基于Lucene的中文倒排索引技术的研究

郑榕增,林世平
(福州大学 数学与计算机科学学院,福建 福州 350002)

摘 要:索引是所有搜索引擎的核心概念,为了进行快速查找,就需要把数据处理成一种高效的、可交叉引用的组织格式。倒排索引是一种高效的索引组织模式,其组织模式和存储结构对检索系统的性能起着至关重要的作用,文中探讨了基于Lucene的倒排索引模式,分析了Lucene索引文件的结构、索引过程以及相关排序算法,讨论了Lucene的压缩算法,并且通过设计一个中文模块实现了基于正向减字最大匹配分词方式的中文索引。实验表明新的分词算法比Lucene自带的分词算法性能有了很大的提高。

关键词:全文检索;倒排索引;索引压缩;Lucene
中图分类号:TP391.3 **文献标识码**:A **文章编号**:1673-629X(2010)03-0080-04

Research of Chinese Full Texts Inverted Index Based on Lucene

ZHENG Rong-zeng, LIN Shi-ping
(Department of Mathematics and Computer Science, Fuzhou University, Fuzhou 350002, China)

Abstract: Index is the core concept of all search engines. For quick search, the data need to be processed into a highly efficient, cross-references structure. Inverted index is a high-performance index model. Its organizational model and storage structure is crucial to the performance of search engine. In this paper, probe around the inverted index model based on Lucene, analyse the structure of Lucene index file, its index process, its compress algorithm and the relevant scoring rank algorithm. At last implemented Chinese-words segmentation module utilizing forwards maximum words subtraction match algorithm, the module work well to segment the Chinese words and deal with Chinese information efficiently. The experiment shows that the new algorithm performances better than the conventional one.

Key words: full-text retrieval; inverted index; index compression; Lucene

0 引 言

随着万维网的飞速发展,现代信息检索系统一般都要处理海量的数据。传统的手工检索方式已经难以适应这种需要,全文检索系统因为其检索功能强大,操作容易等特点受到越来越多用户的欢迎。

全文检索是指计算机索引程序通过扫描文章中的每一个词,对每一个词建立索引,当用户查询时,检索程序就根据事先建立好的索引进行查找,并将查找的结果反馈给用户的检索方式。全文检索的核心技术是将源文档中的所有的基本元素的出现信息记录到索引库中。

倒排索引(Inverted index),也常被称为反向索引,是一种以关键词作为索引关键字和链表访问入口的索引结构,用来存储在全文检索下某个关键词在一个文档或者一组文档中的存储位置的映射。它是文档检索系统中最常用的数据结构。通常把采用倒排索引方式组织的文件称为倒排文件[1](Inverted file)。倒排文件描述了一个词项(TREMS)集合元素和一个文档集合(DOCS)元素对应关系的数据结构,记:

$DOCS = \{D_1, D_2, \cdots, D_N\}$, $TREMS = \{T_1, T_2, \cdots, T_M\}$。

一个典型的倒排索引组织方式[2]是把每个关键词的倒排表数据按文档编号增序排列,压缩保存为整块数据,如图1所示。

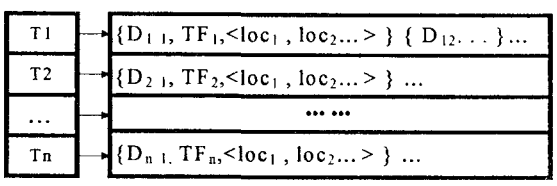

图1 倒排表

图中左侧为词典中的某个关键词 T_i,右侧为该关键词的倒排表内容,其中 TF_i 为关键词 T_i 出现在本文档的次数。D_{ij} 为文档编号,该词在文档内出现的位置序列为$<loc_1, loc_2, \cdots>$。

收稿日期:2009-06-12;**修回日期**:2009-09-07
作者简介:郑榕增(1982-),男,硕士研究生,研究方向为Web智能与信息检索;林世平,副教授,研究方向为数据挖掘。

1　相关研究

由于倒排索引在信息检索系统中的核心作用,人们在倒排文件索引技术上做了大量研究。文献[3]根据汉语词汇的频率分布情况和当前的软硬件环境,提出一种高效的倒排索引结构,在一定程度上节省磁盘空间,提高检索效率。文献[4]提出了一种分块组织倒排文件的方法。通过建立检索性能模型,进行分析和仿真实验,有效地减少检索执行时间。文献[5]提出了一种新的短语倒排索引技术,对关键词典里每个关键词,按其后续词来组织倒排数据项,即为每个关键词与其后续关键词建立辅助倒排索引。文献[6]提出了一种新的全文索引模型——后继数组模型,通过结合目前多个主流全文检索模型(倒排表模型、Pat 数组模型等)的优点,来提高空间效率和时间效率。

2　基于 Lucene 的倒排索引技术

Lucene[7]是一个开放源代码的全文检索引擎工具包,是一个全文检索引擎的架构,提供了完整的查询引擎和索引引擎。为了实现快速检索,Lucene 采用倒排索引的数据结构,用 Lucene 可以实现高效的全文检索模型,全文检索系统模型如图 2 所示。

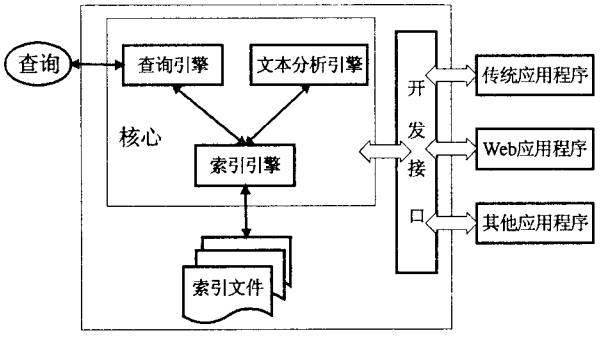

图 2　基于 Lucene 的全文检索模型

2.1　索引结构

为了实现快速检索,Lucene 采用倒排索引的数据结构。Lucene 的索引结构可以分为索引(index),索引段(segment),索引文档(document),索引域(field)和索引项(term)五个层次。

Lucene 的每个索引结构由一个或者多个段组成,每个段包含一个或者多个文档,每个文档又管理一个或者多个域,每个域由一个或多个索引项组成,而每个索引项就是一个索引数据。

2.2　索引过程

图 3 展示了 Lucene 索引过程的主要阶段。

Lucene 的索引过程可以分为三个部分:

1)预处理阶段,将数据转换成 Lucene 能够处理的格式——纯文本字符流。先提取出文本信息,然后利用这些提取出来的数据创建 Lucene 的 Document 对象及其对应的 Field 对象。

2)分析阶段,通过调用索引管理器(IndexWriter)的 addDocument(Document)方法将数据传递给 Lucene 进行索引操作。在对数据进行索引处理时,Lucene 会首先分析数据,使之更加适合被索引。

3)写入索引,对输入数据分析完成后,将结果写入索引文件中,将输入数据以倒排索引的数据结构进行存储。

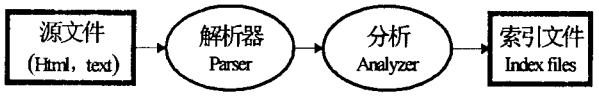

图 3　Lucene 的索引过程

以下为对三篇文章建立倒排索引的过程:

1 Lucy lives in Shanghai.

2 I live in shanghai too.

3 He once studied in Beijing.

经过 Lucene 的分析器(Analyzer)预处理后:

1 [Lucy][live][shanghai]

2 [I][live][shanghai]

3 [he][study][Beijing]

经索引后的倒排表如表 1 所示(关键词已经按字典排序)。

表 1　Lucene 倒排表

关键词 (term)	文章号 id [出现频率(tf)]	出现位置 (position)
he	3[1]	1
Beijing	3[1]	3
i	2[1]	1
live	1[1],2[1]	2,2
lucy	1[1]	1
shanghai	1[1],2[1]	3,3
study	3[1]	2

索引建立后,Lucene 通过 IndexReader 类来实现索引文档的删除、添加功能。

为提高索引的效率,索引可采用分组索引的方式,根据运行系统的内存大小,将索引分为 n 组,使得每组运算所需内存都小于系统所能提供的最大使用内存的大小,再按照倒排索引算法生成 n 组倒排索引。然后将这 n 组索引归并。

2.3　索引的压缩

由于搜索引擎要处理的都是海量的数据,其索引数据通常也都非常大的,所以有必要通过压缩索引来节约存储空间。对于倒排索引模式来说有两种途径来降低索引的大小:采用更高效的编码方式和压缩倒排索引。

编码改进的核心是通过更高效的编码方式来降低

平均码长。Lucene 通过一种可变字节编码的方式对其索引文件的整型数据进行压缩，该编码是一种变长编码，它是一种字节对齐的编码方式。它用不同字节数来表示不同范围的数，每个字节的最高位是标志位，如果最高位是 0，则表示其后续没有字节，如果最高位是 1，则表示其有后续字节。例如，128 可表示为 10000001,00000001。对于 32 位的整数，小于 128 的数可用 1 字节来表示，节省了 3/4 的空间；而大于 2097151 的数需要 5 字节，不但没达到压缩的目的，反而增大了空间。由于计算机多以字节为操作单位，字节对齐的编码往往更能发挥硬件的优势，所以可变字节编码的压缩和解压速度都较快。

为了减少索引空间，需要对倒排索引文件进行压缩，采用压缩技术，可以减少读取数据所需的 I/O 时间，从而提高检索速度。Lucene 采用了前缀压缩技术，压缩后存储的信息为＜前缀长度(Prefix. Length)，后缀(Suffix)＞，(1)前缀长度，该索引项和上一个索引项共同前缀的长度，其中第一个索引项的共同前缀的长度为 0；(2)该索引项的后缀。例如：当前词为“葡萄牙语”，上一个词为“葡萄牙”，那么“葡萄牙语”压缩为＜3,语＞。其次对数字压缩，数字只保存与上一个值的差值，减少数字的长度，从而减少保存该数字所需要的字节数。例如当前文章号是 16385(不压缩要用 3 个字节保存)，上一文章号是 16382，压缩后保存 3(只用一个字节)。

2.4 索引的检索模型

Lucene 关键词的查找基于分级查找机制。关键词按字符顺序排列，可以利用二元搜索算法快速定位关键词。假设要查询的单词是“中国”，Lucene 先对倒排关键词典二元查找，找到该关键词，通过指向频率文件的指针读出所有文章 ID 号，然后将结果按相关排序算法返回。

Lucene 的检索模型基于向量空间模型。先把查询和文档先表示成向量，通过计算文档向量 d 和查询向量 q 夹角的余弦值来衡量文档和查询之间的相似度。

$$\cos(q,d) = \frac{\sum_{t \in q \wedge d} (w_{q,t} \cdot w_{d,t})}{\sqrt{\sum_{t \in q} (w_{q,t})^2 \cdot \sum_{t \in d} (w_{d,t})^2}}$$

2.5 建立中文索引

中文分词是其他中文信息处理的基础，在中文信息搜索引擎中中文分词是实现可用系统的前提。

自动分词的基本方法有：基于字符串匹配的分词方法和基于统计的分词方法[8]。

基于字符串匹配的分词方法，这种方法又称为机械分词方法，它是按照一定的策略将待分析的汉字串与一个充分大的字典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功。按照匹配方向的不同，串匹配分词方法又可分为正向匹配和逆向匹配，按照不同长度优先匹配的情况，可以分为最大或最长匹配和最小或最短匹配。

Lucene 系统中默认使用 StandardAnalyzer 分析器处理中文文档。StandardAnalyzer 分析器支持 Unicode 双字节编码。StandardAnalyzer 中文分词相当于一元分词(1 - gram)，获得的结果是独立的单字索引。这种分词方式不会损失任何索引信息，但索引中会存在大量冗余的字符，难以过滤，造成的结果是索引膨胀比增大，检索的准确度不高。

文中采用的是正向减字最大匹配分词算法，该算法的基本思想：预先将文档预处理成一个纯文本格式。在待切分的文本 d 中，对每个句子 s 从左到右以 M 为最大词长界限选出候选字符串 W，如果 W 在词典中，处理下一个长为 M 的候选字段；否则将 W 最右边的一个字去掉，继续与词典相比较；s 切分完以后，构成词的字符串或者此时 W 已经成为单字，用分隔符隔开输给 $s2$，从 $s1$ 减去 W，继续处理后续的字符串。$s1$ 处理结束，取 d 中的下一个句子赋给 $s1$，重复前述步骤，直到将整个文本 d 都切分完毕。

假设，M = 8 个字节(即 4 个汉字)，$s1 = c1c2c3c4$，算法的描述如下：

(1) 取 $s1 = c1c2c3c4$，在中文词典中查找 $s1$ 看字符串 $s1$ 是否已经在词典中；

(2) 如果已经在词典中，则一次分词结束；

(3) 如果不在词典中，则减去最右边的字 $c4$，

即 $s1 = c1c2c3$，

(4) 将 $s1$ 继续与词典作比较，如果 $s1$ 仍未在词典中，则继续步骤(3)，直至分词结束或者 $s1$ 为单字汉字。

(5) 将分词结果输给 $s2$，$s1$ 处理完毕；

(6) 将 d 中的下一个句子赋给 $s1$；

(7) 重复步骤(1) ~ (6)，直到文本 d 切分完毕，将切分结果存入到倒排表中，以便检索使用。

中文分词程序的伪代码如下：

```
fileSegment(d){
  while(getsentenceoffile(d,s1))
    { s2 = chinesewordcut(s1);
    Output(s2);
    }
}
String chinesewordcut(s1){
  Preprocess(s1)
  While(! s1)
```

```
    { w=s1.substr(0,M);
    While(M.length>1)
      { if(w! =singelword)
      Then w--;
      }
    }
}
```

例如对“中华人民共和国警察”进行分词,假设 M=4,分词结果为“中华人民/共和国/警察”。

Lucene分析器的最主要功能是实现输入文档到索引项的转换。其中索引项是实现索引建立和查询的最基础的单元。Lucene 分析器全部由 analyzer 类派生而来,提供的调用接口使用也非常简单。分析器的基本构成是一个 Tokenizer 加上几个 Tokenfilter 协同组成。在 Analyzer 类方法 tokenSteam 中得以组合实现最终功能。Lucene 每个分析器都是内部一系列 Tokenize 和 Tokenfilter 组合作用的结果。使 Lucene 支持中文信息处理就是构造一个支持中文的 Analyzer。

将上面的正向减字最大匹配分词模块集成到 Lucene 中,构造一个支持中文的 Analyzer 类,实现 Lucene 的中文处理。最后可以用 Lucene 索引查看器 Luke 来查看网页的中文分词效果。图 4 是 Lucene 默认中文分词的结果,图 5 是结合了正向减字最大匹配的 Lucene 中文分词后的结果。从图中可以明显看到 Lucene 的分词准确度有了极大的提高。

Top ranking terms. (Right-click for more options)

No	Rank	Field	Text
7	59950	<contents>	为
8	59744	<contents>	中
9	58483	<contents>	人
10	58335	<contents>	了
11	57242	<contents>	上
12	55543	<contents>	大
13	55232	<contents>	这
14	54744	<contents>	个
15	53782	<contents>	以
16	52470	<contents>	和
17	51991	<contents>	出
18	49486	<contents>	国
19	49178	<contents>	来

图 4　Lucene 默认的中文分词结果

Top ranking terms. (Right-click for more options)

No	Rank	Field	Text
113	1828	<content>	时
114	1822	<content>	科技
115	1817	<content>	超级
116	1814	<content>	网络带宽
117	1796	<content>	dv
118	1774	<content>	房产
119	1773	<content>	浏览
120	1773	<content>	女性
121	1766	<content>	用户
122	1765	<content>	交流
123	1765	<content>	股票
124	1758	<content>	名

图 5　Lucene 基于正向减字最大匹配分词的结果

3　结束语

探讨了搜索引擎的倒排索引模式,分析了基于 Lucene 的全文索引方式,以及索引的压缩、存储方式,在此基础上,实现了一个基于正向减字最大匹配的中文索引方法。

参考文献:

[1] 李晓明,闫宏飞,王继民.搜索引擎——原理、技术和系统[M].北京:科学出版社,2006.

[2] 徐小刚,王俊杰,于　玉. 全文索引的研究[J].计算机工程,2002,28(2):101-103.

[3] 邓　攀,刘功申.一种高效的倒排索引存储结构[J].计算机工程与应用,2008,44(31):149-152.

[4] 彭　波,李晓明.搜索引擎倒排文件的一种分块组织技术[J].电子学报,2005,33(2):358-362.

[5] Williams H E, Zobel J, Anderson P. What's Next? -Index Structures for Efficient Phrase Querying[C]// In Proc. Australasian Database Conference. Auckland, New Zealand:[s.n.], 1999.

[6] 刘学文,陶晓鹏,于　玉,等,一种全新的全文索引模型——后继数组模型[J].软件学报,2002,13(1):149-158.

[7] Gospodnetic O, Hatcher E. Lucene in action[M].[s.l.]: Manning Publications,2004.

[8] 向　晖,郭一平,王　亮. 基于 lucene 的中文字典分词模块的设计与实现[J].信息检索技术,2006(8):46-50.

(上接第 79 页)

ing.[s.l.]:[s.n.], 1999.

[2] Fridrich J, Goljan M, Du R. Detecting Lsb Steganography in Color and Gray-scale Images[C]// IEEE Multimedia Special Issue on Security.[s.l.]:[s.n.], 2001:22-28.

[3] 张　涛, 平西建. 基于差分直方图实现 LSB 信息伪装的可靠检测[J]. 软件学报, 2004,15(1):151-158.

[4] 王国新, 平西建, 许漫坤,等. 一种利用相邻像素相关的隐写分析算法[J]. 信息工程大学学报, 2007,8(1):56-58.

[5] Sullivan K, Madhow U. Steganalysis for Markov Cover Data with Applications to Images[J]. IEEE transitions on information forensics and security, 2006,1(2):275-287.

[6] 盛　骤,谢式千,潘承毅.概率论与数理统计[M].第 3 版.北京:高等教育出版社,2001.

[7] Theodoridis S, Koutsoubos K. Pattern Recognition[M]. 3rd ed.北京:电子工业出版社, 2006.