分布式全网职位搜索引擎的研究与实现

付剑生[1],徐林龙[2],林文斌[1]
(1. 西南交通大学 物理科学与技术学院,四川 成都 610000;
2. 西南交通大学 数学学院,四川 成都 610000)

摘 要:传统招聘网站所查寻的信息仅限于站内搜索,而且每个招聘网站往往都会有重复的招聘信息,导致重复投递,对求职者和招聘者都造成了资源浪费。文中研究和分析了基于 Lucene 的分布式全文搜索引擎 Solrcloud,设计了全网职位搜索引擎系统。该系统采用 Bloom Filter 进行数据及 URL 的去重,通过使用 Zookeeper 提供分布式同步服务,并通过多线程来实现网页并发抓取。通过对系统的测试表明,该系统具有良好的可靠性和应用性,并在大数据量的情况下保证了搜索的效率及准确性。

关键词:Solrcloud;职位搜索;网络爬虫;分布式搜索引擎
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2015)05-0006-04
doi:10.3969/j.issn.1673-629X.2015.05.002

Research and Implementation of Distributed Network-wide Job Search Engine

FU Jian-sheng[1], XU Lin-long[2], LIN Wen-bin[1]
(1. School of Physics and Technology, Southwest Jiaotong University, Chengdu 610000, China;
2. School of Mathematics, Southwest Jiaotong University, Chengdu 610000, China)

Abstract: The information traditional recruitment websites queried is restricted to site search, and each recruitment website may have duplicate information, which will lead to a waste of resource for both job hunters and recruiters. Based on research and analysis about Solrcloud, a distributed full text search engine based on Lucene, design a network-wide job search engine. This system uses a method for date and URL duplication removal based on Bloom Filter, provides distributed synchronization service through Zookeeper, and realizes webpage crawling through multithreading. Experiments indicate that this system has a high reliability and applicability, and improves the efficiency and accuracy on large data sets.

Key words: Solrcloud; job search; web crawlers; distributed search engine

0 引 言

互联网的普及和发展给人们带来了大量的实时信息,满足了用户在信息时代对信息的需求。近年来,各招聘网站的迅速兴起,给求职者提供了一种在网上找工作的选择,通过招聘网站找工作,已经成为了非常重要的一种途径。但是各大招聘网站存在的招聘信息重复、刷新时间过快等问题往往会给求职者带来困扰,进而导致简历重复投递,对求职者和招聘方都造成了资源的浪费。

为此,文中通过对 Solrcloud[1-2] 的研究,以及对全网职位信息抓取、过滤去重、构建分布式倒排索引等,设计实现了一个高效、稳定、准确的分布式全网职位搜索引擎[3-5],解决了上述问题。

1 Solrcloud 及相关技术分析

1.1 Solrcloud 概述

Solrcloud 是基于 Solr 和 Zookeeper 的分布式[6] 搜索方案,主要通过将 Zookeeper[7] 作为集群的信息配置中心,实现分布式同步服务,其中 Solr 是以 Lucene 为基础实现的文本检索应用服务。基于 Solrcloud 的典型应用系统如图 1 所示。

收稿日期:2014-07-04 **修回日期**:2014-10-10 **网络出版时间**:2015-04-22
基金项目:教育部新世纪优秀人才支持计划项目(NCET-10-0702)
作者简介:付剑生(1987-),男,硕士,CCF 会员,研究方向为算法分析与设计、分布式计算;林文斌,教授,研究方向为高性能计算、数据挖掘。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150422.1005.019.html

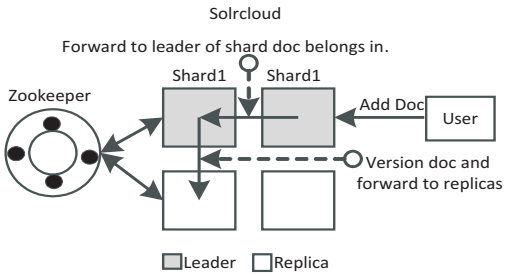

图 1　Solrcloud 的典型应用系统图

从图中可知 Solrcloud 的索引可以分布在多个 Shard 中,同时每个 Shard 又可以有一个 Leader 节点以及多个 Replica 节点。另外,集群的所有配置信息都放在 Zookeeper 中进行统一维护。

1.2　Solrcloud 特色功能

Solrcloud 作为 Solr 的分布式实现具有多个自身独有的功能,包括

(1)集中式的信息配置方式,使用 Zookeeper 进行集中配置;

(2)自动化容错处理,Solrcloud 对索引分片,并对每个分片创建多个 Replica;

(3)近实时数据检索,立即推送式的 Replica;

(4)数据检索时的自动负载均衡,Solrcloud 索引的多个 Replica 可以分布在多台机器上,均衡查询压力;

(5)索引及索引分片的自动分发,发送文档到任何节点,它都会转发到正确节点;

(6)通过 MapReduce 实现索引的批量创建。

其中,近实时索引数据的建立过程是通过设定软提交方式,仅把数据提交到内存,然后每 1 ~ 10 min 自动触发硬提交。另外,通过将索引存放于 HDFS 上,实现了倒排索引的虚拟化存储。

1.3　Replica 概述

Replica 是 Shard 的一个拷贝。每个 Replica 存在于 Solr 的一个 Core 中。一个命名为"jobs"的 collection 以 numShards = 1 创建,并且指定 replicationFactor 设置为 2,这会产生 2 个 replicas,也就是对应会有 2 个 Core,每个在不同的机器或者 Solr 实例。一个会被命名为 jobs_shard1_replica1,另一个命名为 jobs_shard1_replica2。它们中的一个会被选举为 Leader。

1.4　Zookeeper 概述

Zookeeper 是一个为分布式应用提供一致性服务的软件,它是开源的 Hadoop 项目中的一个子项目,是作为分布式应用建立更高层次的同步、配置管理、群组以及名称服务。在编程上,Zookeeper 设计很简单,所使用的数据模型风格很像文件系统的目录树结构。

Zookeeper 分为两个部分:服务器端和客户端。客户端只连接到整个 Zookeeper 服务的某一台服务器上。客户端使用并维护一个 TCP 连接,通过这个连接发送请求、接收响应、获取观察的事件以及发送心跳。如果这个 TCP 连接中断,客户端将尝试连接到另外的 Zookeeper 服务器。

启动 Zookeeper 服务器集群环境后,多个 Zookeeper 服务器在工作前会选举出一个 Leader,在接下来的工作中如果这个被选举出来的 Leader 停止服务了,那么剩下正在运行的 Zookeeper 服务器会知道这个 Leader 已经停止服务,然后会在其他正常提供服务的 Zookeeper 集群中继续选出一个 Leader,选举出 Leader 的目的是为了可以在分布式的环境中保证数据的一致性。在 Solrcloud 中,Zookeeper 提供分布式锁功能并处理 Leader 选举。

另外,Zookeeper 支持 watch(观察)的概念。客户端可以在每个 node 节点上设置一个观察。如果被观察服务端的 node 节点有变更,那么 watch 就会被触发。这个 watch 所属的客户端将接收到一个通知包被告知节点已经发生变化,若客户端和所连接的 Zookeeper 服务器断开连接时,其他客户端也会收到通知。

2　分布式全网职位搜索引擎的设计与实现

2.1　系统架构

分布式全网职位搜索引擎系统架构如图 2 所示。其中 spider 采用非递归的多线程技术并发对 Web 中的职位信息进行采集、存储、处理等;索引与搜索模块通过 Solrcloud 进行建立与检索。

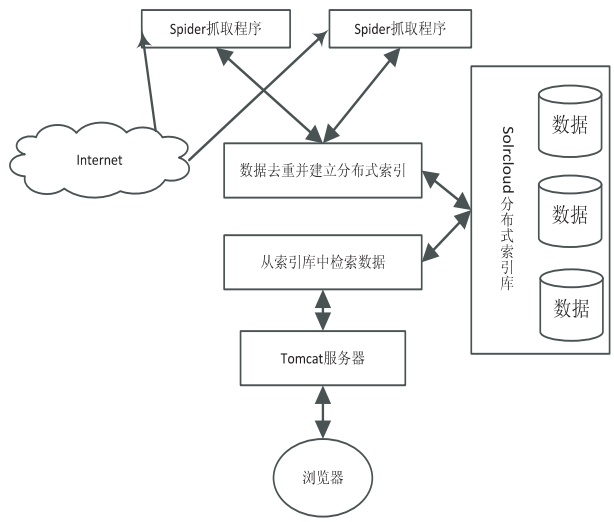

图 2　分布式全网职位搜索引擎架构图

2.2　职位采集的实现

在采集模块主要构造以下组件:

(1)JobsBean 组件:使用类对象的方式存放抽取出的职位各个属性值。具体属性值如表 1 所示。

(2)Dataconn 组件:构建数据库连接池,从而实现对已抓取数据保存时减少数据库连接时间,加速程序

运行。

表1 职位对象的属性

Name	Code
职位id	JobsId
职位名称	JobsName
公司名称	CompanyName
公司规模	CompanyScale
公司类型	CompanyType
公司行业	Companyindu
性别要求	Sex
招聘人数	RecruitNum
年龄要求	EmployeeAge
工作性质	JobsType
截止时间	DeadTime
学历要求	EducateBac
薪资待遇	Salary
工作经验	JobExpe
发布时间	PublishDate
工作地点	WorkPlace
职位类别	JobClass
职位描述	JobDescr
来源地址	JobURL

(3)Bloom Filter[8]组件:实现对URL过滤并对已抓取的数据进行去重,防止对同一公司发布的相同职位信息的重复抓取。

(4)Spider组件:通过启动Initial类先进行程序的初始化,包括种子URL的获取、数据库连接池的初始化及Bloom Filter的初始化,然后启动CrawlerManager类构建抓取的多线程进行分布式并发抓取。

(5)Parser组件:解析来自不同网站的职位信息,抽取相应的属性值,并保存在元数据库中。

2.3 基于Solrcloud的分布式索引

使用Solrcloud构建Jobs索引对象,并使用两个分片(Shard)及三个冗余备份(Replication),以保证数据的可用性及安全性。并且将配置文件信息传入Zookeeper中进行集群统一管理,构建的索引对象结构如图3所示。最后将抓取的职位信息加入到分布式索引中,通过多线程来并发写入索引,加快构建索引的速度。

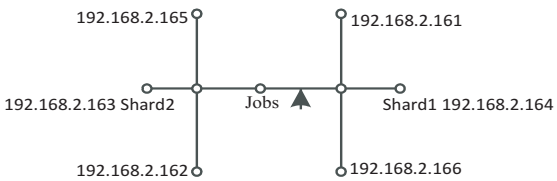

图3 索引对象Jobs结构图

2.4 搜索功能的实现。

对于已抓取的170万数据建立索引后使用solrj[9]的API对Solrcloud进行职位数据的检索,使用的查询方法及部分关键代码如下:

```
/**
 * @TitleSearchResult
 * @Description 通过此方法进行数据的检索
 * @param field 查询的字段名称数组
 * @param key 查询的字段名称对应的值
 * @param start 查询的起始位置
 * @param count 一次查询出来的数据量
 * @paramsortfield 需要排序的字段数组
 * @param flag 需要排序的字段的排序方式如果为true则为升序,如果为false则为降序
 * @paramhightlight 是否需要高亮显示
 * @Return QueryResponse
 * @Date2014-6-25
 */
PublicQueryResponseSearchResult(String[] field, String[] key, int start, int count, String[] sortfield, Boolean[] flag, Boolean hightlight){
//检测输入是否合法
if(null==field || null==key || field.length! =key.length){
return null;
}
…
…
SolrQuery query=null;
//初始化查询对象
query=new SolrQuery(field[0] + ":" + key[0]);
for(int i=0;i<field.length;i++) {
query.addFilterQuery(field[i] + ":" + key[i]);
}
//设置起始位置与返回结果数
query.setStart(start);
query.setRows(count);
//设置排序
if(!(null==sortfield|| null==flag|| sortfield.length! =flag.length)) {
for(int i=0;i<sortfield.length;i++) {
if(flag[i]) {
query.addSortField(sortfield[i], SolrQuery.ORDER.asc);
} else {
query.addSortField(sortfield[i], SolrQuery.ORDER.desc);
…
…
query.setHighlight(true);//开启高亮组件 query.addHighlightField("jobsName");//高亮字段
query.setHighlightSimplePre("<font color=\"red\">");//飘红标记 query.setHighlightSimplePost("</font>");
```

```
…
…
//需要分组统计的数据
query. setFacet( true) ;//设置 facet=on
query. setFacetLimit(10) ;//限制 facet 返回的数量
query. setFacetMissing( false) ;//不统计结果为 null 的值
query. setFacetMinCount(1) ;//设置返回的数据中每个分组的数据最小值,比如设置为 1,则统计数量最小为 1,不然不显示
query. addFacetField ( new String[ ] { " salary", " educateBackground"," jobExperience "," companytype ", " jobsType " }) ;//设置需要 facet 的字段
…
…
try{
rsp=solrServer. query( query) ;
} catch( Exception e) {
e. printStackTrace( ) ;
return null;
}
//返回查询结果
return rsp;
}
```

通过 Solrcloud[10-12]的客户端进行数据的检索测试,发现数据检索性能及稳定性大大提高,表 2 可以直观地反映这一问题。

表 2　Solrcloud 关键字查询

查询关键词	返回数据量/条	查询时间/ms
软件工程师	11 189	32
销售	200 004	26
教师	12 215	19
实习	19 945	17
云计算	245	17

根据检索结果分析,Solrcloud 满足了职位检索系统的稳定性及高效性,同时也暴露出分词算法[13]对检索效率有着很大的影响。

3　结束语

运用分布式搜索引擎 Solrcloud 等技术实现的分布式全网职位搜索引擎系统可以稳定高效的运行,符合分布式搜索引擎原理的探究,具有一定的现实意义。同时系统还有一些有待改进的地方,比如改进分词算法以提高倒排索引建立及查询的效率,数据抓取速度可以进一步加快等,这些都是以后研究的内容。

参考文献:

[1] Smiley D, Pugh D E. Apache Solr 3 enterprise search server [M]. [s. l.]: Packt Publishing Ltd, 2011.

[2] Kué R. Apache Solr 4 cookbook [M]. [s. l.]: Packt Publishing Ltd, 2013.

[3] 印　鉴,陈忆群,张　钢. 搜索引擎技术研究与发展[J]. 计算机工程,2005,31(14):54-56.

[4] 李振龙. Web 信息检索的技术分析与发展策略研究[J]. 计算机科学,2006,33(4):181-184.

[5] 郑榕增,林世平. 基于 Lucene 的中文倒排索引技术的研究[J]. 计算机技术与发展,2010,20(3):80-83.

[6] 吴广印. 分布式学术搜索引擎研制及其大数据应用[J]. 数字图书馆论坛,2013(6):10-18.

[7] Junqueira F, Reed B. ZooKeeper: distributed process coordination[M]. [s. l.]: O'Reilly Media, Inc, 2013.

[8] Xie Y. Index searching using a bloom filter: U. S. , 8396873 [P]. 2013-03-12.

[9] 霍　庆,刘培植. 使用 Solr 为大数据库搭建搜索引擎[J]. 软件,2011,32(6):11-14.

[10] 冯　祥,邱志超. 基于 Solr 的海量日志信息查询性能优化的研究[J]. 硅谷,2014,7(3):37-39.

[11] 傅巍玮,李仁发,刘钰峰,等. 基于 Solr 的分布式实时搜索模型研究与实现[J]. 电信科学,2011,27(11):51-56.

[12] 姚晓娜,祝忠明. 基于分面搜索引擎 Solr 的机构知识库访问统计[J]. 现代图书情报技术,2011(7):37-40.

[13] 张启宇,朱　玲,张雅萍. 中文分词算法研究综述[J]. 情报探索,2008(11):53-56.

分布式全网职位搜索引擎的研究与实现

作者： 付剑生， 徐林龙， 林文斌， FU Jian-sheng， XU Lin-long， LIN Wen-bin

作者单位： 付剑生,林文斌,FU Jian-sheng,LIN Wen-bin(西南交通大学 物理科学与技术学院,四川 成都,610000)， 徐林龙,XU Lin-long(西南交通大学 数学学院,四川 成都,610000)

刊名： 计算机技术与发展 ISTIC

英文刊名： Computer Technology and Development

年，卷(期)： 2015(5)