有效支持全文本检索的 XML 索引技术研究

韩忠明,莫 倩
HAN Zhong-ming,MO Qian

北京工商大学 计算机学院,北京 100038
School of Computer Science & Technology,Beijing Technology & Business University,Beijing 100038,China
E-mail:hanzm@th.btbu.edu.cn

HAN Zhong-ming,MO Qian.Efficient index for XML full text queries.Computer Engineering and Applications,2007,43(28):169-172.

Abstract: Full text retrieval based on XML is the foundation of many related research problems,such as WEB information retrieval and information extraction.Furthermore,an efficiently XML index structure is very important to accelerate retrieval.An index structure based on the node labeling schema[1] is constructed and implemented in this paper,the index structure could effectively support full text retrieval queries.At last,the comprehensive experiments are conducted,the experiment results show the index structure is better on consumed time and storage.

Key words: XML;full text retrieval;index

摘 要:在 XML 文档上进行全文本检索已经成为很多研究课题的基础问题,例如 Web 信息检索,信息抽取等。有效的 XML 索引结构对于加速检索速度是至关重要的,在文献[1]的基础上全面地构建和实现了一个可以有效的支持 XML 全文本检索的索引结构。实验表明提出的索引结构在索引构建时间、空间等性能指标上均有很好的表现。

关键词:XML;全文本检索;索引
文章编号:1002-8331(2007)28-0169-04 **文献标识码**:A **中图分类号**:TP311

1 引言

在以 XML 为语言的网页以及富含文本信息的 XML 文档上进行全文本检索已经成为很多研究课题的基础。例如 WEB 挖掘、信息抽取以及垂直搜索引擎等。基于 XML 的全文本索引研究主要包含三个问题:(1)对查询节点的打分机制;(2)查询处理算法;(3)索引结构。其中索引结构识打分机制和查询处理算法的基础和关键。目前很多索引结构建立在节点的区域编码方法上,而这类编码方法有一些缺点,例如不包含路径信息、不支持基于位置的查询等。本文在节点编码模式[1]的基础上构建一个有效的索引结构来支持包含结构条件和全文本检索谓词的查询。

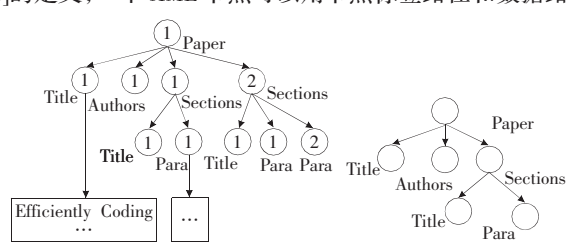
本文构造了一个混合式的索引结构(Hybrid Index Structure,HiD)。HiD 索引结构由结构索引部分和值索引部分组成。结构索引主要用于索引文档的结构信息,值索引用于对文档节点的值进行索引(有的文献也称为术语索引)。从实验结果分析可以得出本文提出的索引结构在主要的性能指标上均有很好的表现。

2 预备知识介绍

设 XML 文档是一个节点标签非循环树。树中的节点来自于一个集合 V,树的边来自于集合 E,标签是从集合 L 里面取值,这里 L 是一个字符串的集合。进一步,节点的文本值取自于一个文本字符串集合 T, 文本值可以附加在任何一个节点上,不仅是叶子节点。

处理树形结构 XML 文档需要对文档的节点和层次做出一些形式化的表示和提取,计算机理解了形式化的表示就意味着理解了对应 XML 文档的结构,这是需要对 XML 的节点进行编码。文献[1]利用节点的标签路径和数据路径构建了一个完整的 XML 节点编号模式,本文不再详细说明相关定义,用一个实例说明其编号模式。

图 1 给出了一个 XML 文档及其对应的摘要树, 根据文献[1]的定义,一个 XML 节点可以用节点标签路径和数据路径唯



(a)一个 XML 文档实例 (b)对应文档(a)的 XML 摘要树

图 1 一个 XML 文档实例和对应该文档的 XML 摘要树

基金项目:北京市教委科技发展计划资助(No.KM200610011002)。

作者简介:韩忠明,男,博士,讲师,主要研究方向:数据库与数据仓库,Web 信息处理,数据挖掘等;莫倩,男,博士,副教授,主要研究方向:Web 信息抽取,信息系统等。

一的确定。一个节点的标签路径为从根节点到该节点的路径上标签列表,例如节点 Para 的标签路径为:

Paper.Sections.Para

节点的数据路径为考虑到相同标签的节点位置列表,如属于第二个 Sections 节点的第二个 Para 节点的数据路径为:1.2.2。

由于标签路径和数据路径难以在计算机表达,故需将标签路径和数据路径转化为对应的标签路径数和数据路径数。首先将 XML 摘要树转化为一个二叉树,然后利用哈夫曼编码对节点的标签进行编码,生成节点标签路径数;对于节点的数据路径,则将数据路径转化为一个二进制位串,生成节点的数据路径数。反之利用节点的元数可以将一个数据路径数唯一的转化为一个位置串。通过一个节点的标签路径数和数据路径数就能唯一的确定一个 XML 文档中的一个节点。

3 索引结构

基于 XML 文档的全文本检索查询通常有两个特点:(1)查询请求通常具有结构条件;(2)查询请求同时具有全文本的检索条件。例如下面的查询:

例 1 从图 1 所表示的 XML 文档中找出所有标题类似于“efficiently Indexing XML Documents”,以及与“XML”、“Information retrieval”有关的论文。

这类检索查询请求的第一个特点决定了一个索引结构应该可以处理这些查询的结构条件,第二个特点决定了支持全文本检索的索引结构同时需要对 XML 文档的文本值建立索引。

3.1 索引结构

本文提出了一个 HiD 索引结构。这是一个混合的索引结构,组合了结构索引和全文本索引的优点。下面分析索引结构。图 2 显示了 HiD 索引的框架结构。

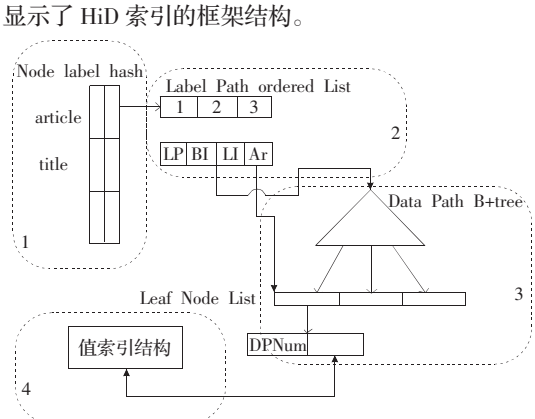


图 2 HiD 索引的框架结构示意图

在一个大的 XML 文档中,标签数量通常较少,因此首先对标签建立一个哈西索引。每一个标签可能有不同的标签路径,但是个数比较少,而且标签路径反映的是文档的结构信息,索引系统为每一个标签创建一个有序列表。在处理节点的数据路径时,本文使用 B+树索引技术,首先,在大型的文档中节点的数据路径数是很大的,需要建立一个有效的索引机制;第二数据路径数是大量的相对分部均匀的数字,B+树恰好可以有效得处理这种特性的数字索引。

从图 2 可以看出,HiD 索引结构包含四层架构。第一层是一个哈西表,用来处理文档中的所有节点标签。通过哈西表,每一个标签被映射到一个指针。指针指向一个有序表,这个有序表存放对应标签的节点标签路径数,并用这些标签路径数作为键值进行排序。这个是第二层的结构。最后一层,即第三层,是用来为节点的数据路径数建立索引的 B+树。B+树的键值是节点数据路径数。在有些查询问题中,系统需要遍历整个子树。为了可以快速的遍历子树,可以在 B+树的叶子节点上建立了一个双向链表。通过这个链表,系统可以快速的进行前后的遍历。

每一个有序表上的节点是一个四元组,四个成员有不同的含义和作用。第一个元素是标签路径数(LP),第二个元素是 B+树指针(BI),它指向对应的 B+树的根。第三个成员元素是叶子节点链表指针(LI),它指向 B+树的叶子节点链表。四元组的最后一个成员是节点的元素(arity)。用 Ar 表示。它可以在解析节点的数据路径数时使用。

B+树的叶子节点的组成结构是两个部分,节点的数据路径数(DPNum)和值索引指针(IX),值索引指针 IX 指向了值索引结构。这个值索引指针 IX 实际上连接了值索引和结构索引。值索引结构将在 3.2 节具体讨论。

HiD 索引在不同层次上有效地利用了不同的索引技术。哈西表是一种可以直接存取的索引方式。通过对标签哈西,可以直接定位到所有的标签和对应的标签路径。因为系统对标签的位置列表的索引技术采用了有序表而不是树形结构,所以查询效率和空间的利用率的到了平衡。相反,在大型的 XML 文档中,如 DBLP 和 XMark 文档,节点数据路径数的数量很大。采用 B+树可以对大数据量和分布均匀的数据进行快速查询。B+树的键值采用的是节点的数据路径数,而数据路径数包含节点的位置信息,所以 HiD 索引结构可以有效的支持基于位置的查询。

3.2 值索引结构

一个良好的索引机制可以支持不同的数据类型,此外,值索引还需要支持 IR 的查询需求。所以,构建一个值索引应该满足如下一些基本原则:

(1)易于和结构索引组合,以便进行查询;

(2)支持不同的数据类型;

(3)对不同的值索引实现一个一致的查询接口,这样可以提供给外部的系统一致的使用。

基于上述分析,可以实现两个类型的值索引:到排索引(Invert List)和数字索引(Number)。在两类索引上实现了一个一致的搜索函数:ValueSearch(VID,predication),其中 VID 是一个指针指向一个值索引,predication 是一个值判断谓词,来表达查询条件。这个函数返回一个满足值谓词查询的节点列表。

倒排列表(Invert list)索引。在传统的倒排文件中,关键词被存储在索引文件(index file)中(比如,按字母顺序存储),对于每个关键词,都有一个指针链表,该表中的每个指针指向与该关键词相关的某个文档,所有指针链表构成置入文件(posting file)。在建立倒排列表时,将所有节点的字符串值作为一系列的关键词处理,用基于字母顺序的方法建立一个哈西表作为索引文件,在关键词和包含这些关键词的节点,以及关键词出现在字符串中的位置之间建立一个映射。从而有效的完成倒排列表索引。

本文改进了传统的倒排索引,提出了三层值倒排索引结构,体系如图 3 所示。

下面简单地分析一下这个值索引结构的体系。最上层是对文档中出现的文本值进行 Hash,形成一个 Hash 表。这个 Hash 表除了文本信息和 Hash 表的地址外,还有一个 IX 值,IX 值表示和结构索引关联。Hash 表的下一层是文本术语发生的节点

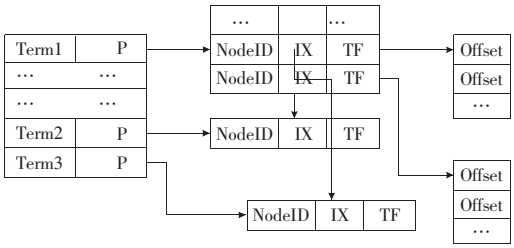


图 3　值索引结构的体系框架

列表。这个位置列表的每一项主要有两个元素,用来标明术语的发生位置。第一个元素是节点标识,第二个是术语在当前节点中发生的频度 TF 值。第三层是一个术语在当前节点中发生的位置列表。这个位置列表用术语在当前节点的文本值中所处的位置表示。

从图 3 上看出，这个倒排索引和传统的两层到排索引不同。传统的两层倒排索引相当于这里的前两层结构,术语表连接术语发生的位置列表。在处理 XML 查询时需要对节点的文本查询,所以需要建立一个节点上术语发生情况列表。在一个节点的文本字符串值上可能会有较多的单词,一个单独的术语也可能发生多次,所以需要建立一个在同一个节点下的术语发生位置列表,这就是第三层的意义。

倒排索引的目的是支持文档上的检索查询,那么索引结构上如何支持检索查询?可以在这个值索引结构上实现一个检索函数 FTABOUT,用来实现检索操作。这个函数输入参数是检索查询的检索关键词，输出的是检索词出现的节点标识和对应 TF 值,然后进行其它处理。

在本节的最后，简单地讨论一下索引结构的创建过程。HiD 索引结构的创建过程可以分为两个阶段。在第一阶段,解析输入的 XML 文档。在解析的过程中自动生成节点标签路径和数据路径。在这个阶段,关于节点的一些统计信息也会被自动收集,例如节点个数、文本值、数字值等。这些统计值在创建索引下一步的阶段需要使用。在第二个阶段,主要有两个核心处理过程。第一个处理过程是在第一个阶段标签路径、数据路径、统计量等信息的基础上生成节点标识。第一个处理过程对标签、标签路径、数据路径以及文本值和数字值等构建相应的索引结构。

4　相关实验及分析

为了测试和评估相关 HiD 索引架构的性能,本文用 C++(VC)语言实现了 HiD 索引架构。XML 的解析器使用了 Xerces SAX2 parser 解析器,这是一个由 IBM 公司提供的快速解析器,它不同于 DOM,而且生成路径是在解析的过程中自动完成(On-the-fly),所以采用 SAX2 解析器。对于 B+Tree 结构,使用了 Berkeley DB 提供的 B+Tree API 进行开发。实验运行在一台 CPU 主频为 P4 1.8 GHz 的 PC 机上。计算机的内存是 256 M,操作系统运行的是 Windows 2000 Server。为了可以有客观的分析和对比,选用的实验比较对象有三个:DataGuid[2]、ViST[3]、XISS[5]。

实验使用的数据集有两个。一个是公开的 XML databases DBLP,另外一个是 XML 基准数据库 XMARK。DBLP 是一个非常流行的计算机科学参考文献的数据库,它被广泛的应用在测试有关 XML 索引技术等研究上,它的特点是结构简单,深度不高,但是包含大量的文本信息。和 DBLP 不同,XMARK 数据集是一个单纪录的文档,它包含复杂的树形记录结构。实验选用了不同规模的几个数据集,表 1 显示了实验选用的这些数据集的特性,其中大小的单位为兆。

从表 1 所列出的实验数据可以看出,实验一共使用五个数据集。每个数据集的节点个数、大小、字符个数以及空格个数均有较大区别。从大小来看基本可以满足实际应用的需求。

表 1　实验数据集的不同特征

文档名称	节点数	字符数	空格数	大小
DBLP	1 906 219	11 660 704	61 485	46.6
DBLP	5 920 583	95 266 119	5 384 135	197.0
DBLP	6 391 621	103 717 843	5 817 551	209.0
Xmark	2 048 193	81 286 567	0	117.0
Xmark	9 621 573	398 304 178	0	500.0

实验首先对索引的构建时间进行分析,主要分析比较使用不同的索引技术来创建检索所需要的时间。然后再分析和比较不同索引技术对文档建立索引需要的空间情况。图 4 显示了不同索引技术创建索引索需要的时间比较结果。

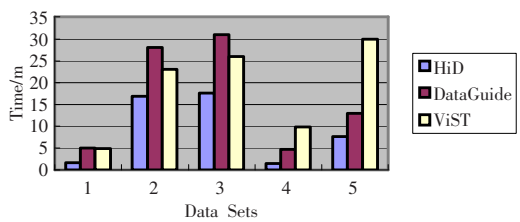


图 4　不同方法构建索引的时间需求

从图 4 可以看出,实验对五个不同特性的文档都建立了索引,并给出了不同方法在不同文档上的时间需求。从图 4 可以明显的看出 HiD 索引结构创建需要的时间在这些方法中是最少的。无论是结构复杂的 XMark 文档，还是结构相对简单的 DBLP 文档，都可以在较快的时间内创建完成 HiD 索引结构。与其他方法类似,HiD 索引结构创建需要的时间也与文档的大小有关系。但是,创建 HiD 索引在结构复杂时也不会有太大的时间变化。而 ViST 和 DataGuide 会因为文档的大小和结构的变化引起时间上的加大变化。节点索引的创建时间没有在图 5 中给出,因为创建节点索引时,部分索引工作是由手工完成的。所以可比较性上较差,没有在图 5 表示。

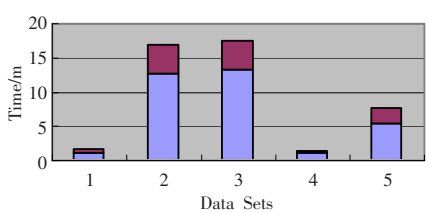


图 5　HiD 索引构建两个阶段的时间需求

因为 HiD 索引结构是分两个阶段进行，所以实验也比较了在两个阶段各自消耗的时间,实验结果显示在图 5 中。图 5 每个文档需要的时间条上的灰色部分表示在第一个阶段需要的时间,黑色的部分表示第二个阶段的消耗的时间。这样合起来正好是全部的时间。现在简单的分析一下这个实验结果。第一、五个文档的灰色部分都比黑色部分要大,表明第一阶段的时间要比第二阶段的时间消耗的更多;第二,随着文档的大小增加,第一阶段的时间也随着快速增加,但是,黑色的部分并没有显著得增加,这一方面表明了在第二个阶段需要的时间不会

的受到文档大小变化而产生太大影响;另一方面,第一阶段收集到得信息为第二个阶段提供了很大的基础,实验证明了通过改进两次解析文档,可以有效的改进建立索引的效率。

对于索引结构,还有一个关键的指标,就是索引空间的大小。索引空间越小,那么索引越容易置于高速内存中,这样越对查询有效。实验中对四个索引机制分别建立了索引,然后察看它们需要的空间。图 6 显示了四种索引机制在索引五个文档后分别需要的空间大小。从图 6 不难看出两个结论。第一,HiD 索引需要的空间在四种索引机制中是最少的;第二,其他三种索引机制在文档的结构和大小变化时,索引需要的空间会很快增加,而 HiD 索引机制不会有很大的变化。

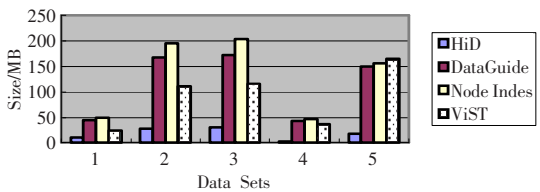


图 6 不同索引方法的空间需求

那么如何解释这两个结论呢? 首先,HiD 索引有效地集成了节点的标签,标签路径,数据路径。把结构信息全部用数字形式表示,这样大大的缩减了空间消耗,所以空间消耗最小。另一方面,如果文档大小增加了,但是结构没有太大变化,这样 HiD 索引机制可以有效的利用原有的路径信息,从而不会产生很大的变化。如果结构复杂度增加了,会给索引消耗的空间带来一些增长,但是由于节点值索引需要空间没有较大的增加。所以,总体空间没有很大的增加。

5 相关工作

信息检索领域的索引组织方法除了倒排索引外,组织索引文件也可以采用更复杂的方法,如:B 树,TRIE 树,Hash 表或者这些方法的变形或混合[10]。STAIRS[11]使用了二级索引文件。以相同字母对开始的词在二级索引中放在一起,一级索引中包含指向二级索引的指针,每个指针指向每个字母对。文献[9]重点讨论了结构索引和倒排索引的集成问题,并提出一个在倒排索引上进行打分的算法,然而文献[9]没有合理的利用节点的编号机制。

文献[6,7,8]的研究主要集中于如何在 XML 文档中进行全文本查询。文献[4]基于 Dewey 编码机制提出了一种新的倒排索引结构和相关的对基于 XML 的关键字查询的查询处理算法。其中的大部分研究结果表明结合结构和倒排索引的查询处理算法具有较好的性能。文献[13]综述了目前绝大多数的 XMl 索引技术。

文献[12]中提出了一个结合结构和语义信息的相关度打分算法,并给出了查询处理的算法。

6 结论与未来研究方向

在富含文本信息的 XML 文档上进行全文本检索已经成为国内外学术界一个非常热门的研究课题,其中索引机制、打分算法以及查询处理算法是重点研究问题。而有效的 XML 索引结构对于打分算法和加速检索速度是基础和关键,本文在文献[1]的基础上全面地构建和实现了一个可以有效的支持 XML 全文本检索的索引结构。这个索引结构集成了 XML 结构化信息以及文本信息,实现了结构索引和文本索引的有机合成。索引结构在不同层次上采用了不同的索引技术,使得索引可以较好的满足 XML 文档的特性。实验表明本文提出的索引结构在索引构建时间、空间等性能指标上均有很好的表现。

基于 XML 的全文本检索的打分机制与查询处理算法都依赖于索引结构,如何在本文提出的索引结构上构造有效的打分机制和查询处理算法都是非常值得研究的课题。另外,XML 主要用于网络上的数据表示,那么如何在大规模网络以及 P2P 网络上进行索引和查询也是一个需要研究的问题。

(收稿日期:2006 年 12 月)

参考文献:

[1] Han Zhongming,Xi CT,Le JJ.Efficiently coding and indexing XML document[C]//LNCS 3453:Proc of the 10th International Conf on Database Systems for Advanced Applications(DASFAA).Beijing:Springer-Verlag,2005:138-150.

[2] Roy Goldman,Jennifer Widom.Dataguides:enabling query formulation and optimization in semistructured databases[C]//Proc of the 23rd VLDB Conference Athens,Greece,1997.

[3] Wang Haixun,Park Sanghyun,Fan Wei,et al.ViST:a dynamic index method for querying XML data by tree structures[C]//SIGMOD,2003.

[4] Guo L L,Shao F,Botev C,et al.XRANK:ranked keyword search over XML documents[C]//Sigmod,2003.

[5] Li Q,Moon B.Indexing and querying XML data for regular path expressions[C]//Proc of the 27th VLDB,Roma,Italy,2001.

[6] Sacks-Davis R,Dao T,Thom J A,et al.Indexing documents for queries on structure,content and attributes[C]//Proc of International Symposium on Digital Media Information Base(DMIB),Nara,1997.

[7] Hugh E,Williams,Justin Zobel,et al.Fast phrase querying with multiple indexes[J].ACM Transactions on Information Systems,2004,22(4):573-594.

[8] Raghav Kaushik,Rajasekar Krishnamurthy,Naughton J F,et al.On the integration of structure indexes and inverted lists[C]//Sigmod,2004.

[9] Kaushik R,Krishnamurthy R,Naughton J F,et al.On the Integration of structure indexes and inverted lists[C]//SIGMOD,June 2004.

[10] Knuth D E.The art of computer programming[M]//Sorting and Searching.[S.l.]:Addison-Wesley,Reading,Mass,1973-03.

[11] IBM.IBM System/370(OS/VS),Storage and Information Retrieval System/Vertical Storage(STAIRS/VS).IBM World Trade Corporation.

[12] Han Zhongming,Le Jiajin,Shen Beijin.Effectively scoring for XML IR queries[C]//LNCS 4080:DEXA 2006,Springer,2006:12-22.

[13] 孔令波,唐世渭,杨冬青,等.XML 数据索引技术[J].软件学报,2005,16(12):2063-2079.