文章编号: 1000-5013(2011)01-0043-05

一种基于 XML Schema 的 XML 索引

郭艳艳, 吴扬扬

(华侨大学 计算机学院, 福建 泉州 362021)

摘要: 分析 XISS, DBXI, DDT 等索引方法的优缺点, 提出一种基于 XML Schema 的 XML 索引, 以提高 XML 查询的效率. 通过利用 XML Schema 结构信息对 XML 进行索引, 支持对基于不同 XML Schema 的多类 XML 文档的统一查询, 同时兼顾对 XML 文档的有效性验证和对无效查询的判断. 通过在编码时预留一定的编码空间, 方便对文档的更新. 本索引方案可有效地支持结构连接的计算, 以及对路径表达式的查询.

关键词: XML; XML Schema; 索引; 查询

中图分类号: TP 311.5 **文献标识码:** A

XML(eXtensible Markup Language)已成为一种网上数据交换和信息集成的工具. 由于 XML 的自描述性、易读性和开放性等优点, 使其受到网络开发者的青睐. 但随着 XML 数据的大量出现, 如何有效地实现对 XML 数据的存储、索引、管理, 成为目前研究的一大热点. XML Schema 作为一种描述 XML 文档的模式信息——结构信息的标准, 对于 XML 索引的建立及查询效率的提高有着重要的作用. 许多专家、学者在 XML 索引的构建上做了大量的研究和探索, 但大部分索引结构中都没有利用到 XML Schema 这一有效资源. W3C 提出的 XML Schema 描述了 XML 文档的结构信息, 它为 XML 文档结构、内容和数据类型建模提供了严格而完整的标准, 能对 XML 索引的建立及查询效率的提高产生影响. XML Schema 将 DTD(Document Type Definition)重新使用 XML 语言规范来定义, 并对 DTD 的缺点进行改进和扩充; 同时, Schema 本身也是一种 XML, 可以用现有的 XML 解析器解析[1-3]. 本文提出一个索引方案, 利用 XML Schema 实现基于不同 XML Schema 的多类 XML 文档的统一查询.

1 XML 索引方法的比较

XISS 是一种高效的 XML 索引查询方法[4], 主要索引结构有元素索引、属性索引和结构索引. 其主要思想是将复杂路径分解为简单路径, 然后对各简单路径的处理结果进行连接. XISS 对路径查询处理, 无须遍历 XML 文档, 但对于一个由 N 个元素/属性构成的查询路径要查找 N 组结点并进行 $N-1$ 次结构连接, 在这个过程中不可避免地会有许多不相关结构中的元素/属性参与结构连接.

DBXI 是利用 DTD 的信息提高路径查询的效率, 对 XML 文档及其遵循的 DTD 同时编码建立索引[1]. DBXI 能够精确定位 XML 文档中参与结构连接的结点集, 对于在待查 XML 文档中不存在匹配结构的路径查询, 能够在比现有 XML 索引方法在较短的时间内给出无查询结果的判断. DBXI 充分考虑了 XML 的模式信息, 不失为对 XML 索引技术研究的一种好的探索, 但 DTD 有其自身的局限性.

与 DTD(Document Type Definition)相比, XML Schema 有如下 3 点优势.

(1) DTD 仅支持自身的特殊语法, 它本身不是 XML 文档, 在解析处理时要对 DTD 文档单独处理. XML Schema 本身就是 XML 文档, 且具有 DTD 的功能, 也就是定义 XML 文档结构和语法的标准, 故可用相同的工具来进行解析处理, 为进一步应用带来方便.

(2) DTD 中仅对属性类型定义了几种非常有限的数据类型, 而 Schema 可以支持更多的数据类型,

收稿日期: 2009-08-11

通信作者: 吴扬扬(1957-), 女, 教授, 主要从事数据库和数据挖掘的研究. E-mail: wuyy@hqu.edu.cn.

基金项目: 福建省科技计划重点项目(2008I0021); 福建省自然科学基金计划资助项目(2009J01289)

并且允许用户自定义数据类型,具有更好的扩展性,灵活性.

(3) Schema 提供了一套更为规范、完整的机制,以约束 XML 文档中的置标的使用.如:能定义可以出现在文档中的元素、元素间的关系、元素的子元素、子元素的属性,以及元素出现的顺序和次数,等等.

与 DBXI 系统的类似,采用的索引方案也充分考虑了 XML 的模式信息.其不同之处在于:(1) 通过采用 Schema,避免因采用 DTD 所带来的不利因素,同时可以兼顾对 XML 文档的有效性验证和对无效查询的判断;(2) 支持多 XML Schema 的多 XML 文档,实现对不同类型的 XML 文档信息的查询;(3) 采用的编码方法可以有效地支持对索引的更新.

2 XML Schema 索引方案

2.1 基本定义

将 Schema 和 XML 都映射为文档树,通过对树的处理,获取所需的索引信息.

定义 1 文档树.一个 XML 文档可以表示成为一棵带标签的有序树 $T=(V,E,\Sigma)$. 其中:V 是结点集;$r\in V$ 为根结点且 $V=V_E\cup V_A$;V_E 是元素结点,V_A 是属性结点;$E\subseteq V\times V$ 是有向边集合,每条边 $e\in E$ 代表结点间的嵌套关系,例如 $e=(x,y)$表示结点 x 到结点 y 的边,称 x 是 y 的父亲,y 是 x 的孩子,孩子之间是有序的;Σ 是结点标签的有限集合;定义标签函数 label:$V\rightarrow\Sigma$,将树中的每个结点分别指派一个标签.以下用 label(x)表示返回结点 x 的标签.

2.2 编码方法

对于编码方法的设计,主要考虑有以下 3 点因素:(1) 被编码的数据结构应能支持祖先/后裔、双亲/孩子等关系的结构查询;(2) 编码算法的复杂度及编码后的查询执行时间;(3) 插入、删除等更新操作导致重新编码的代价.

很多 XML 索引方案都采用了 Zhang 编码[5],同时也对其进行了改进以支持对索引的更新. 文中的索引方案采用了 Sparse Numbering Schema 编码方法的思想[6],以更有效地支持索引的建立与更新.

定义 2 结点标签 lable.文档树的结点标签 lable 是一个五元组 (docID,start,end ,level,nodetype).“docID”是结点所属文档的 ID.“start”和“end”分别是结点在文档中的开始值和结束值,start 的值为在对文档进行先序扫描时,从文档的起始到该结点的开始的计算值再乘以一个放大系数 K,end 的值为从文档的起始到该结点的结束的计算值乘以一个放大系数 K.放大系数 K 为正整数,其值大小由文档的更新量及更新频率来确定,当更新量比较大、更新比较频繁时,放大系数 K 可以比较大.“level”为结点在文档中所嵌套的深度,即在文档树中的层次.“nodetype”为结点的类型,它的取值是 EE,ET,EM,EN,A,分别表示该结点的子结点内容是元素、内容是文本、内容是多种类型的结点的混合、内容为空的元素结点以及属性结点.

由编码方案的结构关系可得到:(1) 对于同一文档树 T 中的两个给定结点 u 和 v,u 是 v 的父亲结点当且仅当 start(u)<start(v)<end(u);(2) 对于同一文档树 T 中的两个给定结点 u 和 v,u 是 v 的左兄弟结点,当且仅当 start(u)<start(v)<end(u),end(u)>end(v),level(v)=level(u);(3) 对于同一文档树 T 中的两个给定结点 u 和 v,u 是 v 的祖先结点,当且仅当 start(u)<start(v)<end(u).

为了实现结构连接和方便查询,需要为 XML Schema 和 XML 文档结点分别填加了不同的信息.

(1) XML Schema 文档结点为(lable(x),parent (x)). 其中:lable(x)是该结点的结点标签,而 parent(x)是该结点的父结点的 start 值,对于根结点,parent(x)值为 0.如果需要实现向上查找结点的父结点,parent(x)可以有效支持这种查询.(2) XML 文档结点为(lable(x),start(x)). 其中:lable(x)是该结点的结点标签,start(x)是该结点所对应的 XML Schema 文档中的结点的 start 值.start(x)将 XML Schema 与 XML 文档结点有机的结合起来,实现了对结点的快速定位.

2.3 索引结构

索引由 XML Schema 索引和 XML 索引两部分构成.XML Schema 索引的目的是在 XML Schema 中快速定位指定名称的元素或属性.将具有相同元素/属性名的节点聚集在一起,以 XML Schema 文档 ID 分组,最终找到对应的 XML Schema 文档结点.该索引以倒排表方式组织,如图 1 所示.

XML 索引旨在通过 XML Schema 索引给出的线索,在各 XML 文档中定位所求的元素/属性的实

例. 如图 2 所示，XML 索引以倒排表的方式组织，通过如下多级索引实现.（1）第 1 级是将 XML 文档按 XML Schema ID 分组.（2）第 2 级在每一个分组中 XML Schema 结点的 start 值列表，指向 XML Schema 中结点对应的 XML 元素/属性实例集合.（3）第 3 级根据 XML Schema 给出的信息，将集合中的元素/属性实例按照文档的标识符 ID 分组，实现对同一文档中结点的快速提取. 记录在 XML Schema 文档树中某个元素/属性、同时属于同一 XML 文档、具有相同名字的 XML 元素属性实例.

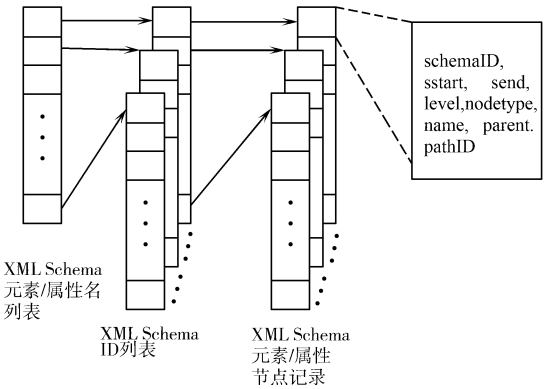

图 1　XML Schema 索引

Fig. 1　XML Schema index

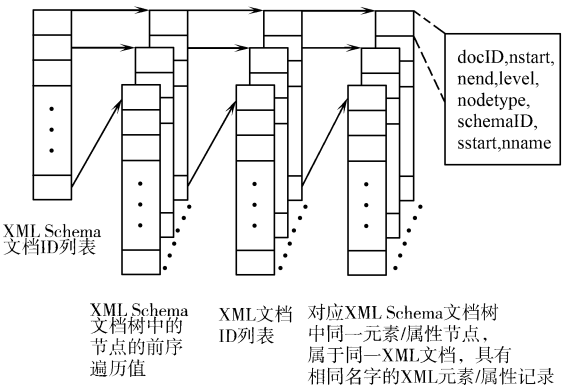

图 2　XML 文档索引

Fig. 2　XML documents index

2.4　索引存储方案

索引的关系存储模式如下：

Schemaindex(SchemaID, sstart, send, level, nodetype, name, parent, pathID);

Pathindex(pathID, SchemaID, path, name, start);

xmlindex(docID, nstart, nend, level, nodetype, SchemaID, sstart, nname);

value(docID, parent, type, nodevalue);

其中：黑体是表的主键. 假设所有列表都按码属性建立聚集索引.

（1）XML Schema 索引表 Schemaindex. 用于存放 XML Schema 索引. SchemaID 为该 Schema 文档标识；sstart，send 分别是该结点所对应的 start，end 值；level 是该结点在文档树中的层次；Nodetype 是该结点的类型；name 是结点的名称；parent 是该结点在 XML Schema 文档树中的父结点所对应的 start 值；PathID 为该结点在路径表中对应的 pathID. 在把 XML Schema 处理为文档树后进行编码，将所得到的 XML Schema 信息存储在该表. 按 XML Schema 索引结构，在各属性上建立索引，实现快速定位指定名称的元素或属性，有效地实现 XML 文档的结构查询，查询结果的文档片断重构，以及实现对查询信息的快速定位. 例如，对于一个的元素，查找它的所有孩子、后裔或属性结点等.

（2）路径索引表 Pathindex. 用来存储在 XML Schema 中每一条从根标记开始到另一个元素标记或属性名的路径表达式 path 和它的标识，并加入 SchemaID 和该结点的 start 和名称，用来实现对简单路径的匹配与查询.

（3）XML 文档索引表 xmlindex. 用于存放 XML 索引. docID 是 XML 文档的标识；nstart，nend 分别为该结点所对应的 start 和 end 值；level 为该结点在文档树中的层次；nodetype 是该结点的类型；SchemaID 为该 XML 文档所对应的 XML Schema 的 ID；sstart 为该结点对应的在 XML Schema 中对应的结点的 start 值；nname 为该结点的名称. XML 文档编码获取文档的索引信息后，存储在该表，根据 XML 索引结构，在该表的各属性上建立索引，在路径匹配完成后，查找到相应的元素/属性结点，从而实现了对 XML 文档的索引.

（4）值表 value. 用来记录 XML 文档的文本内容，包括所有属性结点的值和文本结点的内容，所有属性值和文本内容都以字符串的形式进行存储. Type 的值取为“*A*”或“*T*”；对于文本结点 parent 是该文本结点的父结点的 start 值，而属性结点的 parent 是该结点的 start 值.

2.5　索引的建立

索引的建立工作分两部分进行，即对 XML Schema 的处理，以及对 XML 文档的处理. 由于 XML

Schema 和 XML 文档采用了相似的编码方法和索引结构,如都采用了改进的 Zhang 编码[5],索引都以倒排表的方式进行组织,故对索引的建立采用了大致相同的过程.

通过以下 3 步可建立索引.(1) 要将 XML Schema 及 XML 文档进行解析,得到所需的文档树.(2) 通过对 DOM 树遍历,对其进行编码并获取相应的索引信息.(3) 将得到的索引信息组织为相应的索引结构,存储到相应的关系表. 如对 XML Schema,通过前两步的处理将所得到的索引信息组织为 XML Schema 索引,并存放到 Schemaindex 和 pathindex 两个表;而 XML 文档组织为倒排索引表,存放到 xmlindex 和 value 表. 由此可见,索引建立的重点就放在了对文档的遍历上. 由于对 XML Schema 和 XML 文档采取了相似的编码方法,他们的遍历方法也大致相同. 这里仅给出 XML 文档的遍历方法:

输入:XML 文档树的根结点

输出:编码后的具有 XML 索引信息的 XML 文档树

```
//初始化程序
初始化栈 s;
int n=0,l=0;
int XID=getxid();//获取 XML 的文档 ID
int SID=getsid();//获取 XML SchemaID
node p=root
do{
    if(p! =null){//若 p 不为空,将 p 进栈
      p. XMLID=XID;
      p. SchemaID=SID;
      ++n;
      p. sstart=;getstart(n);//为 p 赋于 start 值
      p. nodetype=getnodetype(p);//获取结点 p 的其他信息
      p. level=++l;
      p. sstart=getsstart(p);
      p. name=getnodename(p);
      p. value=getnodevalue(p);
//判断是否已访问过该结点及其下的所有子结点,如果访问结束将 p 置空,否则获取其
  下一个子结点
      if(! isover(p))
        p=getnextnode(p);
      else
        p=null;
}
else{//p 为空,将 p 出栈,设置其他信息
    p=s. pop();
    ++n;
    p. send=;getsend(n);
    level--;
}
}while(s 非空)
```

在这个算法中,利用栈,随着元素进出栈获得其信息值. 在元素进栈时设置其 start 值,并获得 xmlindex 的其他索引信息. 在元素出栈时设置其 end 值,并为访问下一个元素作准备.

2.6 查询

上述索引结构既可以支持路径法,也可以支持逐步结构连接法. 查询可以下通过两步实现.

(1) 用户输入查询路径或关键字．系统将路径与 Schema 文档树进行匹配，如果查询路径在 Schema 文档树中是不合法的，即 Schema 中没有与查询路径相匹配的结构，则可以肯定在相应的 XML 文档中也不存在与查询路径相匹配的结构．系统到此完成对路径查询的处理，返回“无查询结果”.

(2) 如果查询路径是合法的，也可以分为两种情况.(a) 简单绝对路径查询．如//book//title 的查询，直接采用路径法.首先在路径匹配操作成功后，找到所有满足条件的 XML Schema 结点(1,40,50,2,ET,10)，再通过 XML Schema 索引表所给出的信息——SchemaID，sstart 确定 XML 文档结点(1,40,50,2,ET,40).(b) 带有谓词的路径查询．如//book[@year=2000]/title 的查询，则将查询路径在 pathindex 上进行匹配，并返回叶子结点 year，title 及分支结点[1]book 的前序遍历值；然后，根据条件路径的叶子结点 Year 在 Schema 中的遍历值，从 value 中找出所有符合条件约束的 XML 结点集．其次，根据目标路径的叶子结点 title 在 Schema 中的前序遍历值，从 XML 值索引中找出所有符合条件约束的 XML 结点集；最后，找出所有符合条件的分支结点 book，调用相应的算法进行连接，返回查询结果.

3　结束语

利用 XML Schema 结构信息对 XML 进行索引，支持对基于不同 XML Schema 的多类 XML 文档的统一查询，同时兼顾对 XML 文档的有效性验证和对无效查询的判断.通过在编码时预留一定的编码空间，方便对文档的更新.本索引方案可有效地支持结构连接的计算，以及对路径表达式的查询.

参考文献：

[1] 路燕，张亮，段起阳，等.一种基于 DTD 的 XML 索引方法[J].计算机研究与发展，2005，42(1)：30-37.

[2] 许向阳，代卫宏，班鹏新.XNode：一种新的利用 RDBMS 来利用 RDBMS 来存储与检索 XML 的方法[J].计算机工程与应用，2004，40(16)：188-190.

[3] 万常选，刘云生，徐升华，等.基于区间编码的 XML 索引结构的有效结构连接[J].计算机学报，2005，28(1)：113-127.

[4] LI Quan-zhong，MOON Bong-ki. Indexing and querying XML data for regular path expressions[C]//Proceedings of the 27th International Conference on Very Large Data Bases. San Francisco：Morgan Kaufmann Publishers Inc，2001：361-370.

[5] ZHANG C，NAUGHTON J，DEWITT D，et al. On supporting containment queries in relational database management systems[J]. ACM SIGMOD Record，2001，30(2)：425-436.

[6] 郭松涛，朱征宇．一种新的 Sparse Numbering Schema 及其存储方法研究[J].计算机工程与应用，2004，40(27)：176-178.

An XML Index Based on XML Schema

GUO Yan-yan，WU Yang-yang

(College of Computer Science and Technology，Huaqiao University，Quanzhou 362021，China)

Abstract：In this paper，we analysis the advantages and disadvantages of XISS，DBXI and indexes based on DDT and propose an XML indexes based on XML schema to improve the efficiency of XML query. Our indexes can support searching many XML documents based on different XML schema. It take into account the validation of the XML document and determine the invalid query. We reserve some space when encoding. By this way，it is easy to updata the document. Our indexes can effectively support structure connection computing and path expression queries.

Keywords：XML；XML Schema；index；query

（**责任编辑**：钱筠　　**英文审校**：吴逢铁）