

# Assignment 1

CS 532: Introduction to Web Science

Spring 2016

Manoj Chandra Kompalli

Finished on January 28,2016

# 1

## Question

1. Demonstrate that you know how to use "curl" well enough to correctly POST data to a form. Show that the HTML response that is returned is "correct". That is, the server should take the arguments you POSTed and build a response accordingly. Save the HTML response to a file and then view that file in a browser and take a screen shot.

## Answer

I have started off with getting the response headers of the urls. I then learnt to save the response header into a text file. I researched a bit and found out that the command to post data to a server. I quickly realized that due to security reasons most websites do not use the POST method. I could actually change the method to POST in my local machine if I wanted to. Through the google groups page I found a url <https://www.cs.tut.fi/~jkorpela/forms/testing.html> . The response page shows the value of the input field from the requested page which was on <http://www.cs.tut.fi/cgi-bin/run/~jkorpela/echo.cgi> . Now, I used the curl command to POST the data on to the form using the command

```
curl -data Content=hi_There_Manoj_here_! http://www.cs.tut.fi/cgi-bin/run/~jkorpela/echo.cgi
```

If we want to generate a file and save the response to it directly then,

```
curl --data "Comment= hi_There_Manoj_here_!" http://www.cs.tut.fi/cgi-bin/run/~jkorpela/echo.cgi -o p1.html
```

where p1.html is the output page where the response is saved.

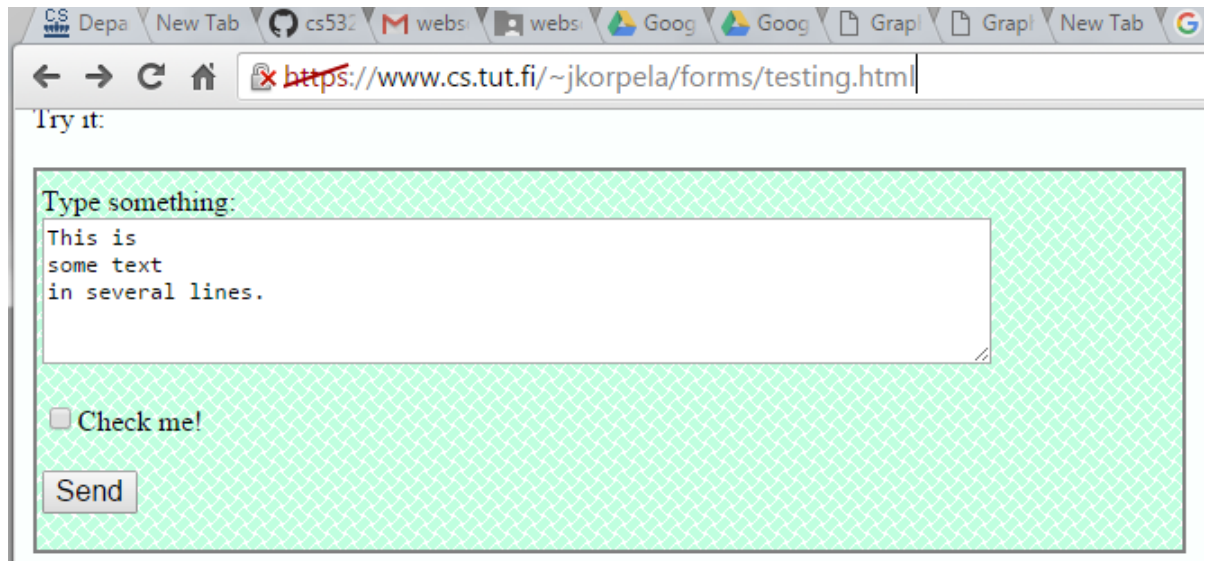


Figure 1: The form at <https://www.cs.tut.fi/~jkorpela/forms/testing.html> where the request is made, rendered in Google Chrome

which renders in the browser like shown in Figure ??.

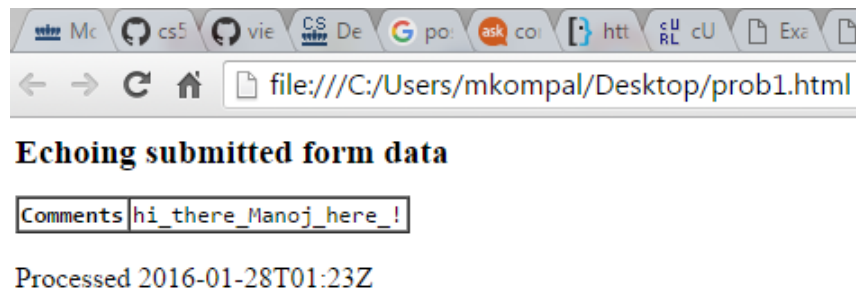
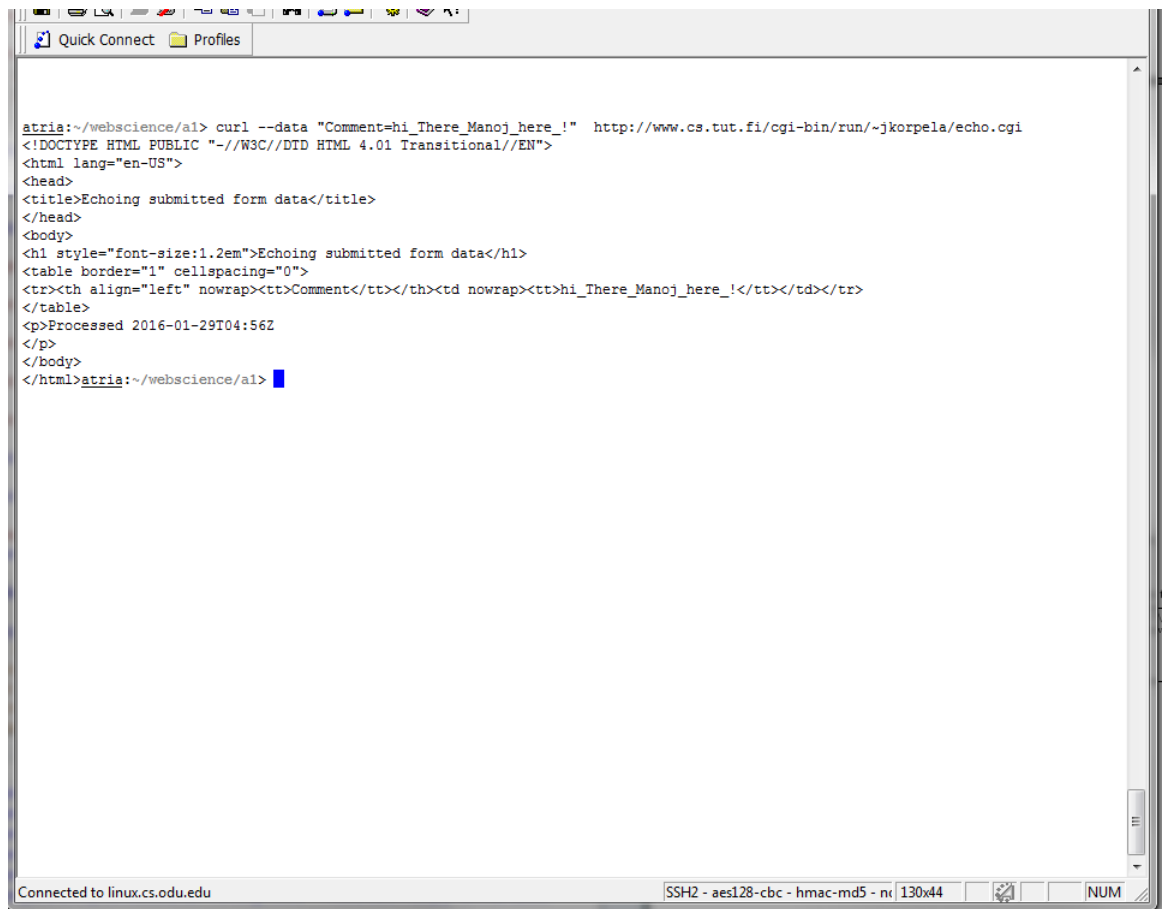


Figure 2: The response at <http://www.cs.tut.fi/cgi-bin/run/~jkorpele/echo.cgi>, rendered in Google Chrome



The image shows a terminal window titled "Quick Connect" with a "Profiles" button. The terminal displays the following text:

```
atria:~/webscience/a1> curl --data "Comment=hi_There_Manoj_here_!" http://www.cs.tut.fi/cgi-bin/run/~jkorpela/echo.cgi
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="en-US">
<head>
<title>Echoing submitted form data</title>
</head>
<body>
<h1 style="font-size:1.2em">Echoing submitted form data</h1>
<table border="1" cellspacing="0">
<tr><th align="left" nowrap><tt>Comment</tt></th><td nowrap><tt>hi_There_Manoj_here_!</tt></td></tr>
</table>
<p>Processed 2016-01-29T04:56Z
</p>
</body>
</html>atria:~/webscience/a1>
```

The terminal window has a status bar at the bottom that reads "Connected to linux.cs.odu.edu" and "SSH2 - aes128-cbc - hmac-md5 - n". There are also icons for a window, a terminal, and a "NUM" button.

Figure 3: Figure showing the response on secure shell CLI

## 2

### Question

2. Write a Python program that:
  1. takes as a command line argument a web page
  2. extracts all the links from the page
  3. lists all the links that result in PDF files, and prints out the bytes for each of the links. (note: be sure to follow all the redirects until the link terminates with a "200 OK".)
  4. show that the program works on 3 different URIs, one of which needs to be:  
`http://www.cs.odu.edu/~mln/teaching/cs532-s16/test/pdfs.html`
3. Consider the "bow-tie" graph in the Broder et al. paper (fig 9):  
`http://www9.org/w9cdrom/160/160.html`

## Answer

I have used the BeautifulSoup library to extract all the links from a given url which is passed as a command line argument. I can use the findAll method which can extract specific html elements from the url. In my case I wanted to extract all the anchor tags. I also used urllib2 library which allows us to read the response header, parse the url and fetch the required content. I have used response.info() to get the response header elements like content type and content length. Now, I can list only the pdf urls by using filtering urls by content type. I found the size of the pdf in bytes using the response header attribute called content-length. Finally I displayed the status using response.getcode(). Below is the python code which extracts urls which have pdf content and displays the size of pdf in bytes.

### 0.1 Code Listing

```
#!/usr/bin/env python

# get_links.py

import re

import sys
import urllib
import urllib2
import urlparse
from BeautifulSoup import BeautifulSoup

class MyOpener(urllib.FancyURLopener):

    def parse(url):#user defined function to parse the url.
        mo = MyOpener()

        page = mo.open(url)

        text = page.read()
        page.close()

        soup = BeautifulSoup(text)#convert the page content into
        soup object
```

```

for link in soup.findAll('a', href=True):#finds all links in
    the url
    link['href'] = urlparse.urljoin(url, link['href'])#parses
    the url and extracts a link into link['href']
    response=urllib2.urlopen(link['href'])#generates a
    response from each individual url

    message=response.info()#contains the response
    information i.e response header
    format=message.type#gives the format if it is plain text
    or pdf etc
    size=response.info().get("Content-length")#gets the size
    of the content in bytes.

    if format == "application/pdf" :#verifies if the url is
        of pdf format
        print link['href']#prints the matching url i.e a pdf
        print "size_is", size, "bytes_and_response_code_is_",
            response.getcode()# prints the size and status
            code

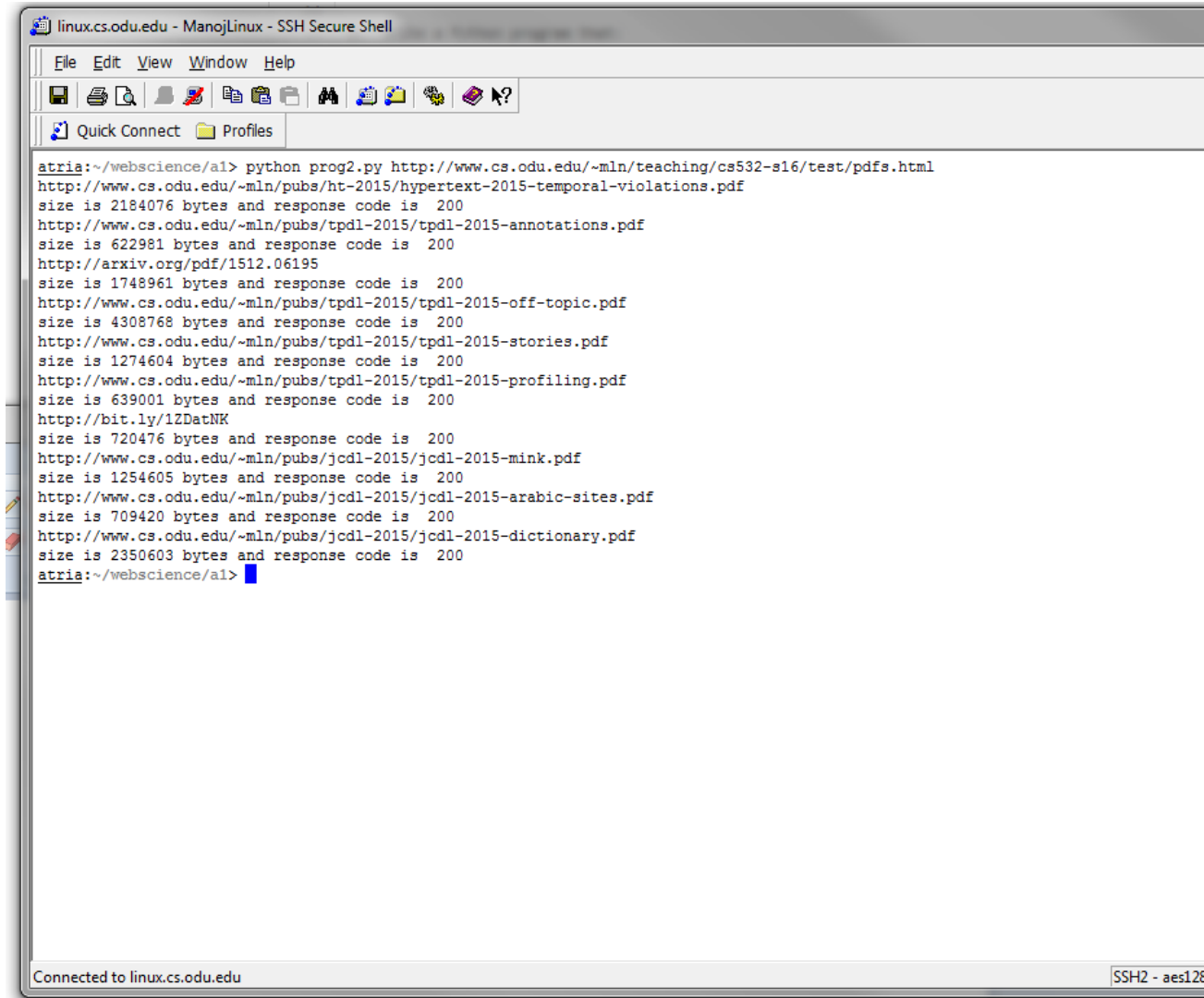
def main():

    for url in sys.argv[1:]:
        parse(url)#calls the parse method
    # main()

if __name__ == "__main__":
    main()

```



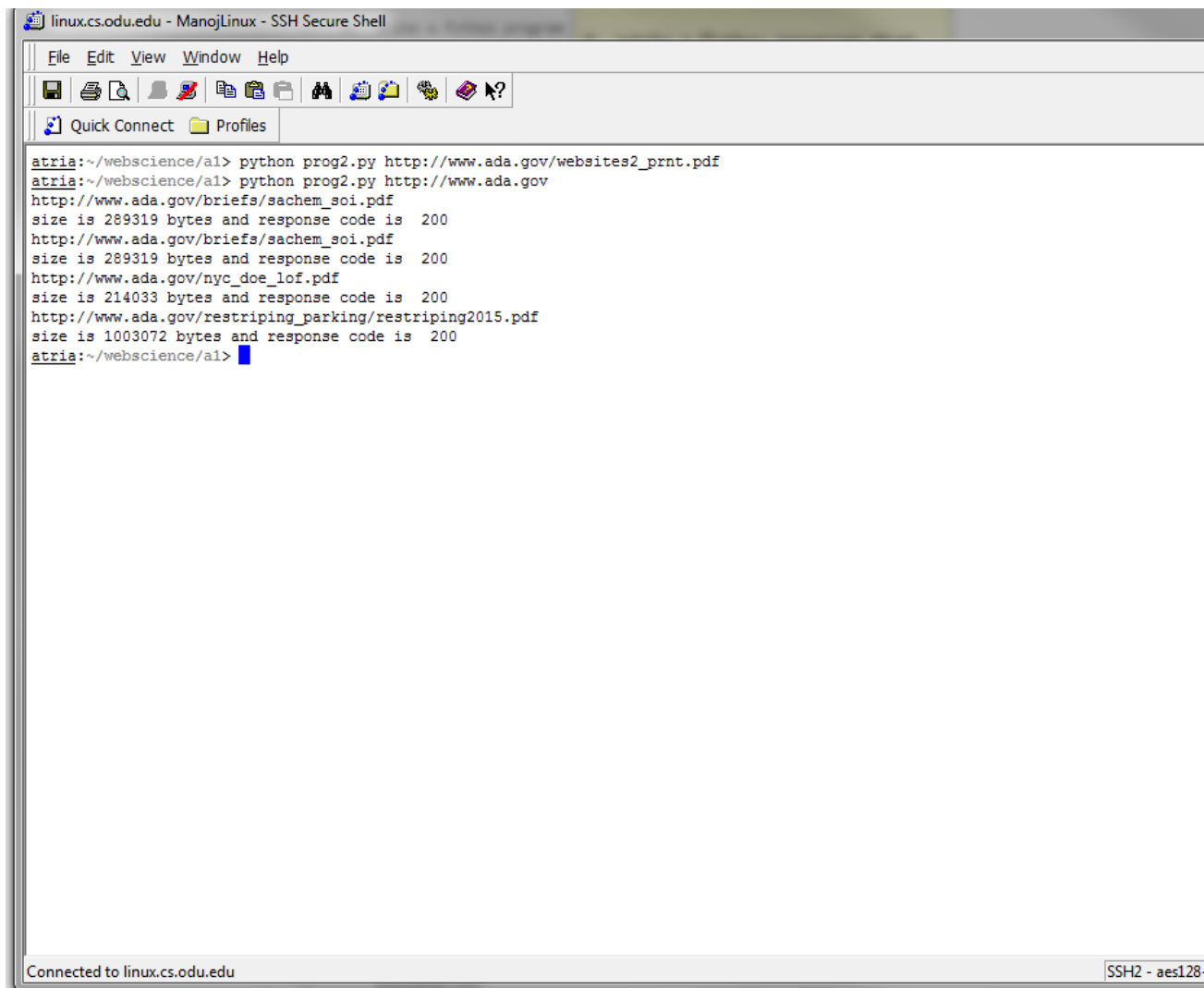


The image shows a terminal window titled "linux.cs.odu.edu - ManojLinux - SSH Secure Shell". The window has a menu bar with "File", "Edit", "View", "Window", and "Help". Below the menu bar is a toolbar with various icons. The terminal content shows a series of HTTP requests and their responses, each followed by the size of the response in bytes and the response code. The requests are as follows:

```
atria:~/webscience/a1> python prog2.py http://www.cs.odu.edu/~mln/teaching/cs532-s16/test/pdfs.html
http://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-violations.pdf
size is 2184076 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf
size is 622981 bytes and response code is 200
http://arxiv.org/pdf/1512.06195
size is 1748961 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf
size is 4308768 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf
size is 1274604 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf
size is 639001 bytes and response code is 200
http://bit.ly/1ZDatNK
size is 720476 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf
size is 1254605 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf
size is 709420 bytes and response code is 200
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf
size is 2350603 bytes and response code is 200
atria:~/webscience/a1>
```

The terminal window also shows a status bar at the bottom with the text "Connected to linux.cs.odu.edu" and "SSH2 - aes128".

Figure 4: Figure showing the response for <http://www9.org/w9cdrom/160/160.html>



The screenshot shows a terminal window titled "linux.cs.odu.edu - ManojLinux - SSH Secure Shell". The terminal has a menu bar with "File", "Edit", "View", "Window", and "Help". Below the menu bar is a toolbar with various icons. The terminal content shows the following commands and output:

```
atria:~/webscience/al> python prog2.py http://www.ada.gov/websites2_prnt.pdf
atria:~/webscience/al> python prog2.py http://www.ada.gov
http://www.ada.gov/briefs/sachem_soi.pdf
size is 289319 bytes and response code is 200
http://www.ada.gov/briefs/sachem_soi.pdf
size is 289319 bytes and response code is 200
http://www.ada.gov/nyc_doe_lof.pdf
size is 214033 bytes and response code is 200
http://www.ada.gov/restriping_parking/restriping2015.pdf
size is 1003072 bytes and response code is 200
atria:~/webscience/al>
```

At the bottom of the terminal window, there is a status bar that says "Connected to linux.cs.odu.edu" on the left and "SSH2 - aes128" on the right.

Figure 5: Figure showing the response for <http://www.ada.gov>

```
linux.cs.odu.edu - ManojLinux - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
atria:~/webscience/a1> python prog2.py http://www.nostredame.info/
http://www.nostredame.info/en/novel.pdf
size is 979273 bytes and response code is 200
http://www.nostredame.info/de/ebook.pdf
size is 1080897 bytes and response code is 200
http://www.nostredame.info/nl/nostradamus.pdf
size is 1252776 bytes and response code is 200
http://www.nostredame.info/sp/novela.pdf
size is 1043674 bytes and response code is 200
http://www.nostredame.info/fr/roman.pdf
size is 1096730 bytes and response code is 200
http://www.nostredame.info/cn/traditionalchinese.pdf
size is 2028555 bytes and response code is 200
http://www.nostredame.info/cn/simplifiedchinese.pdf
size is 1517879 bytes and response code is 200
http://www.nostredame.info/pl/polish.pdf
size is 738227 bytes and response code is 200
http://www.nostredame.info/br/ebook.pdf
size is 202748 bytes and response code is 200
http://www.nostredame.info/jp/japanese.pdf
size is 1005134 bytes and response code is 200
Connected to linux.cs.odu.edu
```

Figure 6: Figure showing the response for <http://www.nostredame.info/>

### 3

#### Question

3. Consider the "bow-tie" graph in the Broder et al. paper (fig 9):  
<http://www9.org/w9cdrom/160/160.html>

Now consider the following graph:

```
A --> B
B --> C
C --> D
C --> A
C --> G
E --> F
G --> C
G --> H
I --> H
I --> J
```

I --> K  
J --> D  
L --> D  
M --> A  
M --> N  
N --> D  
O --> A  
P --> G

For the above graph, give the values for:

IN:  
SCC:  
OUT:  
Tendrils:  
Tubes:  
Disconnected:

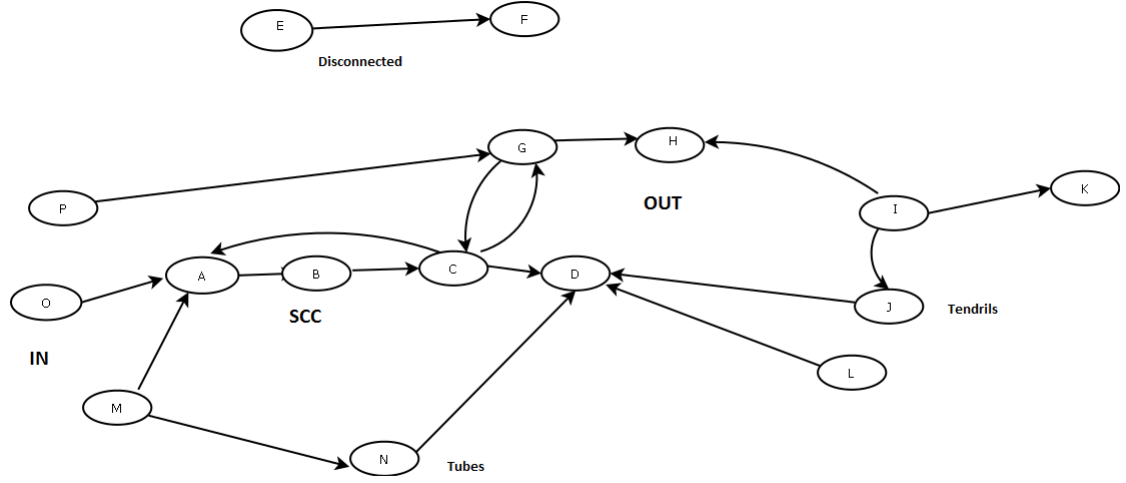


Figure 7: Sketch of the above graph

## Answer

Figure ?? is a graph of the included points, which looks like a bow-tie.

The descriptions of each node is a matter of interpretation. Explanations have been provided for each value assignment.

**IN:**  $M, O, P$

IN is a set of nodes which are directed to the strongly connected nodes in the graph. In the graph we have  $P, M, O$  incident on  $G, A, A$  respectively.  $C$  is directed towards  $A$  but it is in SCC. Therefore  $C$  is not a part of IN.

**SCC:**  $A, B, C, G$

SCC is a set of strongly connected components which are linked to and from both IN and OUT. They are connected to every other node in SCC.  $A, B, C, G$  fall under this category.

**OUT:**  $D, H$

OUT is a set of nodes which are directed from the nodes of SCC. Here we have  $D, H$  in OUT.

**Tendrils:**  $I, J, K, L$

The *Tendrils* are pages that cannot reach the SCC or are not reached from the SCC. The tendrils come from other graphs and only join the whole via  $D, H$ .

**Tubes:**  $N$

*Tubes* pass from IN to OUT without touching SCC.  $N$  is the tube which links from IN to OUT.

**Disconnected:**  $E, F$

$E, F$  are the two nodes which are connected to each other but disconnected from the rest of the graph

## References

- [1] Curl tutorial,  
<http://curl.haxx.se/docs/httpscripting.html> .
- [2] Personal programming blog on python by Hemanth ,  
<http://h3manth.com/content/methods-submit-form-post-using-curl-perl-python-ruby-ly>
- [3] Python adventures wordpress blog ,  
<https://pythonadventures.wordpress.com/2011/03/10/extract-all-links-from-a-web-pag>
- [4] Python documentation , <https://www.python.org/>.
- [5] Broder et al. paper , <http://www9.org/w9cdrom/160/160.htm>.
- [6] Dia tool for drawing charts.Installer , <http://dia-installer.de/download/index.html.en>.