joshua-MS-7917: ~/Desktop 102x24

```
{ "args": {}, "data": "", "files": {}, "form": { "fname": "Joshua", "lname": "Gahan" }, "headers": { "Accept":
"*/*", "Connection": "close", "Content-Length": "24", "Content-Type": "application/x-www-form-urlencoded",
"Host": "httpbin.org", "User-Agent": "curl/7.58.0" }, "json": null, "origin": "108.39.96.99", "url":
"http://httpbin.org/post" }
```

```
100   436  100   412  100    24   2354    137 --:--:-- --:--:-- --:--:--  2491
joshua@joshua-MS-7917:~/Desktop$ clear

joshua@joshua-MS-7917:~/Desktop$ curl -L -d fname=Joshua -d lname=Gahan httpbin.org/post > test.html
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   436  100   412  100    24   3462    201 --:--:-- --:--:-- --:--:--  3663
joshua@joshua-MS-7917:~/Desktop$ 
```

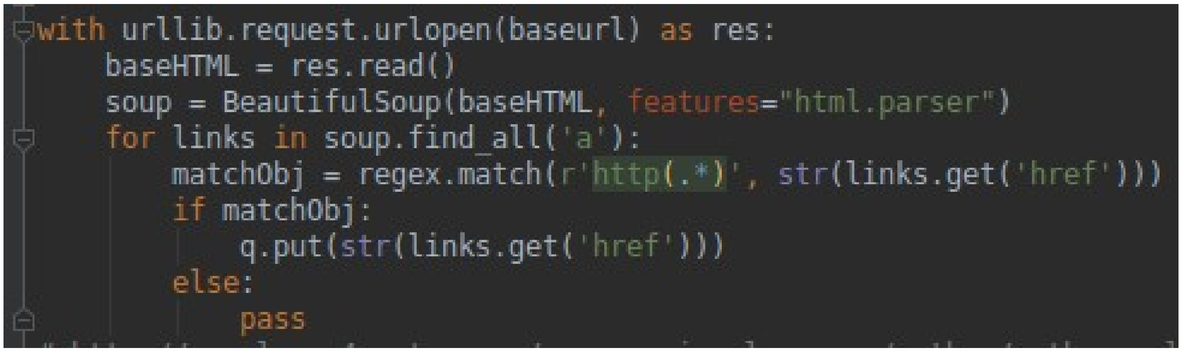
Figure 1: Curl and Post

2 Python Script

Here we discuss and demonstrate the usage of python to find and retrieve the byte size of pdf files found in links within a user provided url.

2.1 Retrieval of URIs in Seed URI for parsing

For retrieving the URIs within the seed URI we utilize the popular BeautifulSoup library. After retrieving the URIs within <a> tags, we then apply regex matching to filter out dummy hrefs.



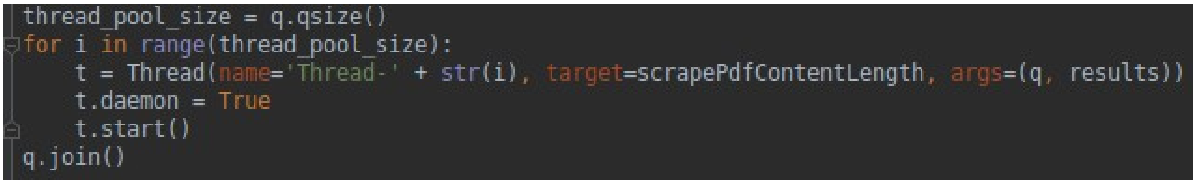
```python
with urllib.request.urlopen(baseurl) as res:
    baseHTML = res.read()
    soup = BeautifulSoup(baseHTML, features="html.parser")
    for links in soup.find_all('a'):
        matchObj = regex.match(r'http(.*)', str(links.get('href')))
        if matchObj:
            q.put(str(links.get('href')))
        else:
            pass
```

Figure 2: Retrieve URIs

Links that have passed regex matching are then pushed onto ”q,” an object of class Queue. This object is used as will be seen below.

2.2 Task allocation of found URIs

For this project, we have chosen to utilize multithreading. We feel that this was the correct decision, as the scraping operation is I/O bound and we can significantly improve the speed of the program with a little extra overhead. In the following, we dynamically create worker threads which pull URIs off the Queue object discussed above. These workers then perform the given ”scrapePdfContentLength” function logic. Finally we pause with q.join() and await thread completion.

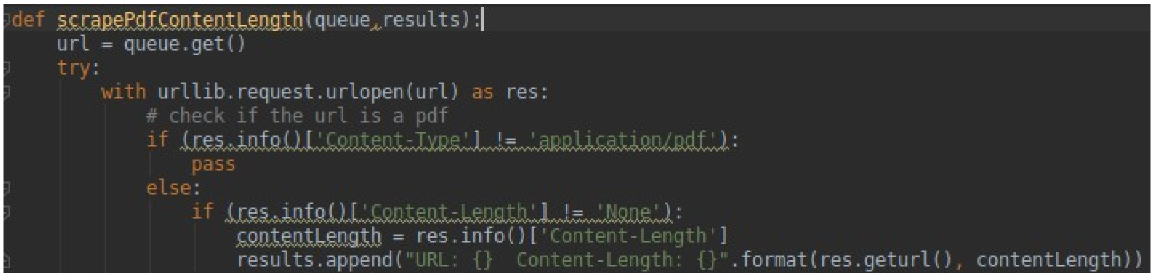


```python
thread_pool_size = q.qsize()
for i in range(thread_pool_size):
    t = Thread(name='Thread-' + str(i), target=scrapePdfContentLength, args=(q, results))
    t.daemon = True
    t.start()
q.join()
```

Figure 3: Spawning workers

2.3 Verify PDF type and retrieve byte size

Within each thread, we perform the following function. We first check to ensure that the MIME type is equal to application/pdf and the ’Content-Length’ header is present. If this is the case, a formatted string is pushed onto the results list for output within the main program. It is important to note here that python’s list datatype is threadsafe.

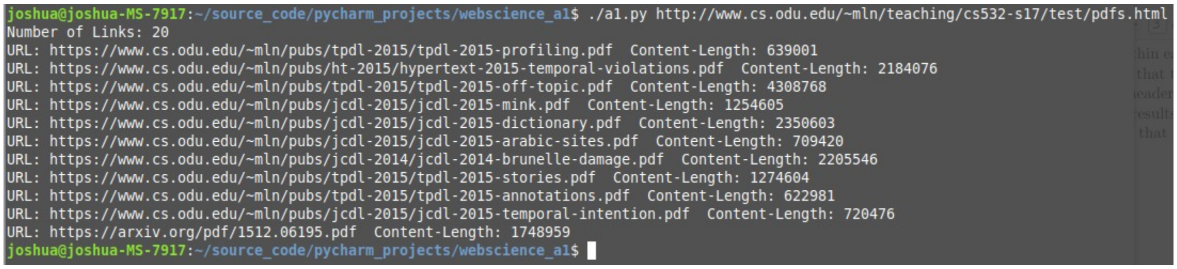


```python
def scrapePdfContentLength(queue,results):
    url = queue.get()
    try:
        with urllib.request.urlopen(url) as res:
            # check if the url is a pdf
            if (res.info()['Content-Type'] != 'application/pdf'):
                pass
            else:
                if (res.info()['Content-Length'] != 'None'):
                    contentLength = res.info()['Content-Length']
                    results.append("URL: {}  Content-Length: {}".format(res.geturl(), contentLength))
```

Figure 4: Parse and verify candidate links

2.4 Demonstration of working program

2.4.1 http://www.cs.odu.edu/ mln/teaching/cs532-s17/test/pdfs.html

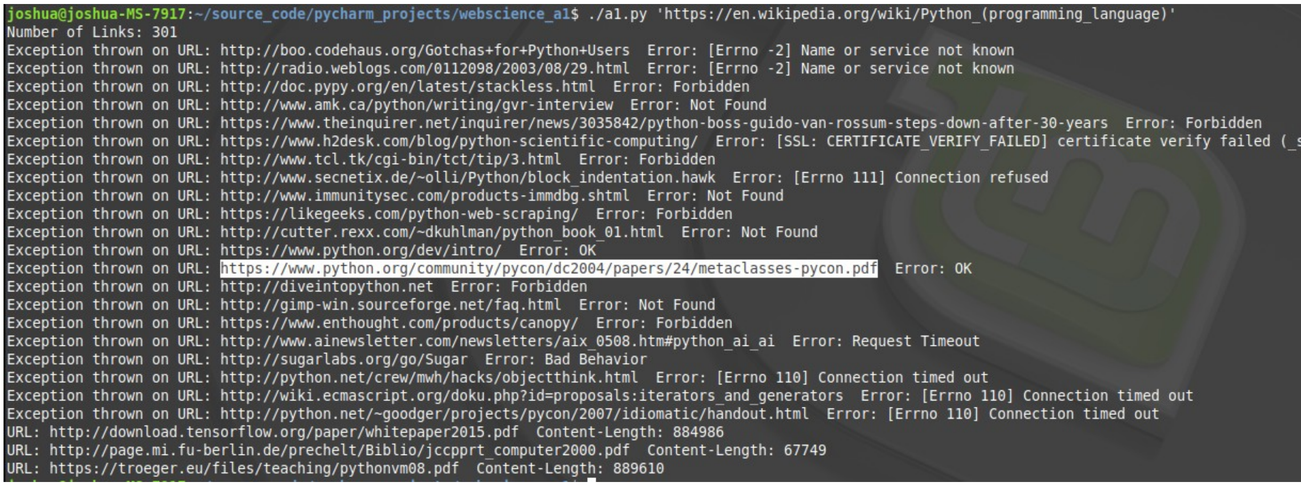


```
joshua@joshua-MS-7917:~/source_code/pycharm_projects/webscience_a1$ ./a1.py http://www.cs.odu.edu/~mln/teaching/cs532-s17/test/pdfs.html
Number of Links: 20
URL: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf  Content-Length: 639001
URL: https://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-violations.pdf  Content-Length: 2184076
URL: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf  Content-Length: 4308768
URL: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf  Content-Length: 1254605
URL: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf  Content-Length: 2350603
URL: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf  Content-Length: 709420
URL: https://www.cs.odu.edu/~mln/pubs/jcdl-2014/jcdl-2014-brunelle-damage.pdf  Content-Length: 2205546
URL: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf  Content-Length: 1274604
URL: https://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf  Content-Length: 622981
URL: https://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-temporal-intention.pdf  Content-Length: 720476
URL: https://arxiv.org/pdf/1512.06195.pdf  Content-Length: 1748959
joshua@joshua-MS-7917:~/source_code/pycharm_projects/webscience_a1$ 
```

Figure 5: Running Script on required URL

2.4.2 https://en.wikipedia.org/wiki/Python_(programming_language)

Demonstrated on a large seed URL filled with many broken links

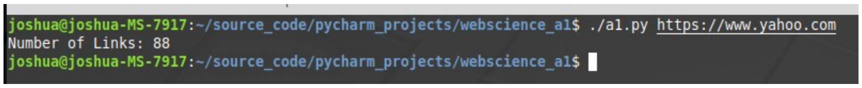


```
joshua@joshua-MS-7917:~/source_code/pycharm_projects/webscience_a1$ ./a1.py 'https://en.wikipedia.org/wiki/Python_(programming_language)'
Number of Links: 301
Exception thrown on URL: http://boo.codehaus.org/Gotchas+for+Python+Users  Error: [Errno -2] Name or service not known
Exception thrown on URL: http://radio.weblogs.com/0112098/2003/08/29.html  Error: [Errno -2] Name or service not known
Exception thrown on URL: http://doc.pypy.org/en/latest/stackless.html  Error: Forbidden
Exception thrown on URL: http://www.amk.ca/python/writing/gvr-interview  Error: Not Found
Exception thrown on URL: https://www.theinquirer.net/inquirer/news/3035842/python-boss-guido-van-rossum-steps-down-after-30-years  Error: Forbidden
Exception thrown on URL: https://www.h2desk.com/blog/python-scientific-computing/  Error: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed (_s
Exception thrown on URL: http://www.tcl.tk/cgi-bin/tct/tip/3.html  Error: Forbidden
Exception thrown on URL: http://www.secnetix.de/~olli/Python/block_indentation.hawk  Error: [Errno 111] Connection refused
Exception thrown on URL: http://www.immunitysec.com/products-immdbg.shtml  Error: Not Found
Exception thrown on URL: https://likegeeks.com/python-web-scraping/  Error: Forbidden
Exception thrown on URL: http://cutter.rexx.com/~dkuhlman/python_book_01.html  Error: Not Found
Exception thrown on URL: https://www.python.org/dev/intro/  Error: OK
Exception thrown on URL: https://www.python.org/community/pycon/dc2004/papers/24/metaclasses-pycon.pdf  Error: OK
Exception thrown on URL: http://diveintopython.net  Error: Forbidden
Exception thrown on URL: http://gimp-win.sourceforge.net/faq.html  Error: Not Found
Exception thrown on URL: https://www.enthought.com/products/canopy/  Error: Forbidden
Exception thrown on URL: http://www.ainewsletter.com/newsletters/aix_0508.htm#python_ai_ai  Error: Request Timeout
Exception thrown on URL: http://sugarlabs.org/go/Sugar  Error: Bad Behavior
Exception thrown on URL: http://python.net/crew/mwh/hacks/objectthink.html  Error: [Errno 110] Connection timed out
Exception thrown on URL: http://wiki.ecmascript.org/doku.php?id=proposals:iterators_and_generators  Error: [Errno 110] Connection timed out
Exception thrown on URL: http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html  Error: [Errno 110] Connection timed out
URL: http://download.tensorflow.org/paper/whitepaper2015.pdf  Content-Length: 884986
URL: http://page.mi.fu-berlin.de/prechelt/Biblio/jccpprt_computer2000.pdf  Content-Length: 67749
URL: https://troeger.eu/files/teaching/pythonvm08.pdf  Content-Length: 889610
```

Figure 6: Running Script on Python Wikipedia page

2.4.3 https://www.yahoo.com

Finally, a run of the script on a url that you would not expect to find any pdfs on.



```
joshua@joshua-MS-7917:~/source_code/pycharm_projects/webscience_a1$ ./a1.py https://www.yahoo.com
Number of Links: 88
joshua@joshua-MS-7917:~/source_code/pycharm_projects/webscience_a1$
```

Figure 7: Running Script on yahoo.com

3 bow-tie graph

Here we analyze an edge graph of a theoretical network and identify nodes that belong to the following categories: IN, SCC, OUT, TENDRILS, TUBES, and Disconnected. Nodes were categorized by the criteria found at https://www.harding.edu/fmccown/classes/archive/comp475-s13/web-structure-homework.pdf . Where appropriate, phrasing has been borrowed from these definitions.

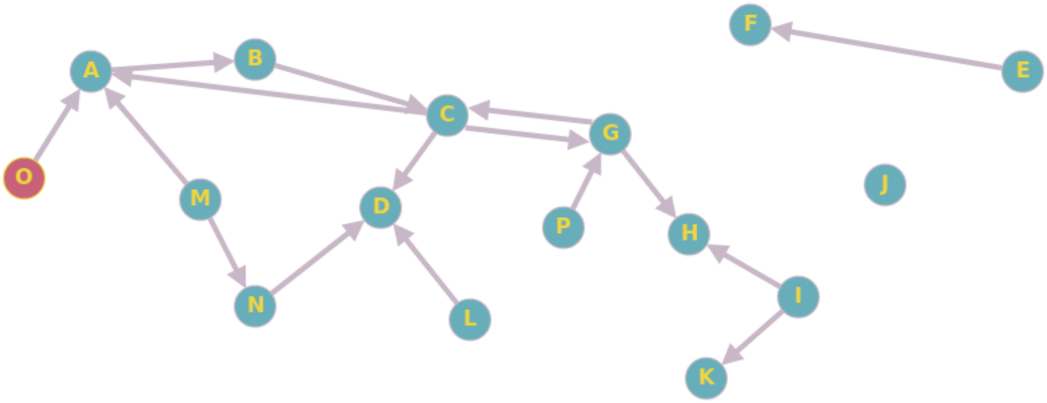


Figure 8: Theoretical Graph, graph built with http://graphonline.ru/en/

3.1 IN

Our IN nodes are O, P, I, and M. These nodes have out-links to SCC, Tendrils or tubes, but have no in-links from IN pages.

3.2 SCC

Our SCC nodes are A, B, C, and G. Each of these nodes can access the others through some path, and all have in-links either from INs or other SCCs.

3.3 OUT

Our OUT nodes are H, K, and D. Each of these nodes can be accessed by in-links but have no out-links to other nodes.

3.4 Tendrils

Our Tendril is N . This node can only be reached from M, and it only has an out-link to D (and OUT)

3.5 Tubes

N also qualifies as a tube. it has in-links from from M (IN) and an out-link to D (OUT). It is not connected to any SCC.

3.6 Disconnected

Our disconnected nodes are J, E, and F. These nodes are node part of the network (despite the in-link from E to F)