Universe Scientific Publishing
Computer System Networking and Telecommunications
计算机系统网络和电信，2019,1(2)
ISSN:2661-3719 (Print); 2661-3727 (Online)

Python-based Web Crawler

Yuchao WU Zhengde BAO Yawen TANG
School of Computer and Software, Jincheng College, Sichuan University, Chengdu, 611731

Abstract

The web crawler has the ability to automatically obtain useful data according to the rules. In this paper, a targeted crawler system was designed and successfully run by using Python language for the recruitment section page of an IT website. The results show that the crawler program has the advantages of accurate collection results, high efficiency and strong scalability. The use of crawler to collect data can effectively save the time of data collection and improve work efficiency.

Key Words

Python, Web Crawlers, Xpath

DOI:10.18686/jsjxt.v1i2.696

基于 Python 的网络爬虫

巫宇超 鲍正德 唐娅雯
四川大学锦城学院计算机与软件学院，四川成都，611731

摘 要

网络爬虫具有根据规则自动获取有用数据的能力，本文针对某 IT 网站招聘版块页面，使用 Python 语言设计了一个定向爬虫系统并成功运行。运行后的结果表明：本爬虫程序具有采集结果准确、效率高、可扩展性强等优点，使用爬虫采集数据，可以有效节约数据收集的时间，提高工作效率。

关键词

Python；网络爬虫；Xpath

1.引言

随着互联网的迅猛发展，网络信息量的规模以指数级飞速增长，但是由于网站本身的多样化和异构性，很多数据都被嵌入到网页复杂的结构中，如何从如此海量而又复杂的信息中有效地提取出有价值的数据，目前已经成为一个巨大的挑战。[1]

为了能够有效解决上述问题，一种能够针对特定网站高效爬取相关需求的数据的计算机程序逐渐得到了广泛应用，这就是网络爬虫。本文以某网站的招聘板块作为爬取对象，介绍了爬虫系统从分析系统需求到具体实现的思路设计了一个定向爬取的爬虫系统，从互联网抓取数据，进行结构化处理后存储到本地，为以后进一步的数据分析和挖掘提供了基础。

2.网络爬虫

网络爬虫（WebSpider）又叫 WebCrawler，通常简称为爬虫，是一种由计算机语言编程实现，能够自动访问指定的网络地址并获取、清理网页中指定数据的一种脚本程序。[2]

根据数据采集范围的大小不同，爬虫分为通用网络爬虫和定向网络爬虫，通用网络爬虫在网络中最重要的应用就是搜索引擎，它将互联网上所有符合条件的网站都视为爬取对象，将爬取到的丰富信息存储在数据库中。[3]从而支撑整个搜索引擎，如 Google、百度等。定向网络爬虫更有针对性，面向特定的主题，目标是爬取符合特定主题的网页，并将爬取到的内容进行筛选、清理，保证爬取的信息与主题相关，然后提供给用户。

相对人工通过浏览器逐个获取数据，网络爬虫在速

度、准确性、后续操作空间等方面都有明显的优势。随着信息时代数据的作用越来越大，各行各业对于数据的采集和整理的需求也越来越迫切，网络爬虫地位也越来越重要。

3.Python 语言

1989 年，荷兰国家数学和计算机科学研究院的一名程序员 Guido van Rossum 创立了 Python 语言，Python 是一种功能强大的高级计算机编程语言，广泛用于数据挖掘、Web 开发、科学计算等领域，能高效地实现面向对象编程。

Python 语言作为一种热门语言，具有以下特点：

1) 语法简洁清晰、易读易学。

2) 应用广泛，具有丰富的标准库和庞大的第三方库支持。

3) 可移植性强，易于操作各种存储数据的文本文件和数据库。

4) 面向对象，支持开源思想。

Python 具有的这些特点，使得 Python 非常适合用于快速开发应用。在使用 Python 之前，需要先安装和操作系统对应的 Python 版本，并在系统环境变量中配置 Python 所在的路径，然后选择合适的开发工具进行开发。

目前 Python 的版本有 2.X 和 3.X 两种，两个版本的区别主要在语法、编码、模块，目前 Python3 已经成为主流，绝大多数的第三方库也已经支持 Python3，所以建议使用 Python3 进行开发。[4]本文中的爬虫程序是在 Python 3.6 环境下开发调试完成的。

4.爬虫系统需求的分析和设计

4.1 系统需求分析

网络爬虫系统的设计需要解决以下几个问题：

（1） 需要爬取数据的网址的提取。首先初始化网址，然后通过对网页 DOM（Document Object Model，文档对象模型）的分析获取同层级或下一层级的网页地址。

（2）对获取到的具体网页地址进行访问，页面上有我们需要获取的信息，如职位名称、公司名称、薪资等等。Python 系统库中自带 HTTP 库 urllib、urllib2 等，这里使用功能强大的 Python 第三方模块 requests 实现。

（3）对网址进行管理。保证能爬取到每个网页，同时记录已爬取过的网页，防止重复爬取。

（4）目标网页源信息的分析和整理。通过对网页的分析获取信息，进行清洗，去除不必要的部分，将整理好的有用信息保存到本地数据库或其他文件中。对于网页的解析可以通过使用正则表达式（Regular Expression，RE），或者使用 BeautifulSoup、lxml 等第三方库来实现，本文中选择使用 lxml 模块通过 Xpath（XML 路径语言）来解析出我们需要的信息供爬虫下载。

本文将以爬取 CSDN 招聘版块的职位信息为例说明定向爬虫系统的具体实现过程，系统处理逻辑图见图 1。

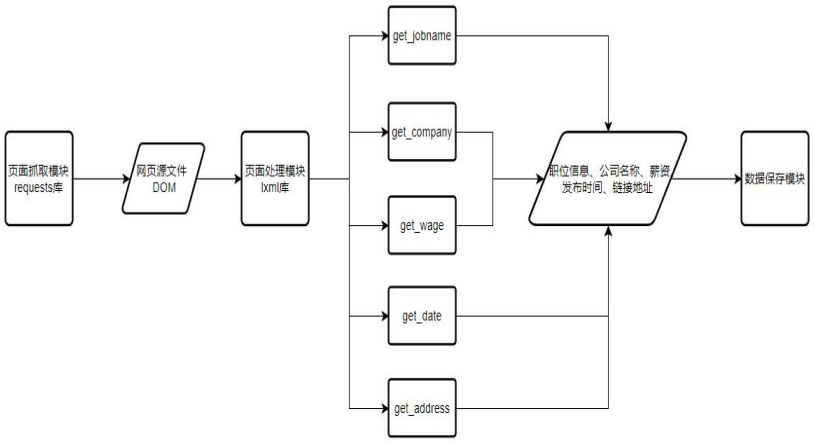

图 1 爬虫处理逻辑图

4.2 爬虫中使用到的模块

Python 具有丰富的系统库，能够快速实现我们想要的功能，而且由于 Python 语言是完全开源的，所以全世界有成千上万的开发者针对 Python 开发出了数量丰富、功能强大的第三方模块，使用这些模块能大大提高我们的开发效率。

1) requests

requests 是目前最流行的 HTTP 库，它采用 Apache2.0 开源协议，使用起来比 urllib 更加高效、方便，开发人员使用 requests 可以轻松地使用 HTTP 请求与网站服务器交互。

2) lxml

lxml 是 Python 的一个第三方解析库，使用 Cython 编写，支持 HTML 和 XML 的解析，支持 XPath 解析方式，而且解析效率非常高。

3) csv

csv 模块是 Python 自带的标准库，提供了基本的 csv 文件读写功能。

5.爬虫功能实现

爬虫系统由三个模块组成，分别是页面抓取模块、页面分析模块、数据保存模块，对应整个爬取工作的三个步骤。[5]

首先在开发环境下安装本文中使用到的 requests 和 lxml 库，并将所有需要使用的库导入到工程中。代码如下（以#开始的为注释行）：

```
#导入程序中所使用到的库
import requests
from lxml import etree
import csv
```

5.1 页面抓取模块

页面抓取模块是爬虫系统的第一个模块，通过模拟浏览器发送 GET 请求给 Web 服务器返回包括数据的页面。

为了保证获取到的页面是正确的，首先要保证我们发送给目标站点的信息是正确而全面的，使站点认为是合法的访问用户。比如有些站点会根据 HTTP 请求的头信息来判断访问请求是否正常，那我们就需要设置爬虫的头信息，在请求中添加 User-Agent、Host、Referer 等必要的信息。

还有一种常见情况是，有些站点提供的信息是针对用户的，那么首先要通过模拟登录后才能获得想要的数据。这种情况下需要将用户的用户名、密码、cookies 等所有登录所需要的信息以 POST 请求发送给站点进行登录，成功登陆后使用返回的 session 和 cookies 与站点进行交互，进行爬虫爬取。本文中由于招聘信息是公开可见的，所以不需要模拟登录。

先使用浏览器浏览目标网页，需要爬取的页面见图 2。使用 requests 的 get 方法向目标网址发送 HTTP 请求，从返回的响应状态码和内容来看，已经成功返回了我们需要的原始数据，下一步进入页面分析阶段。

部分代码如下：

```
#设置请求头
header={‘user-agent’:’ Mozilla/5.0 (Windows NT 10.0; WOW64)’,
‘host’:’job.csdn.net’}
    url="https://job.csdn.net/"
resp=requests.get(url)
print(resp.status_code)
print(resp.text)
```

图 2 浏览器页面显示

5.2 页面分析模块

从页面抓取模块返回的数据是网页的 HTML 代码，包含大量的<…>和</…>，这样的成对结构称为标签（Tag），所有数据都包含在成对的标签当中，那么如何从原始的网页 DOM 源代码中解析出我们需要的职位、公司名称、薪资、发布时间等数据呢？有几种方式可以做到：

（1）正则表达式（Re）。正则表达式匹配是将所有 HTML 代码视为一个字符串，根据设定好的特殊字符及字符组合，通过组合的“规则字符串”进行搜寻和匹配，从而获取用户想要的特定内容。正则表达式使用灵活，功能强大，但是语法相对晦涩、复杂，在待匹配字符串较长的情况下效率较低。

（2）Xpath 方式。Xpath 适用于定位 HTML/XML 文档中节点的技术，它将网页 DOM 视为树形结构，分析节点信息，通过各个节点之间的层次关系解析出数据，效率比较高，这里我们通过第三方模块 lxml 的 Xpath 方式来解析数据。

使用开发者工具确定在页面中的数据所在的位置，对应的标签和 class 值，然后编写 xpath 匹配式进行匹配。如每条职位信息所在的位置都是在一个 class="job_desc job_desc_un_frist"]的 div 标签中，那么我们使用语句 result=tree.xpath('//div[@class="job_desc job_desc_un_frist"]')就能匹配出页面中所有的职位信息，再逐个遍历职位信息，同样使用 Xpath 分别向下提取出相应字段的数据，如从每条职位信息中提取出招聘链接的 Xpath 语句为 xpath("./a")[1].get('href')，经过提取后，最终可以得到一个包含职位、公司、薪资、发布时间、链接这几个字段的嵌套列表，最终传递到数据保存模块进行保存。

部分代码如下：

```
#将页面 HTML 文档建立树形结构
tree=etree.HTML(resp.text)
result= tree.xpath('//div[@class="job_desc job_desc_un_frist"]')
job_info=[["职位名称","招聘公司","薪资","发布时间","招聘链接"]]
for item in result:
    x=item.xpath("./a")
    jobname=x[1].text.strip()
    company=x[2].text.strip()
    wage=x[3].text.strip()
    job_date=x[4].text.strip()
    site='https://job.csdn.net'+x[1].get('href')

job_info.append([jobname,company,wage,job_date,site])
```

5.3 数据保存模块

Python 语言提供了强大的导出功能，可以十分方便地将数据保存到本地或者各种数据库。由于本文中需要爬取的是招聘信息，数据量并不大，所以这里本文选择直接保存在本地 CSV 文件中。需要注意的是，保存数据的时候推荐使用 utf-8 进行编码，避免出现乱码情况。部分代码如下：

```
with open('jobinfo.csv', 'w', newline='',encoding='utf-8') as f:
    writer = csv.writer(f)
    for row in job_info:
        writer.writerow(row)
```

数据保存成功后，用 Excel 打开生成的 csv 文件，显示如图 3 所示，可以看出，和图 2 中浏览器看到的信息是完全一致的。

	A	B	C	D	E
1	职位名称	招聘公司	薪资	发布时间	招聘链接
2	前端工程师	广州西洋汇信息技术有限公司	8-15K/月	2019-03-18更新	https://job.csdn.net/p/159317
3	测试工程师	广州西洋汇信息技术有限公司	7-10K/月	2019-03-18更新	https://job.csdn.net/p/161734
4	高级嵌入式开发工程师	成都正欣德信息技术有限公司	10-20K/月	2019-03-14更新	https://job.csdn.net/p/159339
5	硬件工程师	成都正欣德信息技术有限公司	10-20K/月	2019-03-14更新	https://job.csdn.net/p/159340
6	测试开发实习生 北京 / 实习 /	北京字节跳动网络技术有限公司	5-10K/月	2019-03-12更新	https://job.csdn.net/p/161938
7	C++ 开发实习生 北京 / 实习	北京字节跳动网络技术有限公司	8-12K/月	2019-03-12更新	https://job.csdn.net/p/161937
8	测试实习生 北京 / 实习 / 经验不限...	北京字节跳动网络技术有限公司	5-10K/月	2019-03-12更新	https://job.csdn.net/p/161936
9	前端开发实习生 北京 / 实习 /	北京字节跳动网络技术有限公司	8-15K/月	2019-03-12更新	https://job.csdn.net/p/161935
10	C++开发（高级）工程师 北京 / 社招...	北京字节跳动网络技术有限公司	20-40K/月	2019-03-12更新	https://job.csdn.net/p/161934
11	测试开发工程师 社招	北京字节跳动网络技术有限公司	20-40K/月	2019-03-12更新	https://job.csdn.net/p/161933
12	测试开发（高级）工程师 北京 技术方向...	北京字节跳动网络技术有限公司	20-40K/月	2019-03-12更新	https://job.csdn.net/p/161932
13	前端高级开发工程师 北京 / 社招	北京字节跳动网络技术有限公司	25-50K/月	2019-03-12更新	https://job.csdn.net/p/161931
14	前端开发工程师 北京 / 社招 /	北京字节跳动网络技术有限公司	20-40K/月	2019-03-12更新	https://job.csdn.net/p/161930
15	技术培训生	广州西洋汇信息技术有限公司	3-6K/月	2019-03-12更新	https://job.csdn.net/p/159315

图 3 采集后保存到本地的数据

5.4 改进和完善

在上述代码成功运行之后，为了提高爬虫的效率，我们还可以从网址管理和爬取线程等方面进一步完善爬虫。

由于爬取的网页地址繁多，我们需要对所有网址进行管理，一方面防止遗漏，一方面防止重复爬取甚至产生死循环。网址管理的方法一般有关系数据库管理、缓存数据库管理、内存管理三种，本文采用内存管理的方式。Python 中提供了一种集合（SET）的数据结构，集合的特点是无序并消除重复元素，所以我们使用两个集合分别存储尚未爬取和已经爬取的网址，在爬取过程中实时更新两个集合，这样就实现了网址的管理。

如果爬取的数据量很大，爬虫系统采用单线程逐个爬取的话会导致系统的工作时间大大延长，甚至有可能触发超时导致任务失败，这种情况下我们可以使用多个线程同时对海量目标进行爬取，Python 中提供了 threading 以实现多线程，部分代码如下：

```
import threading
#设置最大线程数
max_threads=10
threads=[]
while threads or topic_list:
    for thread in threads:
        #判断线程是否已经结束，结束的话从线程库移除
        if not thread.is_alive():
            threads.remove(thread)
    #当未爬网址不为空且线程数少于最大线程数时，开启新线程抓取
    while len(threads)<max_threads and topic_list:

thread=threading.Thread(target=process_queue,args=(topic_list,))
thread.setDaemon(True)
thread.start()
threads.append(thread)
```

6.结束语

本文通过使用 Python 语言及一些第三方库，实现了一个简单的网络爬虫系统。在当今的网络和大数据时代，爬虫作为一种采集数据的自动化工具，有着极为广阔的应用。但设计一个好的爬虫系统，一方面要通过多线程、并发等方式提高爬虫采集数据的效率，一方面又要注意不能高频率、大并发地对网站数据进行爬取，使网站服务器承担过多的负荷，如何在这两方面之间取得平衡，是我们在将来的工作中需要进一步进行研究的。

参考文献

[1]魏冬梅,何忠秀,唐建梅.基于 Python 的 Web 信息获取方法研究[J].软件导刊,2018,17(01):41-43.
[2]朱赟.Python 语言的 Web 开发应用[J].电脑知识与技术,2017,13(32):95-96.
[3]贺杰.基于 Webdriver 爬虫技术的研究[J].科技广场,2016(10):27-31.
[4]魏少鹏. 基于 Chrome 浏览器插件的爬虫系统[D].东华大学,2016.
[5]姜琨. 主题搜索引擎中的爬取技术研究[D].国防科学技术大学,2011.

作者简介

第一作者：巫宇超（1998-），男，汉，四川省成都市，本科，四川大学锦城学院，研究方向：大数据技术。
第二作者（通讯作者）：鲍正德（1989-），男，汉，黑龙江哈尔滨，研究生，四川大学锦城学院，研究方向：电子商务。
第三作者：唐娅雯（1999-），女，汉，四川省资阳市，本科，四川大学锦城学院，研究方向：信息管理、J2EE