电类工程学导论 C 实验报告 2

518030910406　郑思榕

一、实验准备

1. 实验环境介绍
   1) 环境：在 windows 系统中使用 VirtualBox 5.2.18 安装 Ubuntu14.04 虚拟机，从而在 UNIX 系统环境下进行本次实验。
   2) 语言：使用 python 2.7 获取目标 URL 的网页源代码，从而对目标网页的 html 信息进行数据解析
   3) 工具：本次实验主要调用 python 的库函数和运用正则表达式
      - 库 urllib2：用 urllib2.urlopen(url).read()打开目标 URL 并让 python 可以对其读取，读取后的网页代码为<type 'str'>类型
      - 库 BeautifulSoup：从 bs4 中引用，是本实验最重要的库，通过其中的 BeautifulSoup(content,'html.parser')函数将已读取的网页代码转换为可以由该库处理的格式，处理后网页代码为<class 'bs4.BeautifulSoup'>类型，再用 findAll()函数查找所需信息
      - 库 sys：使用其中的 sys.argv 函数实现从命令行参数中提取需要爬取的网页链接
      - 库 urlparse：使用其中的 urljoin( )函数将相对 URL 连接成完整 URL
      - 库 re 和正则表达式：实现对所需信息的格式的精确描述，提高爬取信息的准确率
      - 库 cookielib：定义了用于自动处理 HTTP cookie 的类。Cookie 是处理网站需要的小块数据。Cookie 需要通过 web 服务器的 HTTP 响应在客户机上设置，然后在稍后的 HTTP 请求中返回给服务器。
2. 实验目的
   1) 使用自己的账号模拟登陆 BBS 后，修改个人说明档（修改 bbs_set_sample.py）（需要先登录，再 POST 修改说明档）
   2) 修改 crawler_sample.py 中的 union_bfs 函数，完成 BFS 搜索
   3) 修改 crawler_sample.py 中的 crawl 函数，返回图的结构。其中，graph 结构与 crawler_sample.py 中 g 的结构相同。
   4) 进一步修改函数，完成网页爬虫（修改 crawler.py）。需要修改的有：
      - 将练习 2,3 中修改的部分加入 crawler.py
      - 修改 get_all_links(content, page)实现输入网页内容 content，网页内容所在的网址 page，以 list 形式返回网页中所有链接
      - 修改 get_page(page) 实现输入网址 page，返回网页内容 content。注意做异常处理（try/except，防止网页无法访问），建议在 urlopen 时加超时参数 timeout。
      - 修改 crawl(seed, method, max_page)，其中 seed 为种子网址，method 为 dfs 或 bfs 搜索方式，max_page 为最多爬取的网页数。
3. 实验原理

   使用 python 脚本模拟 HTTP 请求，使用设置好的 Cookie 信息登录网站。使用广度优先或深度优先的方法在 python 上模拟爬虫的工作流程。

二、实验过程

1. 第一题（ex2-1.py）

   1) 进行获取、存储 cookie 的配置：

      在访问网站时，往往需要从 response 中获取 Cookie 信息和在 request 中添加 Cookie 信息。而在 python 中用库 cookielib 实现相关功能时，需要先用一系列函数实现 cookie 的获取和存储。

其中`cj = cookielib.CookieJar()` 通过构建一个 CookieJar 对象实例来保存 cookie，使用来创建 cookie 处理器对象，使 `opener = urllib2.build_opener(urllib2.HTTPCookieProcessor(cj))` 用来创建 cookie 处理器对象，参数为 CookieJar() 对象，使用 `urllib2.install_opener(opener)` 来将 cookie 加入 opener，使得后面的 urlopen()函数能自动调用 cookie 信息。

2) 编写针对 bbslogin 的 request-body 以登录 bbs:

要想修改 bbs 的个人说明档需要先在 bbs 登录，而登录需要提交账号密码等信息。所以我们先用

```
postdata1 = urllib.urlencode({
    'id': id,
    'pw': pw,
    'submit':'login',
})
```

编写需要的 cookie 信息，其中 id，pw 可由用户从命令行界面输入或者使用代码里默认的 id，pw。接着使用下图代码用 urlopen()打开 bbslogin 就能成功登录 bbs 以便接下来修改个人说明档。

```
req1 = urllib2.Request(url = 'https://bbs.sjtu.edu.cn/bbslogin', data = postdata1)
urllib2.urlopen(req1)
```

3) 编写针对 bbsplan 的 request-body 以修改个人说明档:

登录后用 chrome 打开 https://bbs.sjtu.edu.cn/bbsplan 可以看见网页源码如下，注意红色圆圈里的代码，修改 text 的内容为想要在个人说明档呈现的内容，并将提交方式 type 的值设为 update，就能在 python 里成功修改个人说明档。代码与上一步类似，具体代码如下图：


```
postdata2 = urllib.urlencode({  #编写针对bbsplan的request-body
    'type': 'update',
    'text':text,
})
req2 = urllib2.Request(url = 'https://bbs.sjtu.edu.cn/bbsplan', data = postdata2)
urllib2.urlopen(req2)
```

4) 再次进入 bbsplan 查看是否修改成功:

```
content = urllib2.urlopen('https://bbs.sjtu.edu.cn/bbsplan').read()
soup = BeautifulSoup(content,'html.parser')
```

用 返回 bbs 的个人说明档内容到 content 中，并用 Beautifulsoup()解析为 soup 可以识别的格式。最后用

`print str(soup.find('textarea').string).strip().decode('utf8')`就能打印出个人说明档的内容，其中 strip()函数去除首尾空格，string 是 Unicode 编码格式，需要 decode（'utf8'）转为 utf8 才能输出。

5) 附加功能：函数 main():

作为附加功能，我还加入了 main()函数和 `if __name__ == '__main__' : main()` 以便能直接运行 ex2-1.py 文件。在 main()函数里，我应用 sys 库添加了从命令行界面直接输入账号 id、密码 pw 和文本 text 的功能。具体代码如下：当命令行不输入参数时，默认为我的账号密码，默认文本为'Hello world!'。注意从命令行参数输入的文本可能有空格，所以 3:表示第三个参数以后都是文本内容。且命令行输入为列表格式，需要''.join()函数将列表转为字符串。

```
def main():
    import sys
    id = 'Zsir'
    pw = 'DIANGONGDAO'
    text = 'Hello world!'
    if len(sys.argv) > 1:
        id = sys.argv[1]
        pw = sys.argv[2]
        text = ' '.join(sys.argv[3:])
        print text
    bbs_set(id,pw,text)


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

2. 第二、三题（ex2-23.py）

1) 第二题：

类比已经给出的深度优先搜索 union_dfs(a,b)，我们可以得出广度优先搜索算法 union_bfs(a,b)的实现。注意到每次都从队尾 pop 出一个网址链接进行爬取，所以深度优先搜索用 append()函数将爬到的新连接放在队尾，而广度优先搜索用 insert()函数将爬到得链接放在队首，具体代码如下：。

```
def union_bfs(a,b):
    for e in b:
        if e not in a:
            a.insert(0,e)
```

2) 第三题：

题目要求的输出图结构就是输出一个字典，其中每个键值对得格式为 "父节点：[子节点 1，子节点 2…]"。所以在爬取每一个节点下得所有子节点时，就以"该节点:[该节点下得所有子节点]"的格式加入到字典 graph 当中。即在 crawl(seed, method)函数里的循环语句中加入 graph[page] = content 语句，具体如图：

```
def crawl(seed, method):
    tocrawl = [seed]
    crawled = []
    graph = {}
    while tocrawl:
        page = tocrawl.pop()
        if page not in crawled:
            content = get_page(page)
            outlinks = get_all_links(content)
            graph[page] = outlinks

            globals()['union_%s' % method](tocrawl, outlinks)
            crawled.append(page)
    return graph, crawled
```

结果展示：

```
graph_dfs: {'A': ['B', 'C', 'D'], 'C': [], 'B': ['E', 'F'], 'E': ['I', 'J'], 'D': ['G', 'H'], 'G': ['K', 'L'], 'F': [], 'I': [], 'H': [], 'K
crawled_dfs: ['A', 'D', 'H', 'G', 'L', 'K', 'C', 'B', 'F', 'E', 'J', 'I']
graph_bfs: {'A': ['B', 'C', 'D'], 'C': [], 'B': ['E', 'F'], 'E': ['I', 'J'], 'D': ['G', 'H'], 'G': ['K', 'L'], 'F': [], 'I': [], 'H': [], 'K
crawled_bfs: ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L']
```

3. 第四题（ex2-4.py）

1) 修改 get_all_links(content, page)：

先用`soup = BeautifulSoup(content,'html.parser')`将网页内容转换成可以被 soup 处理的格式，接下来使用正则表达式和 findAll()函数获取所有符合条件的链接标签。注意网页内同时有绝对链接地址和相对链接地址，需要都取出来所以正则表达式为`^http|^/`。并用库 urlparse 里的 urljoin()函数链接成完整地址，最后使用 append()函数将完整地址加入到列表 list。具体代码如下：

```
def get_all_links(content, page):
    links = []
    soup = BeautifulSoup(content,'html.parser')
    for linkline in soup.findAll('a', {'href': re.compile('^http|^/')}):
        rlturl = linkline.get('href')#relative url
        absurl = urlparse.urljoin(page,rlturl)# absolute url
        links.append(str(absurl))
    return links
```

2) 修改 get_page(page)：

该部分一般只需要使用库 urllib2 的 urlopen()函数将链接打开读取即可，但考虑到有可能 HTTP 请求或回应超时，需要添加异常处理机制(try/exception)和超时时间阈值 timeout。具体代码如图：

```
def get_page(page):
    try:
        content = urllib2.urlopen(page,timeout=3).read()
    except Exception as e:
        content = ''
        print ('TIMEOUT:'+str(e))
    return content
```

3) 修改 crawl(seed, method, max_page)：

在该函数中除了添加练习 3 的部分即`graph[page] = content`，还要考虑到最多爬取网页数 max_page 的实现。所以添加计数功能的 count，当 count>=max_page 时就停止循环返回函数。print count 打印当前已爬取的网页数目。代码如下：

```
def crawl(seed, method, max_page):
    tocrawl = [seed]
    crawled = []
    graph = {}
    count = 0

    while tocrawl:
        page = tocrawl.pop()
        if page not in crawled:
            print page
            content = get_page(page)
            add_page_to_folder(page, content)
            outlinks = get_all_links(content, page)
            globals()['union_%s' % method](tocrawl, outlinks)
            crawled.append(page)
            graph[page] = outlinks
            count += 1
            if count >= max_page: break
    return graph, crawled
```

4) 结果展示：

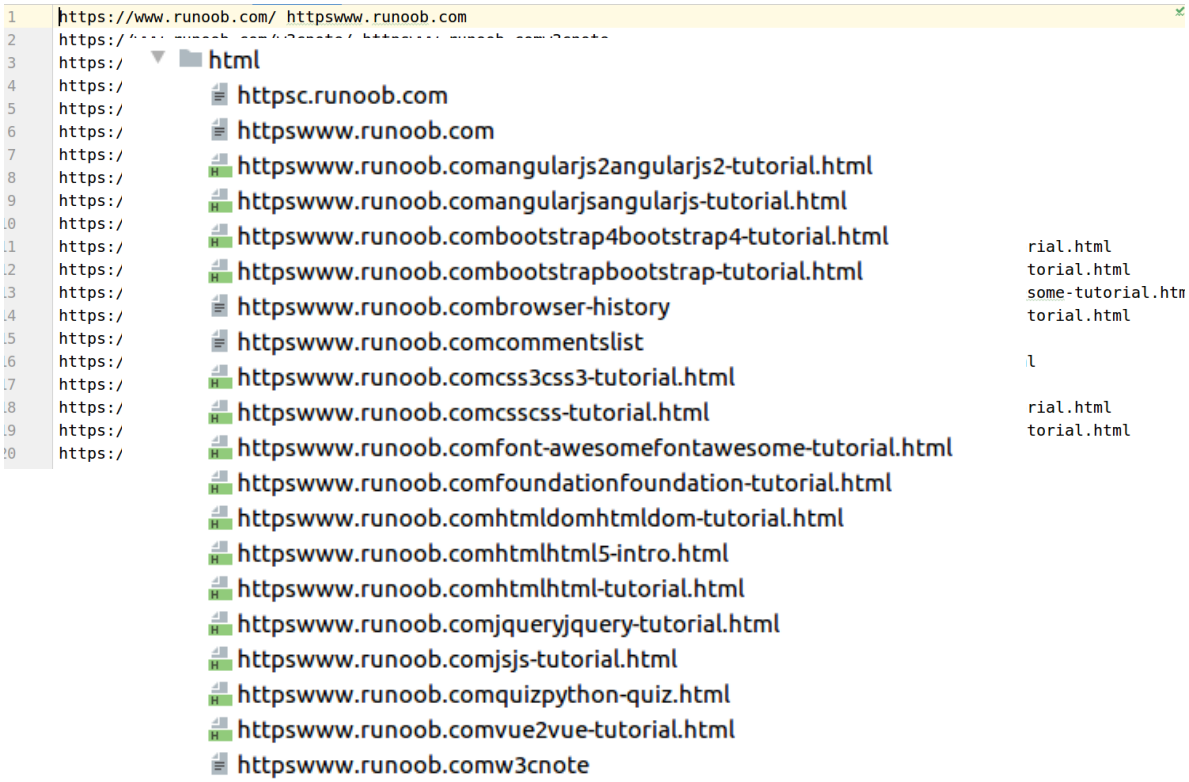
在命令行界面输入如下代码：

```
(base) zsir@zsir-VirtualBox:~$ cd /home/zsir/PycharmProjects/ex2
(base) zsir@zsir-VirtualBox:~/PycharmProjects/ex2$ python ex2-4.py https://www.runoob.com/ bfs 20
```

可以得到命令行窗口返回每次爬取的网页链接：

```
https://www.runoob.com/
https://www.runoob.com/w3cnote/
https://c.runoob.com/
https://www.runoob.com/commentslist
https://www.runoob.com/browser-history
https://www.runoob.com/quiz/python-quiz.html
https://www.runoob.com/html/html-tutorial.html
https://www.runoob.com/html/html5-intro.html
https://www.runoob.com/css/css-tutorial.html
https://www.runoob.com/css3/css3-tutorial.html
https://www.runoob.com/bootstrap/bootstrap-tutorial.html
https://www.runoob.com/bootstrap4/bootstrap4-tutorial.html
https://www.runoob.com/font-awesome/fontawesome-tutorial.html
https://www.runoob.com/foundation/foundation-tutorial.html
https://www.runoob.com/js/js-tutorial.html
https://www.runoob.com/htmldom/htmldom-tutorial.html
https://www.runoob.com/jquery/jquery-tutorial.html
https://www.runoob.com/angularjs/angularjs-tutorial.html
https://www.runoob.com/angularjs2/angularjs2-tutorial.html
https://www.runoob.com/vue2/vue-tutorial.html
(base) zsir@zsir-VirtualBox:~/PycharmProjects/ex2$
```

并且得到 index.txt 文件：



以及 html 文件夹：

三、实验总结

1. 实验概述

练习 1 通过获取存储 cookie 以及修改 html 标签信息来模拟 http 请求和回应，以登录 bbs 并修改其中的个人说明档。练习 2、3 实现了广度优先搜索和深度优先搜索，实现了图节点的遍历。练习 4 整合练习 2、3，将节点替换为网页的 URL 链接，并用网页的分析器抓取网页中的新链接，实现了爬虫的基本工作过程。

2. 实验心得

通过本次实验，我收获颇丰，主要有以下几点：

1) 认识了 HTTP 协议的消息结构，http 的请求 request 和响应 response，request 的两种请求方式 GET 和 POST
2) 学会了 HTML 表单的应用，认识了 Cookie 的含义和应用
3) 学会了如何在 python 模拟 GET 请求、POST 请求、head 头部，如何在 python 环境获取存储 cookie 信息
4) 学会了如何用 python 代码实现广度优先搜索和深度优先搜索算法
5) 了解了如何用 python 实现爬虫的基本工作过程

3. 实验创新点

1) 在 ex2-1.py 中添加了 main()函数以及实现从命令行参数输入 id、pw 和 text，代码如下：

```
def main():
    import sys
    id = 'Zsir'
    pw = 'DIANGONGDAO'
    text = 'Hello world!'
    if len(sys.argv) > 1:
        id = sys.argv[1]
        pw = sys.argv[2]
        text = ' '.join(sys.argv[3:])

    bbs_set(id,pw,text)


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

2) 在 ex2-4.py 中添加超时检测，在 urlopen()函数中添加超时参数 timeout。

最后，衷心感谢实验中老师和各位助教的帮助!