Building a Web Crawler in Python

Frank McCown
Harding University
Spring 2010

Download a Web Page

- urllib2 library
  http://docs.python.org/library/urllib2.html

```
import urllib2
response = urllib2.urlopen('http://python.org/')
html = response.read()

>>> print html.split('\n')[0]
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Specify User-Agent

- Polite crawlers identify themselves with the User-Agent http header

```
import urllib2
request = urllib2.Request('http://python.org/')
request.add_header("User-Agent", "My Python Crawler")
opener = urllib2.build_opener()
response = opener.open(request)
html = response.read()
```

Getting the HTTP headers

- Use `response.info()`

```
response = urllib2.urlopen('http://python.org/')

>>> print response.info()
Date: Fri, 21 Jan 2011 15:56:26 GMT
Server: Apache/2.2.9 (Debian) DAV/2 SVN/1.5.1 mod_ssl/2.2.9
OpenSSL/0.9.8g mod_wsgi/2.5 Python/2.5.2
Last-Modified: Fri, 21 Jan 2011 09:55:39 GMT
ETag: "105800d-4a30-49a5840a1fcc0"
Accept-Ranges: bytes
Content-Length: 18992
Connection: close
Content-Type: text/html
```

Getting the Content-Type

- It’s helpful to know what type of content was returned
- Typically just search for links in html content

```
content_type = response.info().get('Content-Type')

>>> content_type
'text/html'
```

Saving the Response to Disk

- Output html content to myfile.html

```
f = open('myfile.html', 'w')
f.write(html)
f.close()
```

Download BeautifulSoup

- Use BeautifulSoup to easily extract links
- Download BeautifulSoup-3.2.0.tar.gz from http://www.crummy.com/software/BeautifulSoup/download/3.x/
- Extract the file's contents
  - 7-Zip is a free program that works with .tar and .gz files http://www.7-zip.org/

Install BeautifulSoup

- Open a command-line window
  - Start → All Programs → Accessories → Command Prompt
- cd to the extracted files and run setup.py:

```
C:\>cd BeautifulSoup-3.2.0

C:\BeautifulSoup-3.2.0>setup.py install
running install
running build
running build_py
creating build
Etc…
```

Extract Links

- Use BeautifulSoup to extract links

```
from BeautifulSoup import BeautifulSoup
html = urllib2.urlopen('http://python.org/').read()
soup = BeautifulSoup(html)
links = soup('a')

>>> len(links)
94
>>> links[4]
<a href="/about/" title="About The Python Language">About</a>
>>> link = links[4]
>>> link.attrs
[(u'href', u'/about/'), (u'title', u'About The Python Language')]
```

Convert Relative URL to Absolute

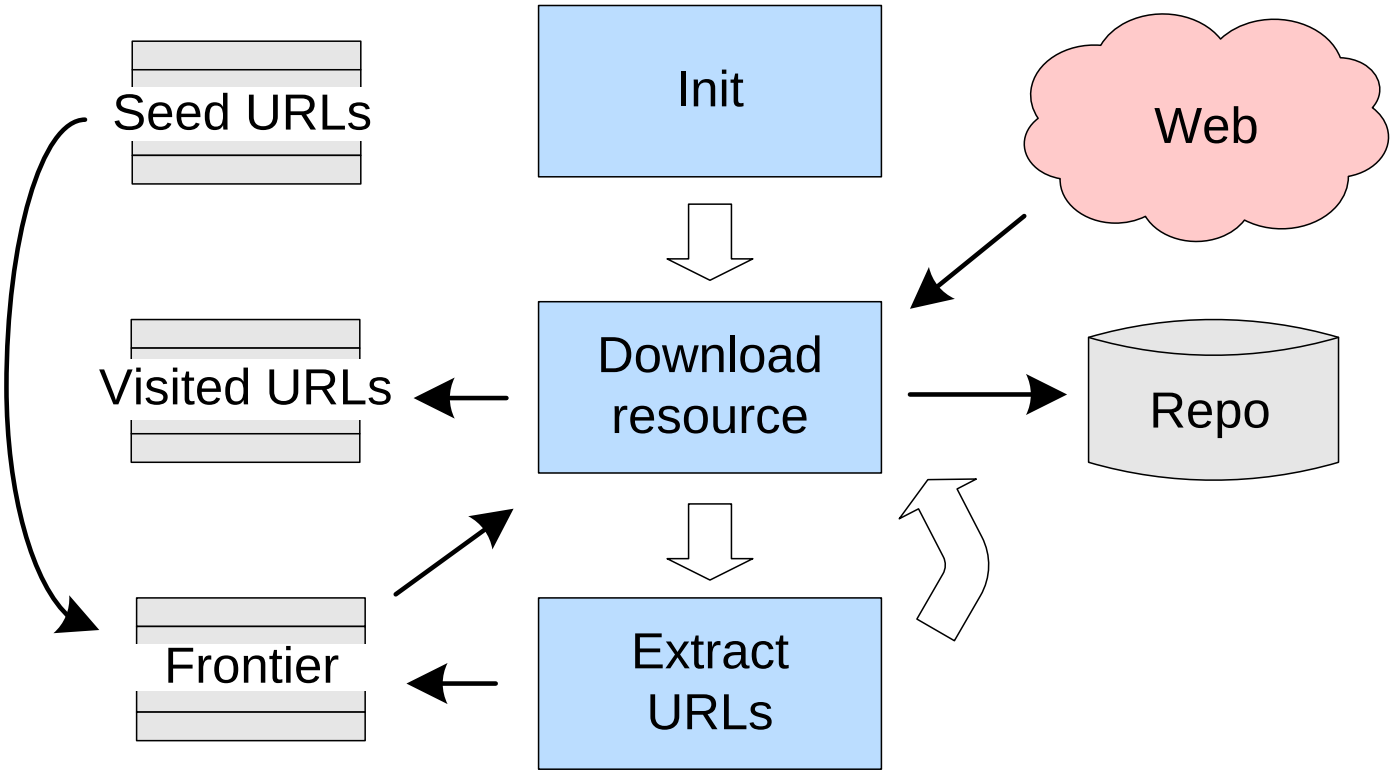
- Links from BeautifulSoup may be relative
- Make absolute using `urljoin()`

```
from urlparse import urljoin

url = urljoin('http://python.org/', 'about.html')
>>> url
u'http://python.org/about/'

url = urljoin('http://python.org/', 'http://foo.com/')
>>> url
u'http://foo.com/'
```

Web Crawler


Seed URLs
Init
Web
Visited URLs
Download resource
Repo
Frontier
Extract URLs

Primary Data Structures

- Frontier
  - Links that have not yet been visited
- Visited
  - Links that have been visited
- Discovered
  - Links that have been discovered

Simple Crawler Pseudocode

```
Place seed urls in Frontier
For each url in Frontier
    Add url to Visited
    Download the url
    Clear Discovered
    For each link in the page:
        If the link has not been Discovered, Visited, or in the Frontier then
            Add link to Discovered
    Add links in Discovered to Frontier
    Pause
```

Simple Python Crawler

```
def crawl(seeds):
    frontier = seeds
    visited_urls = set()

    for crawl_url in frontier:
        print "Crawling:", crawl_url
        visited_urls.add(crawl_url)

        try:
            c = urllib2.urlopen(crawl_url)
        except:
            print "Could not access", crawl_url
            continue

        content_type = c.info().get('Content-Type')
        if not content_type.startswith('text/html'):
            continue

        soup = BeautifulSoup(c.read())
        discovered_urls = set()
        links = soup('a')    # Get all anchor tags
        for link in links:
            if ('href' in dict(link.attrs)):
                url = urljoin(crawl_url, link['href'])
                if (url[0:4] == 'http' and url not in visited_urls
                    and url not in discovered_urls and url not in frontier):
                    discovered_urls.add(url)
        frontier += discovered_urls
        time.sleep(2)
```

Assignment

- Add an optional parameter *limit* with a default of 10 to `crawl()` function which is the maximum number of web pages to download
- Save files to *pages* dir using the MD5 hash of the URL

```
import hashlib
filename = 'pages/'  + hashlib.md5(url).hexdigest() + '.html'
```

- Only crawl URLs that match *.harding.edu
  - Use a regular expression when examining discovered links

```
import re
 p = re.compile('ab*')
 if p.match('abc'):
    print "yes"
```

- Submit working program to Easel