一种改进 XML 模式树查询最小化算法①

魏东平，吴玉雁，朱新向

(中国石油大学(华东) 计算机与通信工程学院，青岛 266555)

摘 要：对 XML 模式树查询进行最小化查询优化，是左右 XML 数据查询处理性能的关键因素。对模式树查询最小化技术进行了研究，结合 XML 结构索引提出了一种改进的 XML 模式树查询最小化算法。该算法有效地解决了普遍意义上的语法层次优化中不能有效删除冗余节点的缺陷，从而提高了查询效率。实验结果表明该算法是正确的和有效的。

关键词：模式树查询；PTQ；最小化；结构索引

Improved Algorithm for Minimizing Pattern Tree Queries of XML

WEI Dong-Ping, WU Yu-Yan, ZHU Xin-Xiang

(Institute of Computer and Communication Engineering, China University of Petroleum, Qingdao 266555, China)

Abstract: At present, Minimization of Pattern Tree Queries, becomes the key factor that affects query processing performance of XML data. An improved algorithm is proposed with structural index of XML, based on analyzing the existing methods of Minimization of Pattern Tree Queries. It's effective to solve that grammar level optimization can't effectively remove redundant nodes of the defect in the common sense, so as to improve the efficiency of the inquiry. Experimental results show the effectiveness and accuracy of the proposed minimization algorithm.

Key words: pattern tree queries; PTQ; minimization; structural index

目前许多 XML 查询语言如 XPath、XQuery、XML-QL 等有一个共同特点，即使用模式树来表达查询需求，将模式树与 XML 文档树相匹配来得到所需要的、与模式树相吻合的查询结果。它的效率在很大程度上取决于 XML 模式树的大小，因此如何快速地找出并删除查询模式树中的冗余节点就变得十分重要，最小化模式树查询 PTQ[1,2]（Pattern tree query）成为了 XML 查询优化研究的重点。

现有的 XML PTQ 最小化方法包括两个方面，无约束的 PTQ 最小化和带约束的 PTQ 最小化[3]，前者以判断节点之间的包含映射关系为基础，分析路径等价关系，从而不断地修剪模式树中的冗余节点或分支。然而现有的方法大都在一定的限定条件下进行最小化，而普遍意义的[4]PTQ 最小化是一个 NP 完全问题。对于后者，现有的优化方法主要基于改进 Chase[5]方法的思想，研究的重点主要集中在如何抽取更多的完整性约束条件。然而现有的方法不能从 XML 文档中获取其隐含的全部结构完整性约束，从而导致了优化不彻底[6]。

对于以上问题，本文针对模式树本身固有的冗余，结合 XML 结构索引，基于最小线性路径概念，采用自顶向下和自底向上相结合的方法对 PTQ 进行最小化优化，不仅有效的删除了分支冗余，而且对冗余节点也进行了很好的判断和删除。同时，对于存在完整性约束条件下的模式树冗余问题，可以对该方法进行扩展，将完整性约束条件应用于冗余分支的判断中，与子路径方法相结合删除冗余路径。

1 有关概念和术语

1.1 XML 文档模型

通常将一个 XML 文档的模型看成是一棵有根、

① 收稿时间:2011-07-20;收到修改稿时间:2011-08-30

有序、带标记的树，其中的结点表示文档中的元素、属性和文本结点。每个结点都对应一个唯一的标签，边表示结点之间的嵌套关系，如图 1 所示。

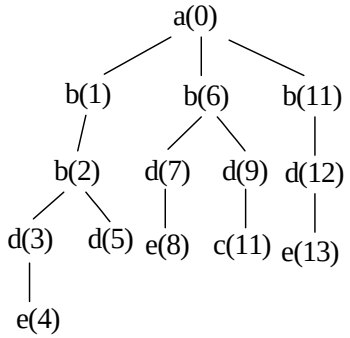

图 1 XML 文档树模型

1.2 查询模式树

一个树模式查询 p 是<tp，op>，其中，tp=（rp，Np，Ep，λp）是一棵树；Ep 被分为三个互不相交的集合 Cp、Dp 和 DSp，分别表示所有孩子边(c-edge)、后裔边(d-edge)以及后裔或自身边(ds-edge)。op 是唯一的输出结点，对应于树模式查询 p 的输出。如图 2 所示分别为 Xpath 表达式：/a//b[//c][d/c]/d/e 和 /a//b[d/c]/d/e 所对应的查询模式树，*所指的节点为输出节点。

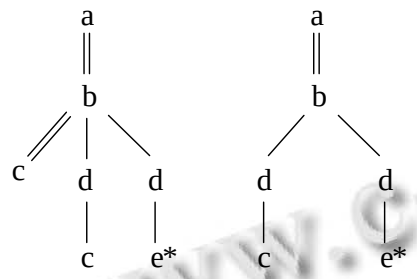

图 2 (a)模式树 Q (b)剪枝后的 Q′

1.3 结构索引 JoinGuide

JoinGuide[7]索引是原 XML 文档的结构索引，它以 XML 树结构中节点的路径信息为基础，采用某种简约方式，使得约简后的树结构只维护不同的路径信息，而不会存在具有相同路径的两个节点。因此它不仅简化了原 XML 文档重复、复杂的结构，而且较好的保留了原 XML 文档的节点信息。

一个 XML 结点的路径为从根开始到该结点所经过结点的标签序列 l1l2…lk。将具有相同路径的 XML 结点归入一个等价类。一个索引结点对应一个等价类，且该索引结点在 JoinGuide 树中具有与该等价类相同的路径。图 3 为图 1 的 JoinGuide 索引。

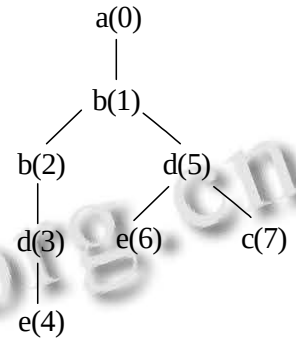

图 3 图 1 对应的 JoinGuide 索引

JoinGuide 索引使 XML 数据树中出现的每种路径在 JoinGuide 中都会唯一出现一次，并且对应唯一一个索引结点，JoinGuide 中出现的路径都会出现在 XML 文档中。我们使用 DG 表来记录 JoinGuide 索引中的节点路径。如表 1 所示，DG 表表示为 DG(PTD，Path，PLabel，Length)。其中，字段 PTD 为 JoinGuide 中每个节点的主键；字段 Path 每个节点对应的从根到该节点的路径，由“/”连接；PLabel 为 Path 对应的节点的 PTD；Length 为 Path 路径中节点的个数。

表 1 DG 表

PTD	Path	PLabel	Length
0	/a	0	1
1	/a/b	0.1	2
2	/a/b/b	0.1.2	3
3	/a/b/b/d	0.1.2.3	4
4	/a/b/b/d/e	0.1.2.3.4	5
5	/a/b/d	0.1.5	3
6	/a/b/d/e	0.1.5.6	4
7	/a/b/d/c	0.1.5.7	4

2 基于结构索引的最小化方法

2.1 基本概念

定义 1.[8] ∀P1,P2∈LP（线性路径表达式），Np1 和 Np2 分别表示 P1 和 P2 路径表达式上的节点集，P2 是 P1 的子路径，记作 P2≤P1，当且仅当存在一个映射 f：Np2→Np1 满足如下条件：

1) 保持节点类型，即∀v∈Np2，若 lab(v) = e，则 lab(f(v)) =e。

2) 保持边关系，即∀u,v∈Np2，若(u,v)是孩子边，则在 P1 中，f(v)是 f(u)的孩子节点；若(u,v)是后代边，则在 P1 中，f(v)是 f(u)的后代节点。

例如，对于图 2 所示的模式树，线性路径表达式“/a//b//c”是“/a//b/d/c”的子路径。

定义 2.[9] ∀P1,P2∈LP 和 XML 文档 D 对应的结构索引 G，假如 P1 与 P2 在 G 上的匹配路径集是相同的，则 P1 与 P2 在 D 上是等价的，记作 P1≡P2。

定义 3. ∀p∈LP，p 的最小线性路径记作 p_{min} ，当且仅当 p_{min} 满足 p_{min}≡p，且不存在 p′≡p 使| p′| < | p_{min} |，其中| p |表示线性路径 p 上的节点个数。

定义 3 给出了最小线性路径概念，本文采用文献[9]提出的自底向上获取最小线性路径的方法。

2.2 改进算法的基本思路

基于最小线性路径概念，采用自顶向下和自底向上相结合的方法进行 XML 模式树查询最小化优化。基本思想是：阶段 1，使用子路径的概念对 XML 查询模式树进行剪枝，删除冗余分支，如图 2(a) 所示的模式树剪枝后如图(b) 所示；阶段 2，首先对剪枝后的查询模式树采用分治方法分解成若干线性路径表达式，如图 2(b) 所示的树模式分解为 3 个线性路径表达式：/a//b、//b/c/d 和//b/d/e；然后分别对这些线性路径表达式采用最小线性路径概念进行最小化，最后对最小化后的线性路径重构模式树，最小化后的模式树如图 4 所示。

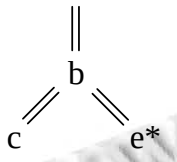

图 4　最小化的模式树 Qmin

2.3 算法描述

输入：XML 文档的结构索引 G，模式树查询 Q。

输出：最小化后的树模式查询 Qmin。

Step1：使用子路径删除冗余分支

optimizeSubPath(Q)；/* optimizeSubPath(Q)得到剪枝后的 Q’，算法略*/

optimizeSubPath(Q)算法在无约束条件下进行冗余路径判断，当 XML 文档中存在完整性约束时，可以对该算法进行扩展，将完整性约束条件应用于冗余分支的判断中，与子路径结合删除冗余路径。

Step2：结合结构索引对剪枝后的 PTQ 进行最小化优化

算法 1. 自顶向下分解线性路径表达式

```
recursionToLine(root)
  linePath ← root;
   if(root.type = branch) then
    for each eChild ∈e.descendant
     recursionToLine(eChild);
    if(e.type=leaf)then /*对于叶节点调用算法 2 进行最小化路径优化*/
     optimizeLinePath(linePath,linePath.size);
    else /*对于分支节点采用递归的方法将模式树分解为线性路径表达式*/
     e.descendant → eChild
     recursionToLine(eChild);
```

算法 2. 自底向上进行最小线性路径优化过程

```
optimizeLinePath(linePath,linePath.size)
shortest ← queryFromDataGuide(linePath);
  candidateList ← getCandidatePath(shortest);
  for each si ∈candidateList
    Boolean judge ← judgeEquivalent(si);
      if(judge) then
      return(si);
```

3　实验结果及分析

实验环境为 Intel CoreTM2 2.00GHz，2GB 内存，160GB 硬盘，操作系统为 Windows XP Professinal，Java 编程语言。实验数据来自经典数据集 XMark，大小为 113MB，其中含有 1666315 个元素节点，其 JoinGuide 中节点个数为 514 个。

表 2　Original query 与 Minimal query 的对比表

ID	Original query	Minimal query	Min Time
Q1	//open_auctions/open_auction/[bidder]/reserve	//open_auction[bidder]/reserve	8.834ms
Q2	/site[//person/name]/regions//name	/site[//person/name]//europe//name	19.953ms
Q3	/site/people/person[address][address/province]/name	//person[//province]/name	11.894ms
Q4	//person[name]/address	//person[name]/address	0.262ms
Q5	/site//closed_auction[buyer]/seller	//closed_auction[buyer]/seller	8.982ms

表 2 显示了实验所使用的原查询路径(Original query)和最小化后的查询路径(Minimal query)的对比，并给出了最小化所需要的时间。图 5 直观的对比了最小化前后的查询响应时间收益，从整体上看，最小化

后的查询时间比原查询所用时间缩短了 20%左右。与图 5 所示的查询时间相比，最小化所需时间较小。因此，本文的最小化方法是有效的。

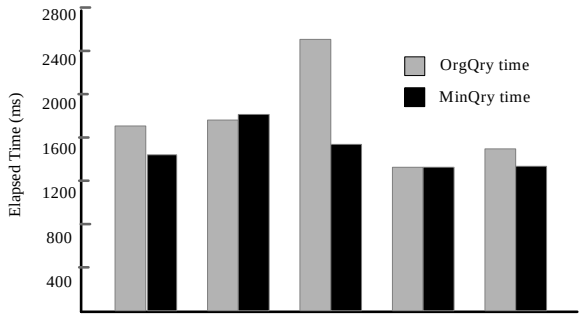

图 5 Original query 与 Minimal query 的查询时间

表 3 改进的 Chase 方法与基于结构索引的方法的最小化结果

ID	改进Chase方法	基于结构索引的方法
Q1	//open_auctions/open_auction/[bidder]/reserve	//open_auction[bidder]/reserve
Q2	/site[//person/name]/regions//name	/site[//person/name]//europe//name
Q3	/site/people/person[address][address/province]/name	//person[//province]/name
Q4	//person/address	//person/address
Q5	/site//closed_auction/seller	//closed_auction/seller

采用文献[8]给出的 XMark 的 XSICs，表 3 给出了采用改进的 Chase 方法与本文提出的基于结构索引的方法最小化后的结果,通过表中数据的对比，明显的看出基于结构索引的方法的优化比较彻底。

4 结语

本文结合 XML 结构索引，采用自顶向下与自底向上相结合的方法对 XML 模式树查询进行最小化优化，有效地解决了普遍意义上的语法层次优化中不能有效删除冗余节点的缺陷，提高了查询效率。

参考文献

1 Amer-Yahis S, Cho S, Lakshmanan LV, Srivastava D. Minimization of tree pattern queries. In: Aref WG, ed. Proc.of the SIGMOD 2001 Electronic. Santa Barbara: ACM Press, 2001. 497–508.

2 Ramanan P. Efficient algorithms for minimizing tree pattern queries. In: Franklin MJ, Moon B, Ailamaki A, eds. Proc. of the 2002 ACM SIGMOD Int'l Conf. on Management of Data. Madison: ACM Press, 2002. 299–309.

3 万常选.XML 数据库技术.北京:清华大学出版社,2005:202–224.

4 Flesca S, Furfaro F, Masciari E. On the minimization for Xpath queries. In: Freytag JC, Lockemann PC, Abiteboul S, Carey MJ, Selinger PG, Heuer A, eds. VLDB 2003, Proc.of the 29th Int'l Conf. on Very Large Data Bases. Berlin: Morgan Kaufmann Publishers, 2003. 153–164.

5 Wood PT, Minimizing simple XPath expressions. In: Mecca G, Siméon J, eds. Proc. of the 4th Int'l Workshop on the Web and Databases, WebDB 2001. Santa Barbara: ACM Press, 2001.13–18.

6 孟小峰,王宇,王小锋.XML 查询优化研究.软件学报,2006, 17(10): 2069–2086.

7 乔健,陈彤兵,汪卫,等.一种基于结构索引的 XML 模式匹配方法.计算机科学,2005,32(10):95–99.

8 张剑妹,陶世群,梁吉业.XML 结构完整性约束下的路径表达式的最小化.软件学报,2009,20(11):2977–2987.

9 Lee KH, Whang KY, Han WS. XMin: Minimizing tree pattern queries with minimality guarantee. World Wide Web, 2010,13:343–371.