对双亲/孩子结构连接算法的研究与改进

王治和　谢　斌
(西北师范大学数学与信息科学学院　兰州 730070)

摘　要　结合区间编码和结点模型映射方法提出一种用于关系数据库的扩展存储模式。通过按广度优先遍历 XML 树实现对双亲/孩子关系结构连接算法的改进。改进后的算法降低了内存空间的开销,缩小了列表的扫描范围,明显提高了查找匹配速度,达到了查询优化的目的。

关键词　XML,查询优化,扩展存储模式,双亲/孩子,结构连接算法

Research and Improve for the Parent/Child Structural Join Algorithm

WANG Zhi-He　XIE Bin
(College of Mathematics and Information Science, Northwest Normal University, Lanzhou 730070)

Abstract　By the use of the region coding and node model mapping method, an extended storage schema is presented for relational-database. We use breadth-first traverse XML tree to improve the structural join algorithms for processing parent/ child relationships. The proposed method has advantages of saving memory, shrinking the scanning area of list and remarkably improving the rate matching of lookup. This algorithm improves the efficiency of XML data query.

Keywords　XML, Query optimization, Extended storage schema, Parent/child, Structural join algorithm

1　改进的扩展存储模式

本文在 XRel[1] 和 XParent[2] 两种模式的基础上,设计了一种基于结点模型映射方法的 XML 数据的关系存储模式——改进的扩展存储模式(extended storage schema),该模式是通过区间编码方案来反应 XML 文档模型的,目的是实现对文档树中任意两个结点对之间的祖先/后裔关系、双亲/孩子关系以及文档位置关系的检测,并加速 Xpath 路径表达式的计算。其基本思想是:对 XML 文档树中的所有结点分别进行一次深度优先遍历和一次广度优先遍历,分别产生这些结点的深度扩展遍历序号和广度遍历序号,以深度扩展遍历序号作为结点的标识。XML 文档树中的结点包括元素结点、属性结点以及属性值。它由四个关系表组成(表 1)。

表 1　改进的 XML 文档关系存储模式

Element	(docID, order, maxorder, depth, pathID, parentorder, parentmax, gdorder)
Attribute	(docID, order, maxorder, depth, pathID, parentorder, parentmax, gdorder)
Value	(docID, order, pathID, value)
Path	(pathID, pathexp)

Element 表、Attribute 表分别用来存储元素结点和属性结点的内容;Value 表用来存储 XML 文档的内容,即所有文本结点的内容及属性结点的值;Path 表存储 XML 文档中所有从根开始到另一个元素或属性的路径。

其中,docID 是该结点所在文档的文档标识;order 是该结点的深度扩展遍历序号;maxorder 是以该结点为根的子树中所包含的所有元素结点、属性结点以及文本结点中最大的 order,以反映结点之间的祖先/后裔关系和之前/之后关系(注:只要是不小于最大 order 的任意一个整数,为便于给后裔对象的插入预留序号的空间,该结点之后的第一个结点的 order 取值只要大于该结点的 maxorder 值即可);depth 是该结点在文档树中所处的层数以反映祖先/后裔关系中的嵌套层数关系;pathID 是从根开始到该结点的路径表达式的标识,以支持对简单绝对路径表达式按路径索引的方法(即路径法)进行计算;parentorder 和 parentmax 分别是该结点的双亲元素结点的 order 和 maxorder;parentorder 用来支持结点之间双亲/孩子关系、左兄弟/右兄弟关系的计算,parentmax 用来加速结构连接的计算;gdorder 是广度遍历序号,用于支持兄弟结点关系的计算。pathexp 域存储标记路径。

Element 表、Attribute 表和 Value 表中(docID、order)是主键,pathID 是外键;Path 表中 pathID 是主键。

图 1 所示的是一个 XML 文档片断及相应的数据模型。

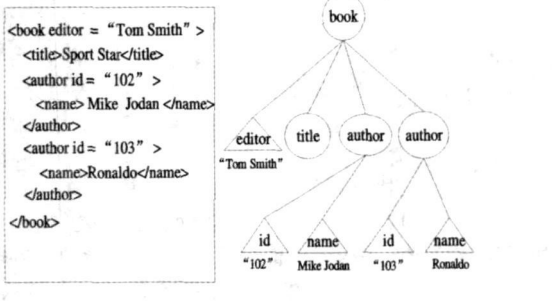

图 1　XML 文档片断及相应的数据模型

表 2 反映的是该 XML 文档的扩展存储模式的编码结果,包括元素和属性结点的(order, maxorder, depth, gdorder)字段。

2　对双亲/孩子结构连接算法的改进

2.1　双亲/孩子结构连接算法

首先引入一棵抽象的 XML 树,如图 2 所示。其中三角

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

形结点代表无关结点，方框结点代表双亲结点，圆形结点代表孩子结点，结点旁边标示的仅仅是它们改进的扩展存储模式编码中的(order，maxorder，depth)字段。因此，该树的双亲列表为 PList(p_1，p_2，p_3，p_4，p_5)，孩子列表为 CList(c_1，c_2，c_3，c_4，c_5，c_6，c_7，c_8，c_9，c_{10}，c_{11}，c_{12})。

表 2　XML 文档片断的扩展存储模式的编码结果

元素或属性名	(order，maxorder，depth，gdorder)
book	(1,150,0,1)
editor	(2,6,1,2)
title	(7,27,1,3)
author	(39,66,1,4)
id	(40,40,2,6)
name	(41,45,2,7)
author	(67,94,1,5)
id	(68,68,2,8)
name	(69,73,2,9)

将 PList 列表中的所有结点分解成若干个不相交的结点子集，每个结点子集包含一个根结点 r 以及所有属于这个根结点的后裔结点，它们是在 PList 列表中的一串连续结点。记 PList 列表中一个不相交子集的第一个结点为 r，结点 r 及 r 在 PList 列表中的所有后裔结点集为 Pr。

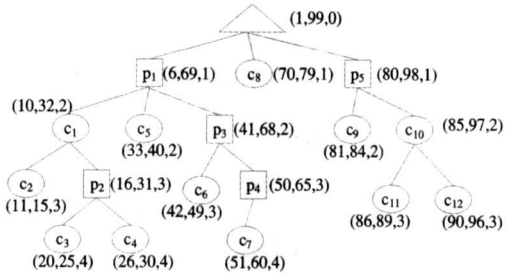

图 2　一棵抽象的 XML 树

该算法实现按双亲有序的结构连接，它的主要思想是：为了实现按双亲有序的输出，需要维护一个队列 CQueue，它用于存储结点 r 在 CList 列表中的所有后裔结点，它们也是 CList 列表中一串连续结点。显然，队列 CQueue 包含了 Pr 中每一个结点在列表 CList 中的所有孩子。

算法的主要处理过程如下：

(1)记 PList 列表中的当前结点为 r。

(2)从 CList 列表的当前结点开始，顺序扫描该列表，定位结点 r 的第一个可能的孩子结点，即定位 order $>r\cdot$order and depth $=r\cdot$depth$+1$ 的第一个结点。

(3)从第一个可能的孩子结点开始，将 CList 表中的属于结点 r 所有后裔结点$\{c_i\}$进入队列 CQueue。

(4)依次对 P_r 中的每一个结点 p_m，在队列 CQueue 中进行双亲/孩子关系匹配，并输出匹配结点对(r,c_i)。

(5) 转第一步到 PList 表中的下一个结点进行处理。

算法的执行过程如图 3。

对于图 2 所示的 XML 文档树，PList 列表中的所有结点被分成两个子集$\{p_1,p_2,p_3,p_4\}$和$\{p_5\}$。对于 PList 列表中的第一个子集$\{p_1,p_2,p_3,p_4\}$，它们在 CList 列表中的后裔子集$\{c_1,c_2,c_3,c_4,c_5,c_6,c_7\}$进入队列 CQueue，以便子集$\{p_1,p_2,p_3,p_4\}$中的结点进行双亲/孩子匹配连接；对于 PList 列表中的第二个子集$\{p_5\}$，由于它只有一个结点(即并没有嵌套)，因此结点 p_5 与 CList 列表中的后裔子集中的结点的双亲/孩子的连接结果直接输出，并不需要将 PList 列表中的后裔子集进队列。

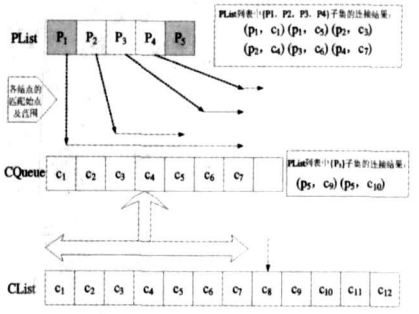

图 3　算法的执行过程

2.2　改进的双亲/孩子结构连接算法

上一节介绍的算法是按深度方向遍历 XML 树的顺序实现的，经过研究我们发现如果按广度方向遍历 XML 树，在寻找双亲/孩子结点的连接关系时方法更为简单，效率更高。

将图 2 按广度方向遍历后转换为图 4，记树的双亲列表为 PList '(p'_1，p'_2，p'_3，p'_4，p'_5)，孩子列表为 CList '(c'_1，c'_2，c'_3，c'_4，c'_5，c'_6，c'_7，c'_8，c'_9，c'_{10}，c'_{11}，c'_{12})。

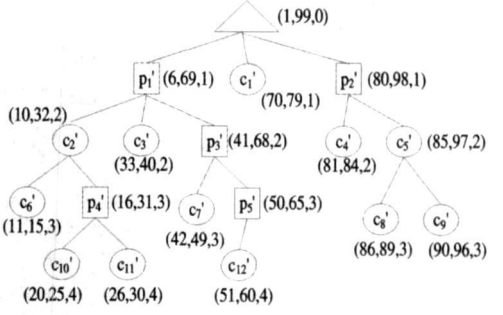

图 4　按广度遍历转换后的图

该算法的主要处理过程如下：

(1)记 PList '列表中的当前结点为 r。

(2)从 CList '列表的当前结点开始，顺序扫描该列表，定位结点 r 的第一个可能孩子结点，即定位 order $>r\cdot$order and depth $=r\cdot$depth$+1$ 的第一个结点。

(3)从第一个可能的孩子结点开始，将 CList '表中进行双亲/孩子关系匹配，并输出匹配结点对。

(4)转第一步到 PList '表中的下一个结点进行处理。

算法：

输入：两个参与连接的表：双亲列表 PList '和孩子列表 CList '。
输出：连接结果 output。

```
Broad-Tree-Children(PList ', CList ')
/* 假设 AList 列表和 DList 列表中的所有结点都有相同的 docID */
1  r=PList '->first Node; c = CList '->first Node; output=NULL;
2  while (the PList ' and CList ' are not empty ){
3  while(c·order <= r·order or c·depth<>r·depth+1) c=c->
          next Node; //在 CList '列表中，定位 r 结
                     点的第一个可能孩子结点;
4  for(;c·order<r·maxorder and c·depth==r·depth+1; c=c->
          next Node){
5  append(r,c) to output;
6  }// 输出 r 的连接结果
7   r=r->next Node
8   }
```

算法的执行过程如图 5。

改进后的双亲/孩子结构连接算法是按广度方向遍历 XML 树实现的，该算法前一次扫描的终点即为下一次扫描的

(下转第 139 页)

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

on Advanced Information Systems Engineering, Stockholm, Sweden, 2000. 247～263

11 McIlraith S, Son T. Adapting golog for composition of semantic web services. In: Proceedings of the 8th International Conference on Knowledge Representation and Reasoning, Toulouse, France, 2002. 482～493

12 Sohrabi S, Prokoshyna N, McIlraith S. Web Service Composition via Generic Procedures and Customizing User Preferences. In: 5th International Semantic Web Conference, Athens, USA, 2006. 597～611

13 Sirin E, Parsia B. Planning for Semantic Web Services. In: Semantic Web Services Workshop at the 3rd International Semantic Web Conference, Florida, USA, 2004

14 Sirin E, Parsia B, Wu D, et al. HTN planning for web service composition using SHOP2. Journal of Web Semantics, 2004, 1(4): 377～396

15 Sirin E, Parsia B, Hendler J. Template-based composition of semantic web services. In: AAAI Fall Symposium on Agents and the Semantic Web, Virginia, USA, 2005

16 Rao J, K¨ungas P, Matskin M. Application of Linear Logic to Web Service Composition. In: Proceedings of the 1st International Conference on Web Services, Las Vegas, USA, 2003. 3～9

17 Rao J, Küngas P, Matskin M. Logic-based Web services composition: from service description to process model. In: Proceedings of the 2004 International Conference on Web Services, San Diego, USA, 2004. 446～453

18 Medjahed B, Bouguettaya A, Elmagarmid A K. Composing Web services on the Semantic Web. The VLDB Journal, 2003, 12(4): 333～351

19 Berardi D, Calvanese D, De Giacomo G, et al. Automatic composition of e-services that export their behavior. In: Proceeding of 1st International Conference on Service Oriented Computing, Trento, Italy, 2003. 43～58

20 Berardi D, De Giacomo G, Lenzerini M, et al. Synthesis of underspecified composite e-services based on automated reasoning. In: Proceeding of 2nd International Conference on Service-Oriented Computing, New York, USA, 2004. 105～114

21 Berardi D, Calvanese D, De Giacomo G, et al. Composition of Services with Nondeterministic Observable Behavior. In: Proceeding of 3rd International Conference on Service Oriented Computing, Amsterdam, USA, 2005. 520～526

22 Bultan T, Fu X, Hull R, et al. Conversation Specification: A New Approach to Design and Analysis of E-Service Composition. In: Proceedings of the 12th International World Wide Web Conference, Budapest, Hungary, 2003. 403～410

23 Berardi D, Calvanese D, De Giacomo G, et al. Automatic Composition of Transition-based SemanticWeb Services with Messaging. In: Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, 2005. 613～624

24 Hamadi R, Benatallath B. A Petri Net-based Model for Web Service Composition. In: Proceedings of the 14th Australasian Database Conference, Adelaide, Australian, 2003. 191～200

25 Qian Z Z, Lu S L, Xie L. Automatic Composition of Petri Net Based Web Services. Chinese Journal f Computers, 2006, 29(7): 1057～1066

(上接第 127 页)

始点，只需分别扫描 PList′列表和 CList′列表一遍，无须维护 CQueue 队列，即可直接输出匹配结果。

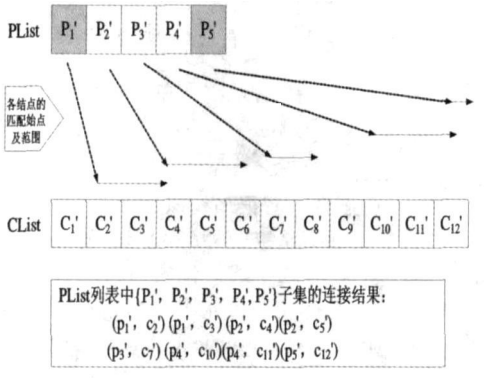

图 5 改进后算法的执行过程

2.3 算法分析及结论

假设改进前双亲列表 PList 的元组数为 m，孩子列表 CList 的元组数为 n，改进前算法的存储空间开销为($m+n+|$CQueue$|$)，$|$CQueue$|$ 表示队列的长度。在最坏情况下，$|$CQueue$|$ 为 n，算法的存储空间为($m+2n$)；在最好情况下(即不存在嵌套现象则无需队列)，$|$CQueue$|$ 为 0，算法的存储空间为($m+n$)。

又设改进后双亲列表 PList′的元组数为 m'($m'=m$)，孩子列表 CList′的元组数为 n'($n'=n$)，因为改进后的算法无论在任何情况下都无需维护队列，所以改进后算法的存储空间开销为($m'+n'$)。也就是说改进后的算法的空间开销总是和改进前算法的空间开销最好的情况下相等。

即得出结论：改进后的算法更节省内存空间的开销。在上例中改进后的算法就比改进前的算法节省了 7 个存储结点的空间，而且当 XML 树中结点数较多并存在嵌套时，队列 CQueue 的长度与 CList 列表的元组数有关，CList 列表的元组数越多且存在嵌套，队列 CQueue 的长度也就越长，这时算法的空间存储开销也就随 CQueue 的长度的增加而增加。改进后的算法不需维护 CQueue 队列，所以效果更加明显。

参 考 文 献

1 Yoshikawa M, Amagasa T, Shimura T, Uemura S. XRel: A path-based approach to storage and retrieval of XML documents using relational databases. ACM Trans. on Internet Technology, 2001, 1(1): 110～141

2 Jiang H, Lu H, Wang W, et al. XParent: An Efficient RDBMS-Based XML Database System. In: Proceedings of the 18th International Conference on Data Engineering, 2002

3 World Wide Web Consortium. XML Path Language(XPath) 1.0. W3C Recommendation. November 1999. http://www.w3.org/TR/xpath

4 Jiang H F, Lu H J, Wang W, Ooi B C. XR-Tree: Indexing XML data for efficient structural joins. In: Dayal U, Ramamritham K, Vijayaraman T M, eds. Proc. of the 19th Int'l Conf. on Data Engineering. Los Alamitos: IEEE Press, 2003. 253～264

5 Fan C, Funderburk J, Lam Hou-in, et al. XTABLES: bridging relational technology and XML. IBM Systems Journal, 2002, 41(4)

6 Al-Khalifa S, Jagadish H V, Koudas N, et al. Structural joins: A primitive for efficient XML query pattern matching. In: Agrawal R, Dittrich K, Ngu A H H, eds. Proc. of the 18th Int'l Conf. on Data Engineering. Los Alamitos: IEEE Press, 2002. 141～152

(C)1994-2022 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net