第36卷 第15期 计 算 机 工 程 2010年8月
Vol.36 No.15 Computer Engineering August 2010
·软件技术与数据库· 文章编号：1000—3428(2010)15—0049—02 文献标识码：A 中图分类号：N945

基于双路索引的XML查询优化研究

万 静，姜 蓉，易军凯
(北京化工大学信息科学与技术学院，北京 100029)

摘 要：为实现各种形式的XML数据查询，介绍一种双路索引方法，采用倒排序技术建立绝对索引模型和相对索引模型，并提出相关查询处理的算法。绝对索引模型将查询路径表达式缩短，减少比较次数，相对索引模型建立父子索引表补全路径，用较小的索引结构替代原始查询。采用DBLP数据集进行测试，实验结果表明，该方法可以提高查询处理的性能。

关键词：双路索引；倒排序；查询优化

Research on Double Index-based XML Query Optimization

WAN Jing, JIANG Rong, YI Jun-kai
(College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029)

【Abstract】 In order to support various queries for XML data, a double index method is proposed. Using the reverse order, an absolute index model and a relative one are built. A query algorithm is presented. The absolute index model reduces the number of comparison by shortening the path expressions. The relative one completes the path expressions by setting up parent-child index table and replaces original queries with small index structure. Experimental results to DBLP dataset show the method works well.

【Key words】 double index; reverse order; query optimization

1 概述

随着大量数据以XML格式保存，如何实现高效的查询成为当前研究的热点。针对XML数据建立有效索引结构，能够有效地改善查询效率，降低查询所需成本[1]。目前，多种支持XML数据查询的索引方法被提出：(1)基于节点的XML索引，如XISS[2]、XR-tree[3]、MPMGJN[4]等；(2)基于路径的XML索引，如DataGuide[5]、A(K)-index[6]、Ctree[7]、APEX[8]、Fabric[9]索引等。

本文基于以上索引方法提出一种双路索引(Double Index, DI)方法，并探讨DI的索引模型及相关查询处理算法，使其不仅能有效支持各种形式的路径表达查询，同时不会占用过大的空间。

2 DI索引模型

DI采用倒排序索引技术，兼顾绝对路径查询与相对路径查询，将规模很大的索引分成多个小规模索引，针对不同的查询应用不同的索引表。DI主要建立2个索引模型：绝对索引模型和相对索引模型。

2.1 模型相关概念定义

定义1(XML逻辑模型) $T=(V, root, E, type, val)$，其中，$V$是XML节点的集合，$V=V_{leaf}\cup V_{non\text{-}leaf}$，$V_{leaf}$为所有的叶子节点集合，$V_{non\text{-}leaf}$为所有的非叶子节点集合；$root$为根节点，$root\in V$；$E$是节点之间边的集合，二元关系$E\subset V^2$，若$u$是$v$的父节点或$v$是$u$的子节点，则$(u, v)\in E$；函数$type$: $V\rightarrow\{leaf, nonleaf\}$返回节点类型，若$v$为叶子节点，则$type(v)=leaf$，若$v$为非叶子节点，则$type(v)=nonleaf$；函数val: $V_{leaf}\rightarrow Str$返回叶子节点的值$value$，Str是XML文档中所有合法字符串的集合。

定义2(父节点集合与子节点集合) 给定T，定义除了叶子节点以外的其他所有节点组成的集合为父节点集合V_{parent}，即$(\forall v)(v\in V_{non\text{-}leaf}\rightarrow v$是父节点)；除了$root$以外的其他所有节点组成的集合为子节点集合$V_{child}$，即$(\forall v)(v\in\{V\text{-}root\}\rightarrow v$是子节点)。

定义3(次绝对路径) 给定T，定义次绝对路径p_i由一串以“/”相隔的连续的标记组成，它是绝对路径去除叶子节点后的路径。例如，绝对路径$v_{non\text{-}leaf1}/v_{non\text{-}leaf2}/\cdots/v_{non\text{-}leaf(i)}/v_{leaf(i)}$的次绝对路径是$v_{non\text{-}leaf1}/v_{non\text{-}leaf2}/\cdots/v_{non\text{-}leaf(i)}$，其中，$v_{non\text{-}leaf(i)}\in V_{non\text{-}leaf}$，$v_{leaf(i)}\in V_{leaf}$。$T$中的每条次绝对路径都根据其绝对路径在$T$中出现的顺序被赋予一个独有的序号$oid$。

定义4(前向路径与后向路径) 给定T，由V_{parent}中某一节点v_i与其子节点组成的路径称为前向路径，相反由V_{child}中某一节点v_i与其父节点组成的路径称为后向路径。

定义5(线索路径集与目标路径集) 给定T，在完成某一相对路径向绝对路径转化的过程中，可能需要分若干步，每一步对应于绝对路径表达式中的一个子表达式，每一步得到的结果路径集称为线索路径集，最终的路径称为目标路径集。

定义6(关键字集合) 关键字集合K是对T中合法字符集Str中的合法字符进行关键词分词后的不同关键词的集合。

2.2 绝对索引模型

绝对索引模型包含2个索引表：路径索引表(PI)和文本索引表(TI)。

对T中所有次绝对路径建立路径索引表PI。该索引表中

基金项目：国家“十一五”科技支撑计划基金资助项目(2006BAK31B04)

作者简介：万 静(1975－)，女，讲师、博士研究生，主研方向：智能信息系统；姜 蓉，硕士；易军凯，教授

收稿日期：2009-12-04 **E-mail：**yijk@mail.buct.edu.cn

的每一条记录是标识该次绝对路径的一个二元组(p_i, OID)。其中，p_i 是 T 的次绝对路径；OID 是次绝对路径分配的相应索引号集，$oid \in OID$。路径索引表 PI 的作用是在绝对路径查找中有效地搜索路径序号。

对所有叶子节点的值中出现的每一个相同的关键字建立一个文本索引表 TI。该索引表中的每一个记录是标识该关键字的一个二元组(key, Y)。其中，$key \in K$；Y 是由二元组(v_{leaf}, oid)组成的集合，v_{leaf} 是该关键字所在的叶子节点，$v_{leaf} \in V_{leaf}$，oid 是 v_{leaf} 所在绝对路径的次绝对路径的索引号。文本索引表 TI 的作用是在绝对路径的查找中有效地实现按关键字搜索，采用倒排序的存储结构，减少了索引数据冗余，以达到节约存储空间的目的。

绝对索引模型将查询路径表达式缩短，减少了比较的次数，大大节约了存储空间，从而减少需要的查询代价，提高了查询效率。

2.3 相对索引模型

相对索引模型包含一个索引表：父子索引表(PCI)。对 T 中的父子关系建立父子索引表 PCI。该索引表中的每一个记录是标识父子关系的一个二元组(Z, v_{child})。其中，Z 是由二元组($layer$, v_{parent})组成的集合，$layer$ 是父节点所在的层数，v_{parent} 是父节点，$v_{parent} \in V_{parent}$；$v_{child}$ 是子节点，$v_{child} \in V_{child}$。父子索引表 PCI 的作用是将相对路径或含通配符路径转化为绝对路径。为了解决 T 中不同层元素相同的问题，父子索引表引入了父元素所在的层数。

相对索引模型通过建立父子索引表补全路径，用较小的索引结构替代原始查询，使查询性能得到显著提升。

3 DI查询算法

DI 查询算法 Query 对查询路径进行分散处理，返回 DOC 集合。DOC 是 DI 查询算法返回的结果集，它的元素 $edoc$ 对 XML 文档的叶子节点的父节点进行创建。$edoc$ 包括遍历路径、子节点、叶子节点对应的文本值。DI 查询算法如下：

Query(v_i/…/v_j@$value$)

(1)给出查询路径表达式 P=(v_i/…/v_j@ $value$)，其中，v_i,…,v_j 为 XML 文档节点，$value$ 为叶子节点的值。

(2)如果(v_i/…/v_j@$value$)是绝对路径，则提取(v_i/…/v_j@$value$)的次绝对路径 p_i、叶子节点 v_i 和叶子节点值 $value$，查找 TI($value$, (v_i, PI(p_i)))，返回 DOC；否则，goto(3)。

(3)计算上一层线索路径集 $\bigcup_{i=1}^{n} path\_left_i$。其中，$v_{child\_left}$ 为相对路径最左边节点。v_{parent_left} 为 v_{child_left} 的父节点。$\bigcup_{i=1}^{n} path\_left_i$ 的计算公式为

$$\bigcup_{i=1}^{n} path\_left_i = \{\mathrm{PCI}(v_{child\_left}).v_{parent\_left} + / v_i / \ldots / v_j\}$$

循环查找 Query($path\_left_k$)，其中，$path\_left_k \in \bigcup_{i=1}^{n} path\_left_i$。

(4)计算下一层线索路径集 $\bigcup_{i=1}^{n} path\_right_i$，其中，$v_{parent\_right}$ 为相对路径最右边节点，v_{child_right} 为 v_{parent_right} 的子节点。$layer$ 为相对路径最右边节点所在层数。$\bigcup_{i=1}^{n} path\_right_i$ 的计算公式为

$$\bigcup_{i=1}^{n} path\_right_i = \{v_i / \ldots / v_j / + \mathrm{PCI}(layer, v_{parent\_right}).v_{child\_right}\}$$

循环查找 Query($path\_right_k$)，其中，$path\_right_k \in \bigcup_{i=1}^{n} path\_right_i$。

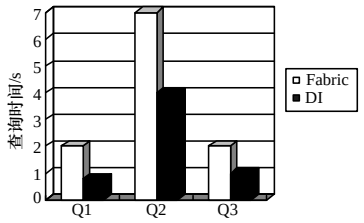
4 实验结果及分析

本文使用的数据集为 DBLP，选取 2 种不同大小的数据集，分别为 DBLP1, DBLP2，以检验数据量的大小对索引有效性的影响。表 1 给出了本文数据集的一些具体参数。

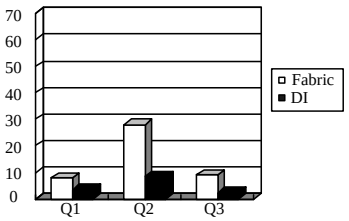
表 1 实验数据

数据集	数据集大小/MB	数据集节点数
DBLP1	15	450 345
DBLP2	138	5 842 321

实验比较了 DI 与 Fabric 索引，采用 3 种不同的查询，包括短查询集 Q1、简单长查询集 Q2 和复杂长查询集 Q3，用于比较查询的长度、复杂度等不同时，2 种索引在性能上的差异。实验以各个查询集中所有查询的平均查询执行时间为标准，结果如图 1 所示。测试平台为 1 GHz 处理器、512 MB 内存、Windows Server 2003、JDK1.5。DI 用 Java 实现。



(a)DBLP1 查询算法执行效率比较



(b)DBLP2 查询算法执行效率比较

图 1 实验结果比较

从图 1 可以得出，无论在何种查询中，DI 始终具有最小的平均查询执行时间，以上结果说明使用 DI 可以获得比 Fabric 索引更高的查询效率，原因在于相比于 Fabric 索引，DI 不仅建立了层次相关的索引，也建立了元素节点间的结构关系的索引，查询效率有明显提升。

5 结束语

本文提出一种 XML 索引模型 DI，它扩展了已有的 XML 索引方法，针对不同的查询应用不同的索引表，有效解决了相对路径和绝对路径的查询问题。实验结果表明，DI 是一种高效的索引方法，其利用相对较小的空间，实现较高的查询效率。

参考文献

[1] 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.

[2] Harding P. XISS/R: XML Indexing and Storage System Using RDBMS[C]//Proc. of the 29th Int'l Conf. on Very Large Data Bases. Berlin, Germany: [s. n.], 2003.

[3] Jiang Haifeng. XR-tree: Indexing XML Data for Efficient Structural Joins[C]//Proc. of the 19th Int'l Conf. on Data Engineering. Washington D. C., USA: [s. n.], 2003.

(下转第 54 页)