文章编号：2096-1472(2022)-10-08-04
DOI:10.19644/j.cnki.issn2096-1472.2022.010.002

基于Scrapy-Redis的分布式爬取当当网图书数据

胡学军，李嘉诚

(上海理工大学机械工程系，上海 200082)
✉HXJ20161645@163.com; jiujiuniansheng@163.com

摘　要：单机的网络爬虫爬取数据效率较低，而研究分布式网络爬虫能有效提高数据的爬取效率。文中选择使用上更为简单的Scrapy-Redis框架，设计一个架构模式为主从式的分布式网络爬虫系统，实现对当当网图书信息的爬取；并对布隆过滤器算法进行研究，分析影响其性能的参数，将算法集成到Scrapy-Redis的Scheduler的去重模块中。系统使用一台主机做Master，两台从机做Slave，最终运行1 小时后，抓取图书信息18，000余条。

关键词：网络爬虫；Scrapy框架；Scrapy-Redis框架；布隆过滤器算法
中图分类号：TP391.1　**文献标识码**：A

Distributed Crawling of Dangdang Book Data based on Scrapy-Redis

HU Xuejun, LI Jiacheng

(*School of Mechanical Engineering*, *University of Shanghai for Science and Technology*, *Shanghai* 200082, *China*)
✉HXJ20161645@163.com; jiujiuniansheng@163.com

Abstract: Aiming at the low efficiency of single-machine web crawler, this paper conducts a research on distributed web crawling that can effectively improve the efficiency of data crawling. This paper proposes to use the simpler Scrapy-Redis framework and design a distributed web crawler system with master-slave architecture mode to realize the book information crawling of Dangdang. In addition, Bloom Filter algorithm is studied, and the parameters affecting its performance are analyzed. The algorithm is integrated into the deduplication module of Scrapy-Redis Scheduler. With one host computer as Master and two slave ones as Slaves, the system captures more than 18,000 pieces of book information after running for one hour.

Keywords: web crawler; Scrapy framework; Scrapy-Redis framework; Bloom Filter algorithm

1　引言(Introduction)

随着互联网技术的蓬勃发展，变化之一就是网络数据的增长曲线就像指数函数图像一般，而网络爬虫是获取数据的手段之一。初期单机模式的网络爬虫技术将需要抓取的链接组成一个队列，再循环逐个地爬取，直至结束。

分布式爬虫技术就等同于同时使用多台单机网络爬虫抓取和处理数据，抓取相同页面信息时，显著地缩短了时间。随着爬虫手段的逐步优化和一次次技术上的迭代，许多大型的互联网公司(如百度、谷歌等)都研发出了自己的复杂的分布式网络爬虫[1]，但是这一类资源是未开放的。目前公开可使用的爬虫资源，如Scrapy，就是典型的单机模式，此外还有很多类似的爬虫技术。当然也存在一些爬虫技术采用的是分布式模式，如Nutch、Igloo等，但是这类系统在实现的过程中是比较复杂的。

本文研究基于Scrapy-Redis框架，设计和实现一个针对当当网图书信息抓取的分布式系统。

2 基础理论(Basic theory)

2.1 Scrapy框架组件和工作原理

在网络爬虫开发中，Python是使用最广泛的设计语言[2]，Scrapy则是使用最为简单和方便的一种。Scrapy框架是基于Twisted开发的，而Twisted是一款使用Python语言实现的基于事件驱动网络引擎框架[3]。相较于其他网络爬虫技术，该框架的优势在于它是已经封装好的，同时能够执行多个任务，且抓取网站内容时速度很快，也能将抓取内容下载到本地，方便使用者使用。

Scrapy框架主要的组件就是作为核心的爬虫引擎(Scrapy Engine)、负责路由(URL)调度处理的调度器(Scheduler)、在互联网上下载所需要的数据时使用的下载器(Downloader)，以及处理需求定义、页面解析等使用到的爬虫(Spider)组件；项目管道(Item Pipeline)组件作为下载下来的数据的加工厂，可以对其进行清除、存储和验证等操作；两个中间件(下载中间件和爬虫中间件)是连接组件的纽带。

Scrapy工作流程如图1所示，URL和下载数据在各组件之间的流走如图中的箭头方向所示(图中编号1—10为数据走向)。

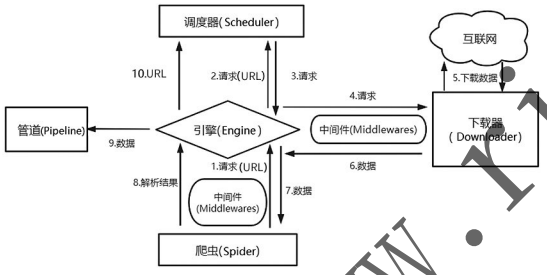

图1 Scrapy工作流程图

Fig.1 Workflow chart of Scrapy

Scrapy的工作原理可以概括如下：爬虫先根据初始的需下载的网页URL到互联网上下载该网页数据；在整个下载数据过程中，新的要下载的网页URL会持续加入Scheduler中的等待下载队列里；爬虫按照下载队列的顺序一次抓取网页，直到下载队列为空[4]。

2.2 网络爬虫的分布式架构技术

(1)Scrapy-Redis框架

Scrapy框架并不支持抓取URL在各个爬虫端之间共享，也就不支持分布式操作。在Scrapy单机爬虫系统中存在一个由deque模块实现的本地爬取队列Queue[5]。这个队列是依靠Scheduler组件负责维护的，是独一无二的，并且它具有不被多个Scheduler共享的特性，造成了Scrapy框架不能实现分布式抓取数据。但是Scrapy框架具有易扩展的特点，通过Scrapy-Redis组件引入Redis数据库代替Scrapy框架的单机内存来存储待抓取的URL，Redis数据库自身又有能共享数据库给多个爬虫端的特性，这样就把抓取队列共享出去了。

Scrapy-Redis架构图如图2所示，它是Scrapy框架和Redis数据库的结合。

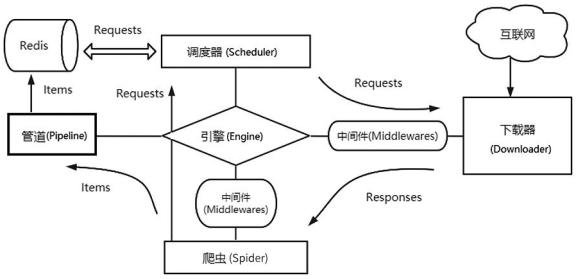

图2 Scrapy-Redis架构图

Fig.2 Scrapy-Redis architecture diagram

(2)Scrapy-Redis框架爬虫的主流模式

分布式爬虫系统架构复杂多变[6]，目前公认的两种主要分类形式：主从模式和对等模式。

对等模式字面上就可以理解，系统里每一台计算机都是平等的，它们都要参与爬虫工作，各自完成自己的任务，互不影响。每台计算机的任务通过哈希(Hash)函数计算分配。

主从模式是由一台主机和若干从机组成的，如图3所示，我们把主机叫作Master节点，把从机叫作Slave节点。主从模式把Redis数据库搭建在Master上，请求的判重、分配和数据的存储都交给Master，不执行抓取任务；Slave负责运行爬虫程序，并把新的请求发送给Master。本文采用的是主从模式。

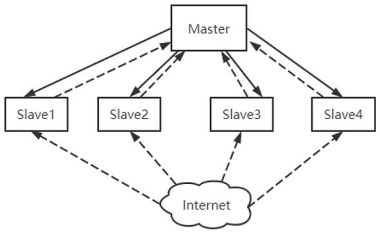

图3 主从分布式策略

Fig.3 Master-Slave distributed strategy

3 布隆过滤器算法(Bloom Filter algorithm)

3.1 基本布隆过滤器原理

巴顿·布隆于1970 年提出了布隆过滤器算法[7]，它可以描述为把所有元素存到一个集合里，然后去比较判断某一元素是否存在，其实现是依靠Hash函数能够把一个大的数据集映射到一个小的数据集上。它的优势就是让大规模的数据得到压缩，减少了存储所消耗的空间，同时查询时间也比其他算法短。

(1)算法描述

第一步是位数组的定义。布隆过滤器是将数据保存在位数组里，所以首先是位数组定义，然后设位数组长度为m，各

个位置初始值为0。

第二步是添加元素。假设独立的Hash函数个数是k，布隆过滤器算法把想要添加的元素用Hash函数计算k次，得到对应于位数组上的不同位置，将多个不相同的位置设为1，如图4所示。

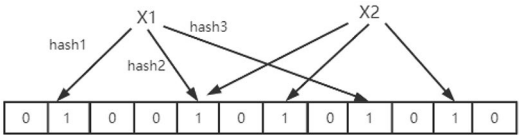

图4 添加元素

Fig.4 Adding elements

第三步是查询元素。将要查询的元素执行和第二步一样的操作，通过Hash函数进行映射，也就是计算k次，把得到的值对应地放到位数组上。假设这些位置上出现一个0，则可以判断该元素肯定不在我们要查询的集合中；如果这些位置全为1，则不能判断是否在集合中，因为布隆过滤器存在误判率。

(2)性能分析

布隆过滤器算法使用k个相互独立的Hash函数[8]，逐一地把原始数据集合n中的元素映射到数组中。查阅相关文献了解到，如果存在一个元素S，其被错误判断为属于该集合的概率如式(1)所示。

$$p_{\mathrm{err}}=[1-(1-\frac{1}{m})^{kn}]^k\approx(1-\mathrm{e}^{\frac{-kn}{m}})^k \tag{1}$$

由式(1)可以看出，影响该值大小的三个因素：Hash函数个数k、位数组长度m和原始数据集合n。当原始数据不断增加时，位数组长度m也要和其同步线性增长，才能让误判率保持稳定。而Hash函数个数k的最优解如式(2)所示。

$$k_{\mathrm{num}}=\frac{m}{n}\ln 2\approx 0.7\frac{m}{n} \tag{2}$$

式(2)中的k_{num}表示在位数组长度和原始数据个数确定的条件下，误判率达到最低的k对应的值。由式(2)可以推导出误判率最低为

$$p_{\mathrm{err}}=(1-\frac{1}{2})^k=2^{-\ln\frac{2m}{n}} \tag{3}$$

由式(3)可以推论出位数组长度应大于等于$-\frac{n\ln p_{\mathrm{err}}}{(\ln 2)^2}$。

3.2 布隆过滤器算法实现

布隆过滤器算法的代码实现步骤如下所述。

第一步是初始化位数组，定义一个Init(self, size, HashCount)方法，其中size表示位数组长度，通过导入bitarray对象可以把二进制串转化为bitarray对象，然后其又能转化为bytes，实现字符串和二进制之间的转化；HashCount表示Hash函数的个数。采用集合(set)形式来表示算法对位数组的初始化。

第二步是对查询元素进行哈希计算，采用的是不同的Hash函数。先导入Murmurhash3，然后把该元素计算k次，得到对应的哈希值。

第三步是验证第二步中要查询的元素是否存在，利用集合查找方法在位数组中进行查找操作。如果对应的位置上出现0，则认为要查询的元素不在集合中，执行第四步。

第四步是把URL添加进集合，定义一个add(self，item)方法实现该目标。

4 系统设计和实现(System design and implementation)

4.1 系统流程图

对当当网的网站结构进行分析，不难发现页面之间存在“父子”关系。我们把网页按不同层次进行处理，把页面内所有的其他链接归类成它的二级页面链接。从首页开始，逐级对剩下的页面进行搜索，把符合规则的链接加入待抓取队列中。系统基本流程图如图5所示。

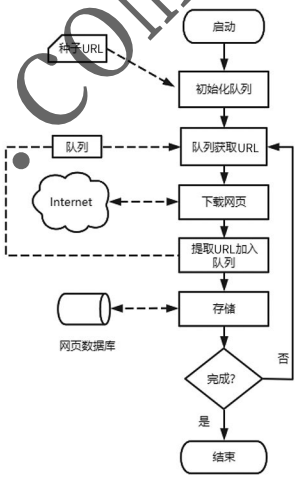

图5 系统基本流程图

Fig.5 Basic flow chat of the system

4.2 数据库设计

(1)数据表设计

Scrapy爬虫下载数据后，通过Item_Loader机制把解析的数据存放到爬虫模型Item中，最后使用项目管道模块存储数据。如表1所示为数据字典设计。

表1 数据字典设计

Tab.1 Data dictionary design

字段名	类型	说明
Name	String	图书的名字
Source	Keyword	图书封面的地址
Price	String	图书的价格
Author	String	图书的作者
Description	String	图书的简介

(2)数据存储

设计完数据字典以后，采用MySQL数据库来存储爬取到的数据，在Pipeline.py文件中编写Scrapy框架和数据库链接

的代码及操作数据库的SQL语句。实现流程如下所述。

第一步：打开Scrapy和MySQL数据库的链接，使用的是open_spider(self，spider)方法，再赋值存储数据服务器的地址(host)、MySQL的端口号(port)、MySQL的用户名(user)、MySQL密码(password)、自定义数据库的名称(name)和字符集(utf8)。

第二步：链接Scrapy和MySQL数据库，引入第三方模块pymysql实现。

第三步：操作数据库，定义一个名为process_item(self，item，spider)的方法，通过Insert into语句来实现数据的存入。

第四步：关闭链接。

4.3 页面解析

Scrapy框架支持多种页面解析工具，本文使用XPath对抓取页面进行解析，主要代码如下所述。

(1)获取所有的selector对象

response.xpath('//ul[@id="component_59"]/li')

(2)图书封面链接Source

xpath(".//img/@src")

(3)图书的名字Name

xpath(".//img/@alt")

(4)图书的价格Price

xpath('.//p[@class="price"]/span[1]/text()')

(5)图书的作者Author

xpath('.//p[@class="search_book_author"]/span[1]/a[1]/text()')

(6)图书的简介Description

xpath(".//p[2]/text()")

4.4 系统测试和运行结果

在测试时，使用一台主机做Master，两台从机做Slave。由Master主机负责URL的调度队列和协调从机之间的抓取任务，Slave从机负责执行抓取程序，到互联网上下载页面数据并存入数据库。以当当网图书为例，运行1 小时后，抓取图书信息18,000余条，主要为图书的名称、作者、价格、封面和简介信息，如图6所示。

id	name	src	author	describtion	price
1	他在云之南（人气作家最行燃情悬爱力作！随书附赠"追光"书签//img3m2.ddimg.cn/71/20/28541672-1_b_6.jpg		最行	人气作家最行燃情悬爱	¥19.90
2	那个不为人知的故事	//img3m2.ddimg.cn/46/8/29124262-1_b_4.jpg	Twentine	★超人气作者Twentin	¥18.90
3	他和她的猫（随机发货，当当专享定制甜蜜3D变幻卡，"唧唧歪//img3m9.ddimg.cn/5/13/25299059-1_b_32.jpg		唧唧的猫	＊你当我男朋友好不好	¥16.50
4	他笑时风华正茂	//img3m6.ddimg.cn/48/0/25216536-1_b_7.jpg	舒远	年少时孟盛精喜欢过一	¥14.70
5	他从火光中走来（人气作者耳东兔子燃情暖心力作，极力呈现//img3m1.ddimg.cn/52/24/29130901-1_b_5.jpg		耳东兔子	★人气作者耳东兔子燃	¥28.00
6	针尖麦（辛夷坞获选五年全新现言力作，随书附赠gua老师亲//img3m4.ddimg.cn/16/27/29332924-1_b_2.jpg		辛夷坞	★辛夷坞获选五年，精	¥34.90
7	你是迟来的欢喜（姬昊《暗恋·橘生淮南》的青春心动故事，随//img3m1.ddimg.cn/74/33/27856001-1_b_7.jpg		顾了之	★新锐暖萌系作家顾了	¥19.90
8	嘿，小家伙	//img3m9.ddimg.cn/8/16/25185509-1_b_15.jpg	温酒	世界已经够冷了，总得	¥18.90
9	提灯照河山（准上作品，当当专享人物透卡）	//img3m2.ddimg.cn/97/31/29170942-1_b_13.jpg	准上	★《破云》《提灯映残	¥22.20
10	寒鸦	//img3m2.ddimg.cn/66/10/29162892-1_b_3.jpg	弱水千流	有个不懂事的年球小子	¥28.30
11	你给的甜（全二册）（是辰日月，朝辉暮光，都不如你给的甜，//img3m7.ddimg.cn/15/7/29286987-1_b_7.jpg		风晓程琴	★小可爱创业策划师X	¥34.90
12	谢谢你出现，这一生我很喜欢 里尔	//img3m7.ddimg.cn/29/7/28469657-1_b_8.jpg	里尔	畅销书作者里尔爆红值	¥23.40
13	愿好春光（云拿月暖心治愈之作）	//img3m7.ddimg.cn/13/20/29287777-1_b_11.jpg	云拿月	★人气作家云拿月暖心	¥20.80
14	我不喜欢这世界，我只喜欢你修订版 乔一 畅销百万册！由吴倩//img3m6.ddimg.cn/64/10/27859456-1_b_6.jpg		乔一	畅销400万册 至臻纪念	¥27.40
15	我见极闪（随书附赠"我喜欢你"书签，惨动青春代言人云拿月，//img3m3.ddimg.cn/82/28/28969363-1_b_9.jpg		云拿月	None	¥19.90
16	你努力去爱的样子，又孤独又美好	//img3m1.ddimg.cn/98/12/25340831-1_b_20.jpg	王潘	治愈系作家王潘情感级	¥30.40
17	山河万里，我来见你	//img3m5.ddimg.cn/49/21/28541155-1_b_3.jpg	咪茶丸子	苏咪天的兵哥哥万里追	¥16.60
18	苍兰诀（九篇非雪亲笔签名版随机发货 当当专享人气画手胭脂//img3m5.ddimg.cn/60/22/29171895-1_b_19.jpg		九篇非雪	★人气作家九篇非雪经	¥17.90

图6 存储在MySQL中的图书相关信息

Fig.6 Book information stored in MySQL

5 结论(Conclusion)

互联网发展带来网络数据的大规模增长，网络爬虫也需要变得更加迅速、高效，才能达到使用要求，所以对分布式爬虫的研究具有一定的实用价值。本文从介绍Scrapy框架的组件及其功能入手，讲述了数据在组件之间流动的过程，分析了Scrapy-Redis组件实现分布式抓取数据的原理和主流模式。以抓取当当网图书信息为例，研究了爬行策略、爬虫设计、解析规则和布隆过滤器算法对抓取URL去重等的设计思路，构建了针对抓取当当网图书信息的主从式分布网络爬虫系统。

但是项目在实际运行过程中出现了令人不满意的问题：(1)Master主机对爬虫任务的管理并不理想；(2)布隆过滤器算法相比其他一般算法，在空间利用上更充分，查询时间也更短，但是误判率也是其非常明显的缺点，后续针对布隆过滤器算法URL去重方法，还需要进一步学习。

参考文献(References)

[1] 罗娇敏,耿茜.一种基于Redis的分布式爬虫系统设计与实现[J].软件,2017,38(10):83-87.

[2] 严慧,彭绪富,朱小婉,等.基于Scrapy-Redis分布式数据采集平台的设计与实现[J].湖北师范大学学报(自然科学版),2019,39(01):19-25.

[3] 汪兵.基于Scrapy框架的分布式爬虫系统设计与实现[D].合肥:合肥工业大学,2019.

[4] 马联帅.基于Scrapy的分布式网络新闻抓取系统设计与实现[D].西安:西安电子科技大学,2015.

[5] 王乐乐.基于Scrapy-Redis的分布式农业网络数据采集平台设计与实现[D].广州:华南农业大学,2019.

[6] 樊宇豪.基于Scrapy的分布式网络爬虫系统设计与实现[D].成都:电子科技大学,2018.

[7] QIAO Y, LI T, CHEN S. One memory access bloom filters and their generalizeation[J]. Proceedings-IEEE INFOCOM, 2011, 28(6):1745-1753.

[8] 步相庆.基于Scrapy-redis和GMM的搜索系统设计与实现[D].成都:西南交通大学,2019.

作者简介：

胡学军(1997-)，男，硕士生.研究领域：智能制造.

李嘉诚(1999-)，男，硕士生.研究领域：CAD图形学.