一种基于扩展区间编码的 XML 结构连接算法①

A Structural Joins Algorithm Based on Extended Region Coding

覃遵跃 黄 云 (吉首大学 信息管理与工程学院 湖南 张家界 427000)

摘 要: 高效的结构连接计算是 XML 进行有效查询的关键。分析了多谓词归并结构连接算法低效的原因，在 Zhang 编码方案的基础上，提出了一种扩展的区间编码方案 BEN，该编码方案可以大规模减少搜索结点的数目。实验结果表明，该编码方案有效提高了支持包含关系结构连接算法的执行性能。

关键词: XML 技术 结构连接 扩展区间编码

XML 文档结构查询通常被转化为两个结点列表之间的包含关系或者文档位置关系的结构连接操作，同时关键字操作也被转化为两个结点列表之间的包含关系的结构连接操作。因此，有效支持结构连接对 XML 查询的有效实现是解决问题的关键。

目前，在结构连接方面已经进行了大量的工作，提出了一系列有效的结构连接算法，这些结构连接算法大都是基于归并的思想，充分利用 XML 数据的结构特点来减少连接的扫描代价，文献[1]提出了一种基于划分的结构连接算法，文献[2]提出了基于二分法的结构连接方法；有些算法在归并的基础上，根据 XML 数据的结构特点利用索引来进行一步减少连接的扫描代价，文献[3]提出了在区间编码基础上建立索引来提高结构连接效率。

在结构连接操作中，计算包含关系结构是结构连接计算的一个重要方面，文献[4,5]提出了计算包含关系结构连接的多谓词归并结构连接算法 MPMGJN(Multi-predicate merge join)。但是，研究发现，在已有的算法中，还有改进的空间。通过扩展区间编码后，在搜索过程中采用跳跃结点的方式可以有效提高计算包含结构关系的效率。

1 扩展区间编码

1.1 相关概念

由 C.Zhang 和 J.Naughton 提出的一种区间编码方案成为 Zhang 编码[4]，它的编码规则是：XML 文档树中的每一个结点被赋予一个二元组<begin,end>。对树 T 的所有结点进行先序遍历，每一个结点在遍历时分别被访问两次并产生两个序号。一次是在遍历该结点的所有后裔结点之前访问该结点，并产生该结点的序号 begin；另一次是在遍历完该结点的所有后裔结点后再一次访问该结点，并产生该结点的另一个序号 end。因此，树 T 中的任意两个结点 u 和 v 是祖先/后裔关系，当且仅当 begin(u)<begin(v) ∧ end(v)<end(u)，即祖先结点 u 的编码区间[begin(u),end(u)]包含后裔结点 v 的编码区间[begin(v),end(v)]。对于该编码方案，begin 作为结点的唯一标识。

基于 Zhang 编码的 MPMGJN 算法[4]的基本思想是：设参加连接的两个关系表 AList(祖先)和 DList(后裔)，则对外表 AList 中的一个元组 a1，首先在内表 DList 中顺序搜索到可能与元组 a1 进行连接的第一个元组(即 begin>a1.begin 的第一个元组)，称为扫描点，然后在内表 DList 中从扫描点开始顺序扫描，对满足 begin<a1.end 条件的所有元组 dj，再判断是否满足连接条件，若满足则产生连接结果元组 a1・dj；继续对外表 AList 中的第二个元组 a2 重复上面的步骤，直到外表 AList 或内表 DList 中的元组连接完毕。

在该算法中，从扫描开始点进行扫描时，扫描的长度与某结点子树结点的个数有关，如果知道了结点子树结点的个数，则可以减少扫描的长度，针对该问题，提出了如下的扩展编码方案。

① 基金项目:湖南省教育厅科学研究项目(06C658)
收稿时间:2008-09-25

1.2 扩展区间编码方案

在 Zhang 区间编码方案的基础上进行扩展，每个节点赋予一个三元组<begin,end,number>，begin 与 end 元素的含义与 Zhang 编码一样，number 元素标识结点子树中子结点的数目，如图 1 所示，称为 BEN 编码。基于 BEN 编码，判断元素之间包含关系的性质与 Zhang 编码一样。基于该扩展编码，每个结点被译码为五元组：<docID,begin, end,level, number>。

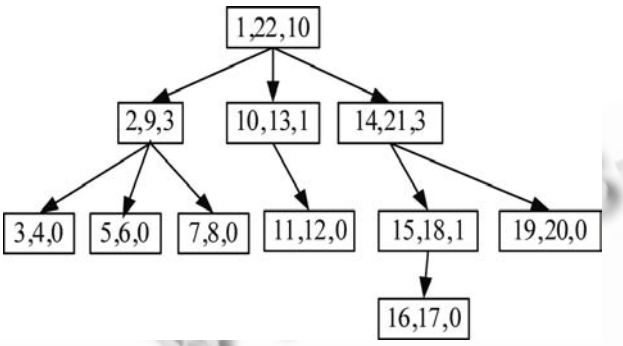


图 1　扩展区间编码 BEN

对于扩展区间编码 BEN，具有如下两个重要的结论：

命题 1　假设列表 List 按<docID,begin>升序有序，任意给定一个结点 r∈List，则结点 r 在列表 List 中的所有后裔结点个数为 r.number，并且这些结点是连续的。

命题 2　假设列表 List 按<docID,begin>升序有序，任意给定一个结点 r∈List，则结点 r 在列表 List 中的第一个可能后裔结点是满足 begin>r.begin 的第一个结点(同时还必须满足 begin<r.end，否则说明结点 r 在列表 List 中不存在后裔结点)，即在列表 List 中满足 begin>r.begin 且 begin 取最小值的结点；并且结点 r 在列表 List 中的所有后裔结点个数为 number。

2　改进MPMGJN算法Improve-MPMGJN

Improve-MPMGJN 算法的思想：设参加连接的两个关系表 AList 和 DList，则对外表 AList 中的一个元组 a1，首先对内表 DList 中的一个元组 r 进行连接判断，如果 r.begin<a1.begin，则跳过 r 的子结点 number 个；否则在内表 DList 中以该元组为扫描点开始顺序扫描，对满足 r.begin<a1.end 条件的所有元组 dj，再判断是否满足连接条件，若满足则产生连接结果元组 a1・dj；继续对外表 AList 中的第二个元组 a2 重复上面的步骤，直到外表 AList 或内表 DList 中的元组连接完毕。

算法 1　基于扩展区间编码 BEN 的改进多谓词归并连接算法 Improve-MPMGJN

输入：两个参与连接的表 AList 和 DList。

输出：连接结果。

```
ImproveContainmentMerge(AList,DList){
    set cursor1 at beginning of AList;
    set cursor2 at beginning of DList;
    while(cursor1≠end of AList and cursor2≠
end of DList){
        if(cursor1.docID<cursor2.docID)
            cursor1++;
        else if(cursor2.docID<cursor1.docID)
            cursor2++;
        else{
mark=cursor2;//记录下一次搜索起始点
while(cursor2.begin<cursor1.begin and cursor2
≠end of DList)
//对内表进行搜索，如果该结点不可能满足包含条件，
则跳//过 number 个结点
    cursor2=cursor2+number;
        if(cursor2=end of DList){
            cursor1++;
            cursor2=mark;
        }
        else    if(cursor1(directly)    contains
cursor2){
            mark=cursor2;//记录下一次搜索起
始点
            do{//对内表进行扫描连接
                merge cursor1 and cursor2;
                cursor2++;
            }while(cursor1(directly)    contains
cursor2 and cursor2≠end of DList);
        }
        cursor1++;
        cursor2=mark;    //定位下一次搜索起始
点
```

```
    }
}
}
```

3　实验测试结果

3.1 实验环境

这里对 Improve-MPMGJN 算法进行了性能测试，并且将它与 MPMGJN 算法进行了性能比较，所有测试程序均用 Visual C++6.0 编写。实验平台是 P4 3.2G、内存 1GMB、硬盘 120GB，操作系统为 Windows 2000 Server。图 3 所示的是测试 XML 数据集的 DTD。利用设计的 XML 文档生成器自动生成一个符合图 2 所示的 DTD 的 XML 文档，文档的大小为 50.6MB。在该文档中，共有 1 328 654 个元素和属性结点，其中，1600 个 book 元素，265 712 个 title 元素结点，3891 个 chapter 元素结点，261 872 个 section 元素结点，98 168 个 description 元素结点，70 372 个 keyword 元素结点。

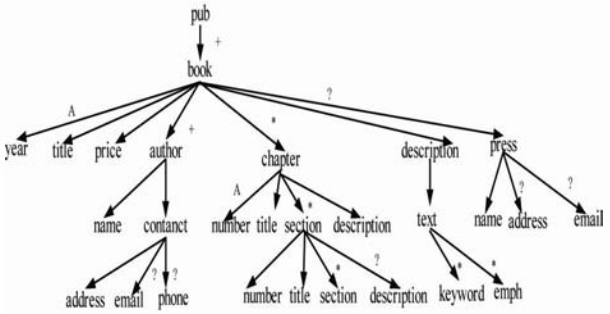


图 2　测试数据集的 DTD

实验选取了 5 个测试查询实例(为了讨论的简洁，所有结点均在一个 XML 文档中)。分别是 book/title、book/chapter/section、book/chapter/ section/ title、book/chapter//description 和 book/ child::*。通过改变选择度来进行测试。为了实现这个目的，在原有元素列表的基础上，通过改变双亲结点的百分比（即从双亲列表中随机删除一定比例的元素结点）来对 Improve-MPMGJN、MPMGJN 等结构连接算法的性能进行测试比较，双亲结点的百分比按从大到小的顺序分别选取 100%,75%,50%,30%四个值。

3.2 实验结果分析

选取的性能指标有两个：①扫描元素的数目：即结构连接时读取元素结点(即记录)的数目，这个指标能够反映出算法跳过不相关元素结点的能力。②耗用的时间：连接耗用的时间，用于反映算法的综合性能。

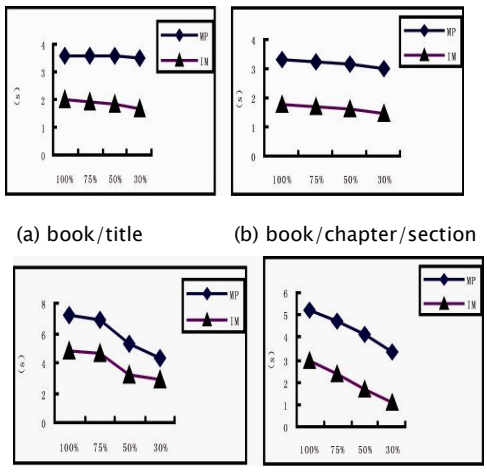
当孩子结点的数量保持不变，双亲结点百分比变化时，两种算法扫描元素结点的数量如表 1 所示。

表 1　孩子结点不变并改变双亲结点百分比时，两种结构连接算法实际扫描元素结点数量的对比

单位：千，其中 MP 表示 MPMGJN; IM 表示 Improve-MPMGJN

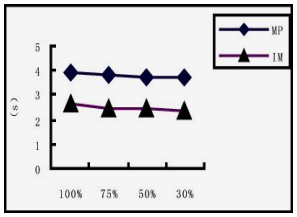
	book/title		book/chapter/section		book/chapter/section/title		book/chapter//description		book//child::*	
Join-P	MP	IM	MP	IM	MP	IM	MP	IM	MP	IM
100%	265	166	261	143	492	331	321	152	721	476
75%	261	154	252	128	451	294	279	108	720	451
50%	255	132	246	122	365	226	214	89	720	409
30%	248	129	231	113	284	178	178	67	716	381

分析表 1 可知，从横向比较，在每种百分比相同情况，两种算法扫描的结点数目是不一样的，Improve- MPMGJN 算法较优。从纵向比较，随着百分比的降低，两种算法扫描的结点数都在减少，但减少的幅度不同，book/chapter//description 查询减少的幅度最大，算法 Improve- MPMGJN 优于 MPMGJN 算法。这主要是因为利用 number，可以跳过事先判断不参与连接的孩子元素结点，这对于在孩子列表中存在大量的双亲元素结点后裔，但不是它们的孩子情况下特别有效。算法运行时间如图 3 所示。其中，横坐标表示双亲(或祖先)元素结点的百分比，纵坐标表示结构连接算法的执行时间，单位秒(second)。



(a) book/title　(b) book/chapter/section

(c)book/chapter/section/title (d)book/chapter//description



(e) book//child::*

图 3　孩子结点不变并改变双亲结点百分比时，两种结构连接算法实际运行耗用时间对比

从运行时间来看，在 book/chapter//descrip-tion 情况下(图 3(d))，Improve-MPMGJN 算法具有最好性能，从图 2 可知，因为 chapter 下 description 中存在嵌套结构，建立了孩子结点索引后可以减少扫描孩子结点的数目。其它四种情况 Improve-MPMGJN 算法也优于 MPMGJN，因为可以通过双亲的 number 来减少扫描孩子结点的数目，所以具有良好的性能优势。

4　结束语

本文在分析比较现有多谓词归并结构连接算法的基础上，针对算法中存在扫描多余的不满足包含关系结构连接的结点问题，提出了一种新的扩展区间编码 BEN，并在此基础上实现了结构连接算法。该算法通过跳跃子结点的机制解决了不必要扫描的多余子结点，从而提高了结构连接的效率。下一步的研究工作是在 BEN 编码基础上，讨论利用索引结构进一步优化查询操作。

参考文献

1 任家东,尹晓鹏,郭晓丹.一种新的基于划分的结构连接算法.计算机工程, 2007,33(6):95－97.

2 张晶,丁怡心,刘山.基于二分法的 XML 结构连接.计算机工程, 2007,33(18):62－63.

3 万常选,刘云生,徐升华,刘喜平,林大海.基于区间编码的 XML 索引结构有效实现结构连接.计算机学报, 2005,28(1):113－127.

4 Zhang C, Naughton J, DeWitt D, et al. On Supporting Containment Queries in Relational Database Management Systems. Mehrotra S eds. Proceedings of the 20th ACM SIGMOD International Conference on Management of Data. New York, USA: ACM Press 2001:426－437.

5 Al-Khalifa S, Jagadish H V, Koudas N, et al. Structural Joins: A primitive for efficient XML Query Pattern Matching. Proceedings of the 18th IEEE ICDE International Conference on Data Engineering. California, USA: IEEE Computer Society, 2002:141－152.

6 Li DY, Li CP.TRACK: A Novel XML Join Algorithm for Efficient Processings Twig Queries. Proceedings 19th Australasian Database Conference (ADC 2008). Wollongong, Australia. Australian Computer Society Inc, 2008:137－144.