XML 正则路径表达式的优化技术研究

Optimizing Technique Research on XML Regular Path Expression

张慧敏　张春玲　孔鲁文　（大连大学土木工程技术研究与开发中心 辽宁大连 116622）

摘　要： 为提高 XML 数据的查询效率，本文基于 XML 数据模型和路径表达式提出一种新的优化方法，方法通过定义及解析一种 XML 树节点的编码，消除正则路径表达式中的闭包运算和通配符，将正则路径表达式简化为简单路径表达式，确定节点的查询路径，最终改善了闭包运算和通配符的不确定性而引起的查询效率低下问题，并验证了方法的优化性能。

关键词： XML XML 树节点编码 路径表达式 正则路径表达式优化 查询

XML（extensible markup language）是由 W3C 组织提出的一个 Internet 上数据表示和数据交换的新标准，其目标是定义计算机和人都方便识别的数据类型。随着 Web 技术的快速发展，XML 数据已大量的存在于当前信息社会中，如何高效地提取需要的信息成为 XML 数据管理的研究重点[1]。对 XML 的查询是提取 XML 数据实现对 XML 数据管理的重要方式之一，目前许多 XML 查询语言，包括 Xpath[2]，Xquery[3]，XML－GL[4]，Quilt[5]，Lorel[6]，XML－QL[7] 等都基于正则路径表达式 RPE（regular path expression），正则路径表达式的性能是 XML 查询效率的关键。

1　引言

优化正则路径表达式中用关系数据库管理 XML 数据的一个通用方法是[8]：首先根据 XML 文档的模式信息或者数据统计信息将 RPE 重写成若干个简单路径表达式（SPE）的集合；然后计算 SPE，再根据关系数据库模式翻译成 SQL 语句进行处理。目前有多种 SPE 路径表达式的查询策略，例如基于外延连接算法提出两个路径表达式查询优化策略：路径缩短策略和补路径策略[8]；将路径表达式表示成有限自动机，简化正则路径表达式去掉它的闭包运算和通配符[9]；采用一种逻辑优化方法研究了一种半结构化数据的查询语言－正则路径表达式的优化[10]；基于 DTD 和统计信息将正则路径表达式扩张为简单路径表达式，再将简单路径表达式重写为 SQL 语句[11]。以上这些方法都没有分析 XML 树节点位置的表示方法和路径表达式本身特征提出优化思想，本文基于一种唯一确定 XML 树节点位置的编码提出一种路径表达式简化策略。首先给出相关定义，它们是进行路径表达式查询处理的基础。然后详细讨论了路径简化策略。最后给出相关性能分析。

2　相关定义

2.1 XML 数学模型

将 XML 文档表示为一个抽象的树状节点的集合[1]，T＝（V，E，root，M），其中 V 是节点集合，E 是边的集合，边表示元素结点之间的嵌套包含关系，root 为根，M 是所有节点所带标签的集合。

树中有两个基本关系[9]—纵向的父子关系和横向的兄弟关系。这两个关系以及它们的逆关系分别可以用两个函数 Firstchild，Nextsibling 以及它们的逆函数 $Firstchild^{-1}$，$Nextsibling^{-1}$ 来表示。即

Firstchild：Node－＞Node. Firstchild 函数表示一个节点到它的第一个孩子节点的映射。

$Firstchild^{-1}$：Node－＞Node. $Firstchild^{-1}$ 逆函数表示一个节点到它父亲节点的映射。

Nextsibling：Node－＞Node. Nextsibling 函数表示一个节点到它下一个兄弟节点的映射。

$Nextsibling^{-1}$：Node－＞Node. $Nextsibling^{-1}$ 逆函数表示一个节点到它前一个兄弟节点的映射。

具体实例如图所示：图 1 为一个实际的 XML 文档，图 2 为满足 Xpath 树模型定义的 XML 树。

2.2 XML 路径及编码

定义 1 路径表达式的形式定义：

正则路径表达式 RPE(Regular Path Expression)是含有闭包、通配符等正则操作符的路径表达式，即包括//或 * 的路径表达式；

简单路径表达式 SPE(Simple Path Expression)是没有闭包和通配符的路径表达式，即不包括//或 * 的路径表达式；

//是纵向的闭包运算，可以在 XML 树的纵向层次上走任意层，又称闭包；

* 是横向的闭包，可以在 XML 横向层次上选择任何 XML 元素节点，又称通配符；

定义 2 XT – Code(XML 树节点编码：XML 树结点对应的字符序列)：

root 的 XT – Code 为 0；

任意非根结点的 XT – Code 为其父结点的 XT – Code &兄弟排行位置(1,2,...,n)，其中 &表示字符串连接符；

对任意给定结点 0x 的下一个兄弟结点为 0 &DtoC(CtoD(x) + 1)，其中 DtoC() 表示数字转换为字符函数，CtoD() 表示字符转换为数字函数；

对任意给定结点 0x 的第一个子结点为 0x &"1"；

给定结点 0x 的子孙结点 0y 满足 0y = 0x &t，t 为结点与子孙结点之间路径结点的兄弟排行位置连接成的字符串。

定义 3 路径表达式优化：

P：正则路径表达式，PS：简单路径表达式，XT – Code：XML 树结点编码

3 路径优化

3.1 优化原理

首先根据 2.2 节的定义将 2.1 节给出的例子中的 XML 树结点进行 XT – Code 编码，编码如图 2 中结点右边字符序列所示。根结点的编码为 0，其它的非根结点的编码都以 0 开始，所以在一个编码序列中遇到 0 代表一个新结点的编码开始。

例如，001012012201211 = > 0 01 012 0122 01211，代表结点序列 a b e l p，即路径 a / b / e / l / p

设计 XT – Code 编码的目的是简化正则路径表达式，去掉它的闭包运算和通配符。

XT-Code解析

P----------->PS

```
<?xml version="1.0" encodeing="UTF-8"?>
<Issue>
  <Articles>
    <Article>
      <Title>XML Data Management</Title>
      <Authors>
        <Author>Akmal B.Chaudhri</Author>
      </Authors>
    </Article>
    <Article>
      <Title>XML and Database</Title>
      <Authors>
        <Author>Bourret</Author>
      </Authors>
    </Article>
    <Article>
      <Title>On Building XML Data Warehouses </Title>
      <Authors>
        <Author>Laura Irina Rusu</Author>
      </Authors>
    </Article>
  </Articles>
  <Publisher>
    <Year>2000</Year>
    <Address>America</Address>
  </Publisher>
</Issue>
```

图 1 XML 文档示例

图 2 XML 文档树示例

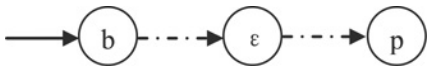
对于简单的路径表达式/a/b，可以看成 a 与 b 的连接，可以表示成：

如果是/a//b，它代表 a 元素下任何位置的 b 元素

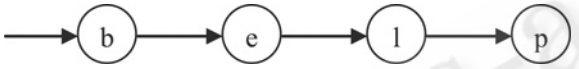
都满足要求,实际上可以看作/a(/*)*/b,可以表示成:

ε 代表途经的节点集合

根据图 2 中的 XML 树设一个正则路径表达式 RPE =/b//p,可以表示成:



经过优化,将其闭包和通配符去掉,对应的简单路径表达式 RPE =/b/e/l/p,可以表示成:



3.2 优化过程

3.2.1 消除闭包

定理 1 给定一个 RPE =/m//n,设 m 的 XT – Code 为 0x,n 的 XT – Code 为 0y,将 x 表示成 r 位,$x_1\ x_2 \ldots x_r$ 将 y 表示成 s 位,$y_1\ y_2 \ldots y_s$ 如果 n 是 m 的子孙,则 $x_1 = y_1, x_2 = y_2, \ldots, xr = yr$ 即截取 y 的前 r 位等于 x;

y 的前 r +1 位是 x 的第 y_{r+1} 个孩子,以此类推,就可以得到 m 到 n 的路径表达式。

通过对 y 的分析确定路径表达式,消除闭包。

例如,根据图 2 中的 XML 树设 RPE =/b//p

b 的 XT – Code 编码为 01

p 的 XT – Code 编码为 01221

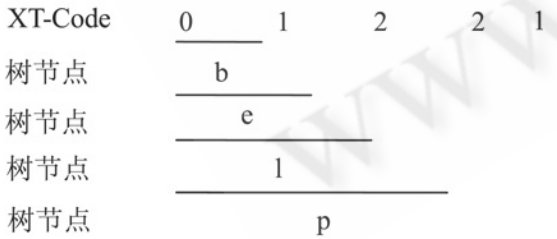


图 3 XT – Code 编码转化为路径表达式的过程示例

编码转化为路径表达式的过程如图 3 所示,消除闭包后的简单路径表达式为 SPE =/b/e/l/p,于是就可以消除闭包带来的不确定性,确定一条查询路径。

3.2.2 消除通配符

定理 2 给定一个 RPE =/m/*,设 m 的 XT – Code 为 0x,将 x 表示成 r 位,$x_1\ x_2 \ldots x_r$ 则

前面以 0 $x_1\ x_2 \ldots x_r$ 开始的编码对应的节点都是要搜索的节点。

例如,根据图 2 中的 XML 树,设 RPE =/b/*,b 的 XT – Code 编码为 01,那么 011,012,013 等编码以 01 开始的节点都是要搜索的节点。消除通配符的简单路径表达式为 SPE =/b/d,/b/e,/b/f,可以确定通配符代表的路径集合。

设 RPE =/b/*//1,那么 01121 就是符合路径表达式的节点,即节点 o. 消除通配符的简单路径表达式为 SPE =/b/d/j/o。

4 性能评价

根据上文提出的路径优化方法,对 XML 文档树进行深度优先遍历来进行编码,时间复杂度为 O(n +e),n 为树中节点的个数,e 为树中边的个数。编码的查询与分析过程时间复杂度为 O(n) 代价相对较小。为了测试此技术的性能,设计了两个查询,如表 1 所示,从表中可以看出此方法的性能是显著的。

表 1 优化方法性能测试查询

原查询	优化查询	减少的查询
a/b/d/i;a/b/d/j/o;a/b/e/k;a/b/e/l/p	a/b/e/l/p	a/b/d/i;a/b/d/j/o;a/b/e/k
a/b/d/i;a/b/d/j/o;a/b/e/k;a/b/e/l/p;a/b/f/m;a/b/f/n/q	a/b/d/j/o	a/b/d/i;a/b/e/k;a/b/e/l/p;a/b/f/m;a/b/f/n/q

5 结束语

本文针对正则路径表达式的 XML 查询如何进行优化问题,设计了一种 XML 树节点的编码方法,通过对节点编码的解析将正规路径表达式简化为简单路径表达式。相对于传统的优化算法,该方法分析了节点的位置与路径表达式特征之间的关系,根据关系对于路径表达式进行变换。利用编码来优化路径表达式,扩展了查询优化方法的研究范围。测试结果说明了它的可行性和有效性。

(下转第 35 页)

(上接第 38 页)

参考文献

1 Kong LB, Tang SW, Yang DQ, Wang TJ, Gao J. Querying techniques for XML data. Journal of Software, 2007,18(6):1400 - 1418.

2 Clark J, DeRose S. XML Path Language (XPath) Version 1.0 W3C Recommendation. World Wide Web Consortium, 1999. http://www.w3.org/TR/xpath.

3 Chamberlin D. XQuery: A query language for XML W3C working draft. Technical Report, WD - xquery - 20010215, World Wide Web Consortium, 2001. http://www.w3.org/TR/xquery/.

4 Ceri S, Comai S, Damiani E, Fraternali P, Paraboschi S, Tanca L. XML - GL: A graphical language for querying and restructuring XML documents. Computer Networks, 1999, 31(11 - 16):1171 - 1187.

5 Chamberlin D, Robie J, Florescu D. Quilt: An XML query language for heterogeneous data sources. In: Suciu D, Vossen G, eds. Proc. of the Int'l Workshop on the Web and Databases (WebDB 2000). Dallas: Springer - Verlag, 2000.1 - 25.

6 Abiteboul S, Quass D, McHugh J, Widom J, Wiener J. The Lorel query language for semistructured data. Int'l Journal on Diital Libraries, 1997,1(1):68 - 88.

7 Deutsch A, Fernandez M, Florescu D, Levy A, Suciu D. A query language for XML. Computer Networks, 1999,31(11 - 16): 1155 - 1169.

8 Lü JH, Wang GR, Yu G. Optimizing path expression queries of XML data. Journal of Software, 2003, 14(9):1615 - 1620.

9 Zhu MS. Path Expressions's Oprimization and Its Evaluation Techniques for XML Query and Filtering [Eng. D. Thesis]. Shanghai Institute of Computing Technology Chinese Academy of Sciences. Graduate School of Chinese Academy of Sciences, 2003.

10 M. Fernandez, D. Suciu, Optimizing Regular Path Expressions Using Graph Schemas, Proceedings of the 14th ICDE Conference, Orlando. Florida, February 1998.

11 Zheng Gang. The Query of Currency Path Expression Based on XML Data. Microcomputer Development, 2004,11(14):94 - 97.