分布式爬虫的研究与实现

马 蕾,冯锡炜,窦予梓,高天铸,朱 睿,吴衍兵
(辽宁石油化工大学 计算机与通信工程学院,辽宁 抚顺 113001)

摘 要:网络中的数据蕴藏着大量有价值信息,在实际的项目需求中,为了实现能够自动的在网页上对大量数据的数据信息的收集、解析、格式化存储的过程,提出了基于分布式的网络爬虫技术。采用 Nutch 爬虫框架和 Zookeeper 分布式协调服务,配合高性能的 Key-Value 数据库 Redis 对数据进行存储,采用 Solr 引擎将抓取信息进行清晰地索引、展示。运用提取页面信息算法优化提取页面信息流程,通过关键词匹配优化算法根据指标从抓取的数据中获取指标相关数据。通过分布式集群的搭建,Nutch 项目的实现,及大量数据的采集,验证了基于 Nutch 的分布式网络爬虫的可行性。通过页面解析流程实验分析,基于 Nutch 的分布式爬虫与其他爬虫多组实验数据对比结果表明,基于 Nutch 的分布式爬虫项目在性能和准确度方面都优于传统其他爬虫。

关键词:分布式集群;Nutch;Solr;企业官网
中图分类号:TP391 **文献标识码**:A **文章编号**:1673-629X(2020)02-0192-05
doi:10.3969/j.issn.1673-629X.2020.02.037

Research and Realization of Distributed Crawler Based on Nutch

MA Lei,FENG Xi-wei,DOU YU-zi,GAO Tian-zhu,ZHU Rui,WU Yan-bing
(School of Computer and Communication Engineering,Liaoning Shihua University,Fushun 113001,China)

Abstract:The data in the network contains a lot of valuable information. In the actual project requirements,in order to realize the process of automatically collecting,parsing and formatting the data information of a large amount of data on the webpage,a distributed web crawler technology is proposed. The Nutch crawler framework and the Zookeeper distributed coordination service are used to store data in conjunction with the high-performance Key-Value database Redis. The Solr engine is used to clearly index and display the captured information. The extraction page information algorithm is used to optimize the process of extracting page information,and the keyword matching optimization algorithm is used to obtain the indicator related data from the captured data according to the index. Through the construction of distributed clusters,the implementation of the Nutch project,and the collection of large amounts of data,the feasibility of Nutch-based distributed web crawlers is verified. Through the analysis of the page analysis process,the experimental data comparison between the Nutch-based distributed crawler and other reptiles proves that the Nutch-based distributed crawler project is superior to other traditional crawlers in performance and accuracy.

Key words:distributed cluster;Nutch;Solr;enterprise's official website

0 引 言

网络爬虫,是能够按照程序设计者所指定的要求,有序、自动地获取指定网站中有用信息的程序。根据性能和所用技术的不同,爬虫的种类可以分为很多种[1]。

传统的单机爬虫,只有一个单一的本地爬取队列,内存空间小,计算空间小,在需要爬取海量数据的情况下,已经不能满足实际需求。因此需要找出方法,解决这些问题,优化爬取方式,改进解析规则,使爬取出的数据更精准、爬取过程更快速[2]。

将分布式的思想引用到网络爬虫中,分布式文件系统和大型分布式并行计算框架的使用,提高了爬虫在速度和准确度上的性能。通过搭建分布式集群,扩充了工作资源。通过设计爬取流程合理分配空间、对抓取队列进行管理、优化爬取规则、扩充算法,使分布式爬虫的抓取过程更流畅合理[3]。

文中所研究的基于 Nutch 的分布式爬虫就适用于针对大批量数据的操作,其可编写插件的机制利于爬

收稿日期:2019-03-15 **修回日期**:2019-07-16 **网络出版时间**:2019-11-07
基金项目:辽宁省自然科学基金(20180550130)
作者简介:马 蕾(1994-),女,硕士研究生,研究方向为网络爬虫与数据挖掘;冯锡炜,博士,教授,研究方向为分布式计算。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20191107.0912.048.html

虫的模块化和可扩展化。Nutch 提供了可扩展接口,用于扩展爬虫功能,编写不同的插件可实现不同的操作,根据不同的需求可实现自定义功能。开源的全文搜索框架 Solr 直接搜索 Nutch 获取的页面信息,为爬取下来的页面维护一个索引,也可对抓取结果进行复杂条件查询、模糊查询。在爬取时,可以指定数据源获取信息,使抓取更有针对性和目的性[4-7]。

1　分布式集群的搭建

Nutch 依赖于 Hadoop,在 Hadoop 集群中运行可以用于对爬取任务的批处理。利用 HDFS 分布式文件系统对所有的数据进行访问与管理,并将 MapReduce 大型分布式并行计算架构与 HDFS 无缝结合。在分布式集群中,利用 Zookeeper 分布式协调服务,对分布式各个服务提供辅助功能,保证集群高效可用[8-9]。

在实际项目中采用 Redis 这种高性能的 Key-Value 数据库对数进行存储,运用 Solr 分布式高性能全文搜索引擎,支持对文件、文本等数据源建立索引,提供毫秒级查询。选择用 Squid 高性能代理缓存服务器,它支持 FTP、gopher、HTTPS 和 HTTP 协议,这样能更好地接受来自需要下载的目标的请求,并且处理这些请求,获取到信息后,Squid 会复制一份,以便下次使用[10]。

2　项目的爬取原理和爬取流程

Nutch 作为一种透明,开源,基于 Java 语言的搜索引擎框架,提供了非常全面的工具,全文搜索和 Web 爬虫[11]。

2.1　项目框架

设计的爬虫工作流程如图 1 所示,总体分为以下几个步骤:

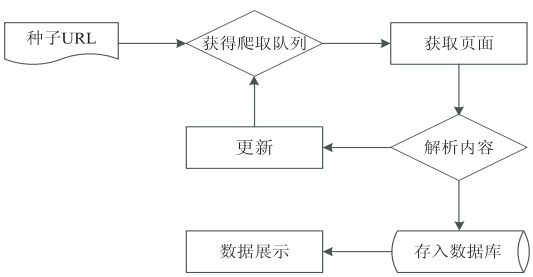

图 1　爬虫工作流程

(1)获取到网页的内容,即下载网页。需要通过向目标服务器发送请求并等待响应,服务器响应后就能得到想要获取的内容信息;

(2)解析所得到的信息内容,获取到有用的信息。并通过解析到的信息,同时找到其中新的 URL 加入爬取队列,获取新的页面;

(3)涉及到对爬取出的数据保存的问题,分布式爬虫需要保存的数据多样化,而且数据量大,通过 Redis 这种 Key-Value 数据库对数进行存储。

2.2　通过 Nutch 定义的抓取流程

Nutch 具有强大的可扩展性,有四个可扩展点,可通过编写其插件实现自定义功能,项目的抓取流程定义如下:

(1)程序会创建一个 WebDb 文件夹,这个 WebDb 文件夹用来存放将要抓取的 URL。

(2)根据 WebDb 中内容生成 fetchlist,并将 fetchlist 写入到相应的 segment,每个 segment 内最终存储的是一次抓取循环中抓到的网页和网页的索引。

(3)可以根据这些 fetchlist 中的 URL 抓取所要的网页。

(4)从网页中获取到新的 URL,将这些新得到的 URL 加入到 WebDb 中,更新 WebDb。

循环进行前面的步骤,直至达到之前预先设定的抓取深度。

(5)可以根据 WebDb 得到的网页评分和 links 更新 segments;

(6)对所抓取的网页进行索引,在索引中,要丢弃掉拥有重复内容的网页和重复的 URL。

(7)将 segments 中的索引进行合并生成用于检索的最终 index 索引。

上面是定义的基本抓取流程,在实际的抓取过程当中,采用的是广度优先抓取的方法,它贴近所爬取的大部分网站的层次和结构,网站相邻模块间的内容更具有关联性。在爬取时,广度优先策略可以避免对一个网站服务器在一段时间内的反复访问,可以减少对所爬取网站造成的影响[12]。

3　爬虫的数据解析模块

通过下载步骤之后获取到的网页,要对其进行解析处理,才能从中获取到所需要的价值信息。传统的获取页面内容的方法为使用 XPath 语句,正则表达式,或者 css 选择器。运用这些方式,首先要找到网页中的节点,即元素、属性、文本、命名空间等。例如,XPath 使用路径表达式来选取 XML 文档中的节点或者节点集。

通过上述方法来获取结果,不仅需要提前知道网页的结构逻辑,且对于所要获取信息,不同的网站信息所处的位置不同,标签不同,网页结构不同,这样不可能对所有页面做出统一规则。在爬取大量网页时,消耗的人力和时间较多。因为 Nutch 可以嵌入算法,在总结大量网页的情况并且对现有的解析算法进行优化后,设计了一种通用的算法将正文提取出来,也使其能更好地与爬虫框架贴合[12-15]。

4 实 验

通过对项目的实现,对大量数据的爬取实验,对数据进行验证和分析,得出以下实验结果。

4.1 运行环境

Nutch 是基于 Linux 系统内核的框架,所以项目框架要发布运行在 Linux 系统的服务器上。项目采用三台 CentOS 7 服务器来搭建分布式集群,CentOS 是 Linux 系统,具有高度稳定性。集群组件以 Yarn、HDFS、Zookeeper、Solr 为主,并且支持横向扩展。集群中 NodeManager 内存 12 G,内核六颗,Container 1 G 内存。爬虫集群角色如图 2 所示。

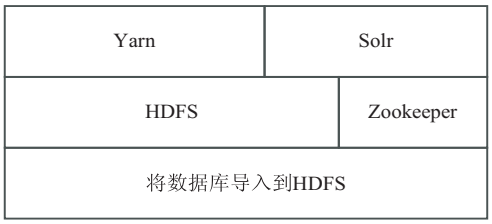

图 2 爬虫集群角色

4.2 爬虫中获取页面信息模块的实现

在爬虫的获取页面信息的模块中,设计了按照文本密集程度来找到正文的算法。以一篇文章为例,正文的部分主要集中在一定范围内,且密集度高,其他部分相对松散不密集,不是要提取的正文部分,按照此特征,可以将算法实现[16-18]。

首先要对页面进行去除标签的处理,例如"<head>…<head><body class="…">"等,去除标签后再得到的文本信息,可以降低标签对所要提取内容的干扰。去除标签后,取文本的行号,对文本进行计算,设定以每五行文本为一组,将五行中所有的字符数加起来取到和,用第一行的字符数除以五行累加的字符数和,得到一个比值 d。

$$d = \frac{M_i}{\sum_{i}^{i+4} M_i}(i = 1,2,3\cdots)$$

其中,M 的值为每一行的字符数目和。

然后,从第二行开始,计算五行所有的字符数和,用第二行的字符数除以五行累加的字符数和,得到一个比值 d,再按照第三行计算,以此类推重复计算直到最后。在后面行数不足五行时总数按照后五行总数计算。如果在计算中,连续几个得到的数值大于确定的阈值时,可以判断从第一个大的数值所在行开始,进入正文部分。

关于 d 的阈值,经过大量实际处理调整,实验统计,计算得出一个合适的数值,经过专家建议将阈值设置为 0.2。所以 d 的值数一般在 0.2 左右,可以判定为进入正文部分了。

在抽取页面信息的过程中,对于新发现的 URL 是否加入爬取队列,会有去重机制。对爬取过的网页信息,会建立一个索引,一起存在 WebDb 中,新发现的 URL 会与爬取过的 URL 进行对比,如果爬取过直接丢弃不再重复爬取,没有爬取过则增加索引加入到爬取队列中并且更新 WebDb。可以有效节省爬取时间,节省资源,避免不必要的重复数据的加入,优化爬取流程[19]。

4.3 项目的实现

通过搭建分布式集群,对上述抓取流程和解析过程的实现,爬取了大量公司的官网网页中所展示的内容,规定爬取网站上尽可能全面的信息。爬取的网站数目多,每一家公司网站不相同,深度,样式不统一,数据量就不能确定,因此不对爬取深度做要求。把爬取后的数据放入到 Solr 引擎中,Solr 会对数据进行管理,方便搜索。

通过测试发现,分布式爬虫有很好的爬取性能。在 Solr 引擎中展示的所爬取的信息,其中有 URL 地址和所爬取的具体正文内容。

4.4 爬虫关键词匹配信息模块

运用爬虫技术从网页中爬取的信息数量庞大且有冗余,同时掺杂有用的数据和无用的数据,因此,从大量所得的数据中提取出真正所需要的信息十分必要。普通的爬虫不能区分什么是有用的数据,什么是无用的数据。以"市场占有率"的相关信息为例,从网络中爬取并且存储下来[20-22]。根据相应指标关键字,从数据中抽取有用信息的方法如下:

以获取到的页面信息为对象,首先要对文本进行过滤处理,过滤掉文本中夹杂的"\n"换行符、空格等内容,这些符号不必要且干扰后续分析。以句号"。"为拆分符,对文本进行初步拆分。然后将拆分后的每一句话再进行拆分,拆分成词组的形式。根据指标对应的关键词,对拆分好的词组进行权重计算,提取出权重大于 0 的句子,并且对其排序,根据权重排序,只取权重高的前三句话。其中权重的计算方法如下:

基于句子间的内容覆盖率,给定两个句子 S_i 和 S_j,采用如下公式进行计算:

$$\text{Similarity}(S_i,S_j) = \frac{|\{t_k \mid t_k \in S_i \wedge t_k \in S_j\}|}{\log(|S_i|) + \log(|S_j|)}$$

其中,分子 t_k 是在两个句子中都出现的词组的数量,$|S_i|$ 是句子 i 的词组数,$|S_j|$ 是句子 j 的词组数。

这两个句子语义相关,并将它们连接起来,即权重:

$$W_{ij} = \text{Similarity}(S_i,S_j)$$

表 1 为对按照"市场占有率"此项指标进行匹配算法分析后的结果。

表 1　算法解析后的结果

指标名称	基础数据	数采解析结果 1	数采解析结果 2
市场占有率	企业主要产品(包括服务)评价期内的市场占有率	特殊品种系列则属于高附加值品种,用于冶金、水泥、陶瓷行业	在我国煤炭行业,本公司环境建设位居领先水平

通过表 1 可以看出,通过对一些指标的解析和匹配查找,分析出了所爬取网页中和此项指标相关联的信息。例如,“市场占有率”此项指标,从采集到的数据中解析出所需信息,印证了在数据爬取阶段,分布式爬虫可以自动完成数据的收集、解析、格式化存储,所采集的数据全面、高效。

4.5　实验分析

Nutch 底层基于 MapReduce,直接受益于分布式的所有优点,可以在集群中对资源进行调配和监控,可以很好地管理爬虫。

4.5.1　爬虫速度分析

在实验中,使用分布式爬虫和单机爬虫分别对相同的 300 家公司进行抓取,要求按照域名,爬取每家公司官网中所有展示的允许爬取的信息,观察爬虫速度上的性能。在爬取时,分布式爬虫每次所抓取的公司数量大致成整数倍增加,以此来测定爬虫爬取数量对爬取速度的影响。单机爬虫因为其自身性能限制,在爬取过程中只能对每个公司分别爬取,在实验过程中单机爬虫爬取前相同 50 家公司时,用了 17 个小时。为了对数据做比较分析,将分布式爬虫和单机爬虫爬取时间统一用分钟做单位。单机爬虫和分布式爬虫抓取公司的数量,抓取此数目公司所花费的时间和速度,对照如表 2 所示。

表 2　爬虫所爬取的公司数量和抓取所需要的时间对照

Nutch 分布式爬虫			单机爬虫		
公司数量	时间/min	平均抓取速度	公司数量	时间/min	平均抓取速度
50	20	2.5	50	1 020	0.049 0
100	32	3.125	100	2 100	0.047 6
150	43	3.488 4	150	2 940	0.051 0
200	53	3.773 6	200	3 900	0.051 3
250	60.5	4.132 2	250	4 800	0.051 4
300	77	3.896 1	300	6 120	0.049 0

图 3 为单机爬虫和分布式爬虫爬取数量和平均抓取速度折线对比图。可以看出,分布式爬虫在抓取大量信息上具有很强的优势。在一定范围内,爬取的数据量越大,分布式爬虫速度越快,越能现实其分布式的优点。而单机爬虫不仅操作繁琐,在性能上也不及分布式爬虫。分布式爬虫在大量数据采集中优势明显。

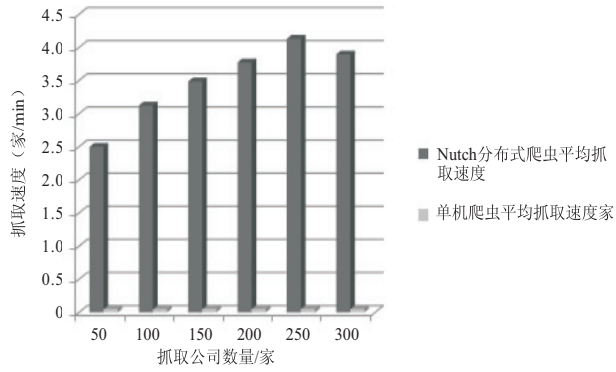

图 3　爬虫爬取数量和速度

4.5.2　爬虫数据量分析

对于爬虫所爬取的数据量的分析,要在一段规定的时间内,查看爬虫爬取的数据量,性能高的爬虫在实际的爬取作业中占有优势。一些所爬取的网站可能会有爬取时间的限制,在相同的较短的时间内,爬取的数据量越大,所收集的信息相对就越全面。实验过程中,分别用分布式爬虫和单机爬虫来爬取相同的公司,查看在相同的固定时间内分别爬取的数据量。爬取的时间成递增趋势,爬取的时间和对应的所爬取的数据量如表 3 所示。

表 3　爬虫爬取时间和对应爬取数据量对照

分布式爬虫		单机爬虫	
规定时间/min	爬取数据量/条	规定时间/min	爬取数据量/条
10	4 876	10	122
11	5 768	11	153
12	6 673	12	187
13	7 539	13	219
14	8 807	14	242
15	9 901	15	273

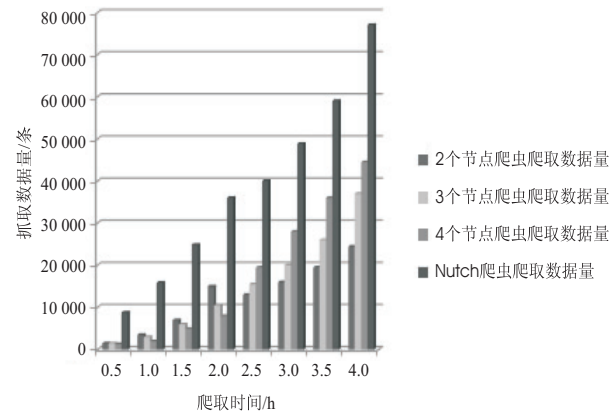

图 4　基于 Nutch 的分布式爬虫和其它分布式爬虫爬取数据量

表 3 为单机爬虫和分布式爬虫在统一规定时间内爬取时间和对应爬取数量的对照,体现了基于 Nutch

分布式爬虫在抓取大量信息上所具有的优势。在相同的时间内,分布式爬虫所抓取的数据量明显多于普通单机爬虫,相对而言,爬取信息自然也更全面。

图 4 为基于 Nutch 的分布式爬虫和其它分布式爬虫爬取数据量对比柱状图,从数据中可以看出,在相同时间内,基于 Nutch 的分布式爬虫所抓取数据量在一定程度上明显多于其它分布式爬虫所抓取的数据量,基于 Nutch 的分布式爬虫在性能上更优。

5 结束语

通过调研和总结,了解了目前网络爬虫的存在价值和意义,分布式爬虫的优势,以及实现方法等。通过对分布式集群环境的搭建,运用 Nutch 对大量网站的爬取、解析,印证了项目设计和实现的可行性。Nutch 爬虫的可扩展性和灵活性有很重要的意义。在异构数据的融合上,所设计的自动去重机制使爬虫在工作中对于相同地址不做重复爬取,从源头上避免了数据的重复。设计的提取页面信息模块,能准确找出网页中展示的信息,关键词匹配模块,可以根据关键词语义有效匹配出所要收集的信息,高效准确。方便快捷的检索能力,极大地方便了用户,为其带来了极好的体验。

将爬虫技术与其它现有技术相结合,更好地服务生活,将是接下来研究的目标。

参考文献:

[1] 陶耀东,向中希. 基于改进 Kademlia 协议的分布式爬虫[J]. 计算机系统应用,2016,25(4):156-161.

[2] CHEN X,SHANG W. Research and design of web crawler for music resources finding[J]. Applied Mechanics and Materials,2014,543-547:2957-2960.

[3] WANG R,RHO S,CHEN B,et al. Modeling of large-scale social network services based on mechanisms of information diffusion:Sina Weibo as a case study[J]. Future Generation Computer Systems,2017,74:291-301.

[4] 潘 巍,晋 松. 分布式网络爬虫系统的研究现状[J]. 经济技术协作信息,2017(23):85-89.

[5] 刘光金. 大数据处理对电子商务的影响分析[J]. 计算机光盘软件与应用,2014(17):25-26.

[6] 徐雁飞,刘 渊,吴文鹏. 社交网络数据采集技术研究与应用[J]. 计算机科学,2017,44(1):277-282.

[7] 郭 涛,黄铭钧. 社区网络爬虫的设计与实现[J]. 智能计算机与应用,2012,2(4):65-67.

[8] 唐晓凤. 数据挖掘课程教学改革探讨[J]. 中外企业家,2016(27):190.

[9] 靳永超,吴怀谷. 基于 Storm 和 Hadoop 的大数据处理架构的研究[J]. 现代计算机:专业版,2015(4):9-12.

[10] 栾景超,马志强,李昊甦,等. Hadoop 分布式文件系统资源管理器的设计与实现[J]. 科研信息化技术与应用,2014(1):41-52.

[11] 黄志敏,曾学文,陈 君. 一种基于 Kademlia 的全分布式爬虫集群方法[J]. 计算机科学,2014,41(3):124-128.

[12] 范传辉. Python 爬虫开发与项目实践[M]. 北京:机械工业出版社,2017.

[13] 董禹龙,杨连贺,马 欣. 主动获取式的分布式网络爬虫集群方法研究[J]. 计算机科学,2018,45(S1):428-432.

[14] BEDI P,THUKRAL A,BANATI H,et al. A multi-threaded semantic focused crawler[J]. Journal of Computer Science and Technology,2012,27(6):1233-1242.

[15] RUO R, WANG H, CHEN M, et al. Parallelizing the extraction of fresh information from online social networks[J]. Future Generation Computer Systems,2016,59:33-46.

[16] GU R,JIANG J F. Research and implementation of topic crawler based on Hadoop[J]. Applied Mechanics & Materials,2014,651-653:1896-1900.

[17] GUPTA K,MITTAL V,BISHNOI B,et al. AcT:accuracy-aware crawling techniques for cloud-crawler[J]. World Wide Web,2016,19(1):69-88.

[18] 池勇敏,郝泳涛. 分布式主题爬虫的设计与实现[J]. 计算机应用与软件,2010,27(12):135-138.

[19] LIN K,CHUNG S,LIN C. A fast and distributed algorithm for mining frequent patterns in congested networks[J]. Computing,2016,98(3):235-256.

[20] 周孝锞,郭克华. 基于网络爬虫和改进的算法的 LCS 网站更新监测[J]. 计算机应用与软件,2017,34(1):222-229.

[21] JAGANATHAN P,KARTHIKEYAN T. Highly efficient architecture for scalable focused crawling using incremental parallel Web crawler[J]. Journal of Computer Science,2015,11(1):120-126.

[22] YU D,CHEN N,RAN X. Computational modeling of Weibo user influence based on information interactive network[J]. Online Information Review,2016,40(7):867-881.