

文章编号:1006-2475(2015)06-0069-05

基于 Elasticsearch 的数字图书馆检索系统

张建中¹,黄艳飞²,熊拥军³

(1. 中南大学信息科学与工程学院,湖南 长沙 410083; 2. 中南大学软件学院,湖南 长沙 410075;
3. 中南大学图书馆,湖南 长沙 410083)

摘要:针对大数据时代下图书馆文献的存储和检索难题,运用 HDFS 分布式文件系统实现图书馆文献资源的海量存储,采用 Elasticsearch 分布式索引技术对资源进行分布式索引和检索,构建了一个高效的、分布式的数字图书馆检索系统。测试结果显示,在大数据量下,系统检索时间约为传统 Oracle 数据库检索时间的 1/20,并带有缓存功能。

关键词:ElasticSearch; 数字图书馆; 海量存储; 分布式索引; 分布式检索

中图分类号:TP393 **文献标识码:**A **doi:** 10.3969/j.issn.1006-2475.2015.06.015

Digital Library Retrieval System Based on ElasticSearch

ZHANG Jian-zhong¹, HUANG Yan-fei², XIONG Yong-jun³

(1. School of Information Science and Engineering, Central South University, Changsha 410083, China;
2. School of Software, Central South University, Changsha 410075, China;
3. Central South University Library, Changsha 410083, China)

Abstract: For the storage and retrieval problem of library works in the era of big data, this paper implemented the massive storage of library works with HDFS and used ElasticSearch to distributed index and retrieval. Test results show that under the large amount of data, this system needs about one-twentieth retrieval time of the traditional Oracle database for searching the same search words, and it has the cache function. So an efficient and distributed digital library retrieval system is built.

Key words: ElasticSearch; digital library; mass storage; distributed index; distributed retrieval

0 引 言

随着大数据时代的来临,数据量以极快的速度增长,形成了电子期刊、音视频资源在内的海量数字资源,图书馆也不可避免地面临着大数据信息浪潮的冲击。首先,日益增长的数字资源的存储问题;其次,海量数据中快速检索有用信息越来越困难的问题。目前,数字图书馆检索系统主要采用 2 种技术模式:1) 以数据库为基础,利用成熟的商业数据库检索信息,如 Oracle;2) 在全文检索技术基础上开发的检索系统,如百度。然而,数据库随着数据量的不断增大,检索效率也会大幅度降低^[1]。因此,本文运用第 2 种模式,提出一种基于 HDFS 存储和 Elasticsearch 分布式索引和检索的方案,为用户提供更快捷的检索服务。

1 技术综述

1.1 元数据

在数字化的过程中,异构的图书馆数据库间进行检索存在不少障碍。元数据为图书馆提供了很好的信息内容描述。目前信息资源组织中常用的元数据格式包括: MARC 格式、文献编码计划书(TEI)和都柏林核心数据(Dublin Core, DC)等。在众多的元数据项目中,DC 在图书馆界和情报界是应用最广、影响最大的一个国际性项目。DC 的元素是结构化,有层次的,支持字段检索,提供对特定资源足够全面的描述信息,使用户不用真正链接到检索资源本身就能对资源有全面的了解。它有内在性、可扩展性、独立句法结构、可选择性、可重复性和可修改性 6 大特性,得

收稿日期:2015-02-03

作者简介: 张建中(1955-),男,河北张家口人,中南大学信息科学与工程学院教授,博士,研究方向:数字图书馆,信息处理;黄艳飞(1989-),男,河北邯郸人,中南大学软件学院硕士研究生,研究方向:数字图书馆文献检索;熊拥军(1972-),男,湖南长沙人,中南大学图书馆副研究馆员,博士,研究方向:数字图书馆,信息处理。

到了越来越多的图书馆界的响应,成为图书馆网络信息组织与管理的发展主流^[2]。

1.2 HDFS

HDFS 是 Hadoop 框架的子项目,是按流式数据访问的分布式文件系统,具有处理超大文件的能力,可以运行于廉价的机器上。它所具有的高容错、高可扩展性、高吞吐率、高可靠性等特征为海量数据提供了安全保障,为海量数据存储应用处理带来了很大便利,适合处理大数据文件。

在一个 HDFS 中主要的 2 个角色是存放管理文件系统元信息 NameNode 和存放管理实际文件数据 DataNode。HDFS 对外暴露的是一个操作系统的命名空间,让用户对其操作。而底层存储是通过将文件切割成小块(Block)存储在不同的 DataNode 上。HDFS 使用多份存储来保证数据不会丢失,每一个 Block 会被复制多份存储在不同的 DataNode 上,在 Block 放置时,尽可能均匀地分布在集群中,防止单个机器或者机柜的失败造成数据的丢失。当 NameNode 在一段时间没有收到某台 DataNode 的“心跳”后,则将该 DataNode 判定为死亡状态,并将丢失的 Block 在现有的集群中拷贝到指定份数。同时,还会根据一些情况去重新平衡数据分布,保证数据分布的平衡性^[3]。

1.3 ElasticSearch

ElasticSearch^[4-7] (简称 ES) 是一个基于 Lucene^[8-12] 构建的开源、分布式、RESTful 搜索引擎。在云计算中,ES 能够达到实时搜索,稳定,可靠,快速,支持通过 HTTP 使用 JSON 进行数据索引。主要特性有:

1) 集群(cluster)。集群中有多个节点,其中一个为主节点,可以通过选举产生,主从节点是相对于集群内部来定义的。ES 的一个概念是去中心化,是相对于集群外部来定义的。因为从外部来看 ES 集群,在逻辑上是个整体,与任何一个节点的通信和与整个集群通信是等价的。

2) 索引分片(shards)。ES 把一个完整的索引分成多个分片,把一个大的索引拆分成多个,分布到不同的节点上,构成分布式检索。

3) 索引副本(replicas)。ES 可以设置多个索引副本,主要作用:①提高系统的容错性,当某个节点的某个分片损坏或丢失时,可以从副本中恢复;②提高 ES 的查询效率,ES 会自动对检索请求进行负载均衡。

4) 数据恢复(recovery)。ES 在有节点加入或退出时会根据机器的负载对索引分片进行重新分配,挂掉的节点重新启动时也会进行数据恢复。

5) ES 索引的持久化存储方式(gateway)。ES 默认先把索引存放到内存中,当内存满了时再持久化到硬盘。当这个 ES 集群关闭再重新启动时就会从 gateway 中读取索引数据。

6) discovery.zen: ES 的自动发现节点机制。ES 是一个基于 P2P 的系统,先通过广播寻找存在的节点,再通过多播协议进行节点之间的通信,同时也支持点对点的交互。

ES 的索引文件存储方式有 4 种:

- 1) 像普通的 Lucene 索引,存储在本地文件系统中。
- 2) 存储在分布式文件系统中,如 frees。
- 3) 存储在 Hadoop 的 HDFS 中。
- 4) 存储在亚马逊的 S3 云平台中。

1.3.1 索引数据结构

ES 索引采用倒排索引机制构建索引。

1) 项(Term):最小的索引单位,直接代表一个关键词以及其在源文档中的出现位置和出现次数等信息。

2) 域(Field):一个关联的元组,包括域名和域值。域名是一个字符串,域值是一个项。

3) 文档(Document):包括所有域信息。

4) 段(Segment):包含若干文档,若干段组成子索引或索引^[13]。如图 1 所示。

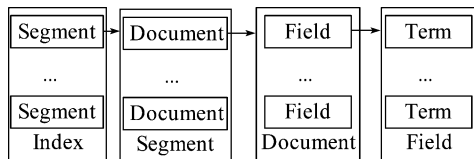


图 1 ES 索引的数据结构

1.3.2 分词技术

分词技术是查询串进行初步处理后,用各种匹配方法进行匹配的一种技术。分为 3 类:

1) 机械分词法:按照一定策略将待分词的汉字串与词典中的词条进行匹配,若在词典中找到该汉字串,则匹配成功。根据扫描方向的不同分为正向匹配和逆向匹配;根据不同长度优先匹配原则,分为最大(最长)匹配和最小(最短)匹配。根据与词性标注过程是否相结合,又可以分为单纯分词方法和分词与标注相结合的一体化方法。

2) 语义分词法:该方法主要基于句法、语法分析,并结合语义分析,通过对上下文内容所提供信息的分析对词进行定界。目前基于知识的分词系统还处在试验阶段。

3) 统计分词法:根据词组的统计,计算 2 个相邻字出现的频率,确定是否为词。依此作为分词依据。

目前主要的分词器^[13]如表 1 所示。

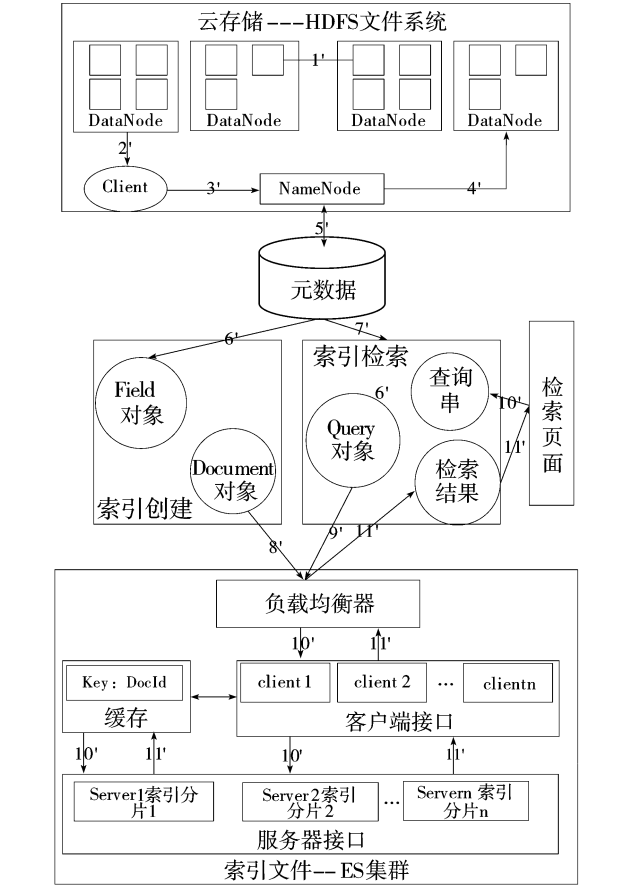
表1 各类分词器

分词器	内部操作步骤
Standard-Analyzer	基于复杂的语法实现词汇单元化,这种语法规则可以识别Email地址、首字母缩写、字母数字等
Simple-Analyzer	在分字母字符切分文本,并将其转化为小写形式
IK-Analyzer	实现了以词典作为基础信息的正反向切分,以及正反双向最大匹配切分,是第三方实现的分词器
ICTCLAS4J	中科院的分词器,是基于语义分词的,简化了传统分词程序的复杂度

2 设计与实现

2.1 系统体系结构

数字图书馆面临的难题是海量数据存储和快速检索。针对这2个难题,本系统采用HDFS文件系统存储图书馆数字资源附件和ES集群实现分布式索引与检索。整体主要分为4个模块:数据存储、索引创建、索引文件和索引检索。如图2所示。



(注:1'—复制;2'—读取;3'—元数据操作;4'—块操作;5'—通过filepath关联;6'—1k分词解析;7'—获取元数据;8'—IndexWriter创建索引,并保存;9'—IndexSearch分布式检索;10'—查询请求;11'—返回结果)

图2 数字图书馆检索系统体系结构

2.2 数据存储模块

云存储HDFS分布式文件系统,采用4台Linux机器组成的集群,每个块保存3个副本。云成为基础设施后,使得图书馆无需承担日益高昂的数据中心管理成本,不再需要购买多种服务器和大容量存储设备,可“按需供应”服务,降低成本,更重要的是,它为图书馆提供了硬件及应用托管的环境,为数字图书馆公共服务平台供高效、安全和稳定的存储环境^[14]。

1)元数据设计。

对于学校的自主资源,必须建立自己的元数据库,对资源进行分类,编目标引,建立完整的数据库。本文以DC元数据核心为依据,根据实际需要,对元素进行利用和扩展,形成一套适合高校数字图书馆要求的元数据的数据结构。在本系统的元数据设计时,除了保留DC_Title、DC_Creator、DC_Type等原有23个字段,还对元数据字段进行了扩展:DC_DOWN-LOAD(文献被下载次数)、DC_ACCESS(文献被访问次数)。

2)数据存储。

图书馆文献资源的存储分为2种:文献元数据的存储和文献附件的存储。元数据保存在Oracle数据库中,且在每个元数据中添加了filepath字段来保存对应附件在HDFS云存储上的路径;附件保存在HDFS云存储上。在HDFS系统中的修改操作是通过先删除附件,再上传附件来解决,同时需要修改该附件对应元数据的filepath字段;在删除云存储上的附件时候,也要把该附件所对应的元数据filepath字段值清空。

2.3 索引创建模块

ES是基于Lucene的全文检索技术,其索引技术的底层实现和Lucene一样。因此将创建索引的过程分为4步:从存放元数据的Oracle数据库获取数据、解析生成Field对象、构建Document对象、IndexWriter建立索引。其中Field对象的生成是按照Oracle数据库字段和ES索引中字段的对应方式进行^[15],流程如图3所示。

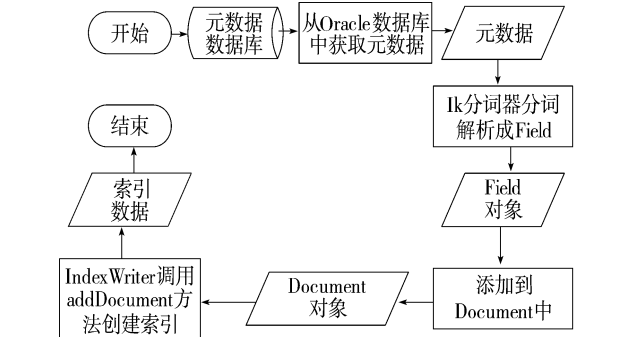


图3 ES索引创建流程图

2.4 索引文件模块

2.4.1 ES 集群搭建

- 搭建过程:
- 1)每个节点安装jdk1.6.0;
 - 2)下载elasticsearch-1.1.1.zip压缩包,解压;
 - 3)下载elasticsearchservicewrapper,解压;将service文件夹放到ES的bin目录下,配置service中elasticsearch.conf文件,主要属性配置如图4所示;
 - 4)配置elasticsearch.yml文件,如图5所示,并添加ik分词器的配置数据;
 - 5)运行bin目录下elasticsearch.bat,搭建集群完成。

本系统的ES集群包含3个ES节点,节点之间局域网连接,可以互相ping通。其中用的分词器是ik分词器,需分别在ES配置、创建索引Mapping、索引检索时进行分词器设置,才能达到分词的效果。

```
set. default. ES_HEAP_SIZE = 1024
wrapper. java. initmemory = % ES_HEAP_SIZE%
wrapper. java. maxmemory = % ES_HEAP_SIZE%
```

图4 service 的配置

```
indices. fielddata. cache. size: 60%
indices. breaker. fielddata. limit: 80%
cluster. name: es_library
node. name: "es_library_node1"
node. master: true
node. data: true
index. number_of_shards: 5
index. number_of_replicas: 1
discovery. zen. ping. multicast. enabled: true
```

图5 ES 集群配置信息

图4、图5中,ES_HEAP_SIZE是ES堆内存的大小,一般设置为系统内存的50%。indices.fielddata.cache.size是结果缓存的大小,设置为ES堆内存的60%,默认是没有设置的。这个属性是node级别(即分别在ES节点都有各自的结果缓存)。ES检索结果存放在此,下次访问时不用再次从磁盘加载。

indices.breaker.fielddata.limit是断路器的限值,不能比fielddata.cache小,默认是80%。如果待加载的检索结果存放到结果缓存后超过ES堆内存的80%时,就会事先利用LRU算法将结果缓存中最近不访问的检索结果删除。

fielddata.cache和breaker.fielddata.limit如果均保持默认,fielddata会一直增长直到触发breaker为止,旧数据不会被删除,新数据也不会再加进来,严重影响ES性能。因此在使用中对这2个参数进行设置。

2.4.2 分布式索引文件

本系统在ES集群上建立多个索引库,不同的索

引库存放不同类别的图书馆数字文献,如:医学、科技、生活等类别。ES将索引创建模块所创建的索引数据分片,根据索引类别和elasticsearch.yml的配置存放到各个ES节点的不同索引库的不同分片中。分布式索引文件存放格式如图6所示。



	_state	2015/1/29 11:41	文件夹
	0	2015/1/29 11:40	文件夹
	1	2015/1/29 11:40	文件夹
	2	2015/1/29 11:40	文件夹
	3	2015/1/29 11:40	文件夹
	4	2015/1/29 11:40	文件夹

图6 分布式索引文件

2.5 索引检索模块

本系统分为统一检索、二次检索和高级检索3种检索方式。但三者内部分布式检索流程是一样的,过程:1)ik分词器对输入的查询语句进行分词解析;2)为实现在多个索引列上支持多个关键字检索,采用MultiFieldQueryParser生成Query对象;3)创建IndexSearcher实例,各个ES节点进行分布式检索索引文件,将各节点中符合条件的结果进行合并、排序,保存到SearchReponse结果集中;4)通过ObjectMapper将JSON数据直接转换成List(hashMap)样式返回到结果列表中;5)进一步将检索结果显示给用户。用户可以通过filepath路径从HDFS文件系统上下载文献附件^[15-16],检索页面如图7所示。



图7 检索页面

3 系统测试

3.1 测试环境

- 1)ES集群:3台相同配置的电脑(操作系统:32位Windows7;CPU:2.00GHz;内存:1.98GB)。
- 2)各节点配置均为:ES堆内存为1G、field内存为堆内存60%、断路器限值是堆内存80%。
- 3)节点ip:*. *. *.91、*. *. *.92、*. *. *.93。
- 4)主要技术: Elasticsearch-1.1.1。
- 5)开发工具: MyEclipse10、Oracle11g、PL/SQL Developer。
- 6)运行环境: JDK1.6.0。

3.2 测试数据

本系统选取的测试数据为中南大学图书馆的文

献资源数据。选取医学、工业、农业等方面的文献数据,约 32 万份。测试检索词选择:维生素、观察、胃镜 3 个词,分别在 Oracle 数据库、单个 ES 和 ES 集群下进行约 2 万、10 万、30 万不同数量级别的检索速度测试,并进行对比和分析。

3.3 测试结果

表 2 22013 份测试数据的结果

检索词 时间 次数 /s		维生素	观察	胃镜
Oracle	1	2.805	2.884	2.626
	2	2.675	2.875	2.759
单个 ES	1	0.25	0.256	0.241
	2	0.093	0.102	0.087
ES 集群	1	0.263	0.457	0.169
	2	0.097	0.107	0.093
检索结果数/份		47	1397	21

表 3 107634 份测试数据的结果

检索词 时间 次数 /s		维生素	观察	胃镜
Oracle	1	2.424	4.548	2.571
	2	2.566	3.897	2.32
单个 ES	1	0.497	0.54	0.302
	2	0.105	0.106	0.104
ES 集群	1	0.447	0.56	0.358
	2	0.114	0.116	0.108
检索结果数/份		193	7211	102

表 4 323249 份测试数据的结果

检索词 时间 次数 /s		维生素	观察	胃镜
Oracle	1	13.128	15.548	13.386
	2	15.061	16.896	14.104
单个 ES	1	1.476	1.651	1.361
	2	0.109	0.111	0.108
ES 集群	1	0.545	0.573	0.459
	2	0.118	0.12	0.113
检索结果数/份		241	7830	108

根据表 2~表 4 测试结果,可以得出 2 条结论:1)数据量较小的情况下,检索速度由慢到快依次是:Oracle 数据库、ES 集群、单个 ES;数据量较大情况下,检索速度由慢到快依次是:Oracle 数据库、单个 ES、ES 集群;海量数据下,ES 集群第 1 次检索的时间约为传统数据库检索的 1/20。2)从第 1 次和第 2 次检索时间得出,较传统数据库优越处是 ES 有缓存技

术,且 ES 集群第 2 次检索时间约为第 1 次检索的 1/4。总之,在检索速度方面,基于 ES 的数字图书馆检索系统很大程度上优越于基于数据库的检索系统。

4 结束语

本文在深入研究大数据存储、分布式检索原理和全面剖析 ES 相关技术的基础上,实现了基于 Elasticsearch 的数字图书馆检索系统,进行了检索速度测试,并与传统的 Oracle 数据库检索速度进行对比,得出较好的测试效果。不足之处,本系统对中文分词采用了通用的 ik 分词器,以及对检索结果只采用了简单的相关度排序算法。故在后续的开发中,将针对数字图书馆中文分词技术和结果排序算法进行进一步深入研究与改进。

参考文献:

[1] 王宏霞,艾树峰. 数字图书馆信息检索技术的研究[J]. 浙江传媒学院学报, 2007(4):69-71.

[2] 严武军,马小燕. 高校数字图书馆元数据检索系统的设计与实现[J]. 计算机工程与设计, 2006,27(1):162-164.

[3] 邹敏昊. 基于 Lucene 的 HBase 全文检索功能的设计与实现[D]. 南京:南京大学, 2013.

[4] Elasticsearch. ElasticSearch[EB/OL]. <http://www.elasticsearch.org/>, 2015-01-15.

[5] Huwei. 胡杰的专栏[EB/OL]. <http://blog.csdn.net/huwei2003/article/category/2655411>, 2015-02-02.

[6] IT 技术精华网. ElasticSearch 资料整理[EB/OL]. <http://www.chepoo.com/elasticsearcharticles>, 2015-02-02.

[7] Rafał Kuc, Marek Rogoziński. Mastering ElasticSearch[M]. Birmingham:Packt Publishing, 2013.

[8] 于天恩. Lucene 搜索引擎开发权威经典[M]. 北京:中国铁道出版社, 2013.

[9] 邱哲,符滔滔,王学松. 开发自己的搜索引擎——Lucene + Heritrix[M]. 2 版. 北京:人民邮电出版社, 2010.

[10] 罗刚. 解密搜索引擎技术实战:Lucene&Java 精华版[M]. 北京:电子工业出版社, 2011.

[11] Hatcher E, Gospodnetic O. Lucene in Action[M]. Manning Publications, 2005.

[12] Michael McCandless, Erik Hatcher, Otis Gospodnetic. Lucene 实战[M]. 牛长流,肖宇,译. 2 版. 北京:人民邮电出版社, 2011.

[13] 韩云辉. 基于 Lucene 的数字版权资源库的构建与应用研究[D]. 北京:北方工业大学, 2013.

[14] 张建中,钟慧敏. 基于云计算的图书馆云服务及其安全问题浅析[J]. 高校图书馆工作, 2012,32(2):67-69.

[15] 邹燕飞,于成尊,赵亮. 基于 Lucene 的文本搜索引擎的设计和实现[J]. 计算机与现代化, 2011(9):40-42.

[16] Zhao Wei. The design and research of Literary retrieval system based on Lucene[C]// 2011 International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT). 2011:4146-4148.