第33卷 第18期 计 算 机 工 程 2007年9月
Vol.33 No.18 Computer Engineering September 2007
·软件技术与数据库· 文章编号：1000—3428(2007)18—0062—02 文献标识码：A 中图分类号：TP311.12

基于二分法的XML结构连接

张 晶，丁怡心，刘 山
(中国民航大学计算机学院，天津 300300)

摘 要：在XML数据的查询处理过程中，基于区域划分的连接算法在处理XML数据无序和不存在索引时，是一个效率较高的算法。该文利用区域编码的特点对输入集合进行穷尽的递归划分，在划分的代价下，逐步定位祖先-后代的结构关系。使用二分法进行划分后，再完成结构连接，提高了结构连接的效率，实验表明该算法在XML数据的查询处理上是一个有效的方法。

关键词：XML结构连接；区域划分；区域编码

XML Structural Join Based on Dichotomy

ZHANG Jing, DING Yi-xin, LIU Shan
(College of Computer Science & Technology, Civil Aviation University of China, Tianjin 300300)

【Abstract】In the process of querying and managing XML data, structural join algorithm based on range partition is an efficient algorithm when the XML data are not sorted or indexed. In this paper, this algorithm takes advantage of the region numbering scheme to recursive partition the input sets exhaustively and with the cost of partition, the structural relation between ancestors and offspring can be localized. The partition of dichotomy before structural join enhances the efficiency of join. The experiment indicates that this algorithm is an efficient one when querying and managing XML data.

【Key words】XML structural join; region partition; region coding

1 概述

随着互联网络Internet的蓬勃发展，XML作为一种数据表现和交换的格式在WWW上得到了广泛的应用。正因如此，XML数据的管理和查询问题越来越引起人们的重视。XML文档的结构是一种层次结构，可以用一个树状模型表示，树中的节点一般由一个二元序偶的编码表示，每个二元序偶对应XML的一个元素，树中的边描述了元素之间的父子关系。针对这种树结构数据XML查询语言，如Xpath、XQuery等，都将路径表达式作为重要的组成部分。为了高效地计算路径表达式，学者们提出一种查询处理策略，将一个复杂的查询模式分解成为若干个二元基本结构关系的集合，通过计算二元基本结构关系及合并，完成XML数据的结构连接。因此，计算二元基本结构关系成为查询处理的关键操作，这种操作被称为结构连接。

在基于基本结构关系计算的XML查询处理中所关注的两个问题：一是快速判断树中任意节点之间的结构关系，包括父子关系和祖先-后代关系；二是从给定的两个元素节点集合中高效地得到满足某种结构关系的所有数据。节点间结构关系的判断依赖于对节点的编码，而计算两个集合中所有满足结构关系的节点对，则是XML结构连接所关注的问题。

由于XML结构连接技术在Internet发展过程中的重要性，该领域的研究备受关注，目前已有的算法大致可以分为两类：排序合并[1,2]和划分方法[3]。前者需要索引，而后者不要求数据集合有序或存在索引，能够取得较优的I/O效率，在数据无序和不存在索引的情况下优于排序合并类的算法。文献[4]提出了一种划分方法，对文献[3]的算法作了微小的改动。上述算法都存在一个利用计算机查找面临的基本问题，即对元素编码的比对。特别是对于大的XML数据，算法效率必然受到极大的影响。本文基于文献[4]分治的思想，给出一种通过等分区间尽可能减少元素编码的比对，逐步定位祖先-后代结构关系。实验表明该算法较以往算法在效率上有明显的提高。

2 背景知识

2.1 区域编码

根据XML文档中元素的编码判断节点之间的结构关系。节点编码有多种方法，K-ary tree编码方法，PbiTree编码等。而区域编码是目前应用最广泛的一种，它赋给每个元素节点一个编码(start,end)。start表示元素在文档中的起始位置(以单词的个数为单位)，end表示元素在文档中的结束位置。根据XML文档的规定，编码所覆盖的区间不会部分相交，即节点之间或者完全包含，或者不相交。

2.2 区间的相关定义

结构连接是路径查询处理中的核心操作，给定一个可能的祖先元素节点的集合A和一个可能的后代节点的集合D，结构连接操作的任务是找到所有的节点对(a_i, d_i)，a_i属于A，d_i属于D，a_i是d_i的祖先节点。区域划分方法:将连接任务$A \triangleright D$分界为若干个较小的子任务$A1 \triangleright D1$，…，$Am \triangleright Dm$保证$A \triangleright D=\bigcup_{1}^{m} Ai \triangleright Di$。

定义1 假设XML元素节点的编码范围为R(start,end)，将编码范围R中的任意一个范围(si,ei)称为一个区间R_i。

定义2 给定一个区域编码为(s,e)的节点E和一个区间

基金项目：国家自然科学基金资助项目(60572168)
作者简介：张 晶(1982－)，男，硕士研究生，主研方向：数据挖掘与信息处理；丁怡心，硕士研究生；刘 山，教授
收稿日期：2006-09-24 **E-mail：**breeze5025@msn.com

R_i (si,ei)，如果它们满足如下条件之一，则称 E 与区间 R_i 相交：

(1)$s<si<ei<e$；

(2)$si \le s<e \le ei$；

(3)$s<si<e<ei$；

(4)$si<s<ei<e$。

如果满足 $si \le s \le ei$，则区间 R_i 是 E 的开始区间。所有与 E 相交的区间组成的集合称为 E 的相关集，所有 E 的开始区间组成的集合称为 E 的开始集。

定理[4] A中可能与Di中元素具有祖先-后代关系的元素一定存在于Ai中。

分析上述定理在利用相交区间定义了区间的 A 中元素，利用开始区间定义了区间的 D 中的元素。这时，并不能精确地定位祖先-后代关系。采用这种划分方法避免了 D 集合的数据复制。因此，需要通过递归划分区间，直到 Ai 或 Di 剩下唯一的元素为止，这样最大限度地缩小了祖先-后代关系节点的查找范围。

3 算法的问题和改进

基于区域划分的 XML 结构连接算法：该算法基于任务分解的思想，利用区域编码的特点对输入集合进行划分。划分原则：尽量使划分得到的子集合完全装入内存时，假设可用内存的大小为 M，所选择的划分个数应当大于等于 $[\min(||A||,||D||)/m]$。这样产生的较小集合的子集合会有较大的可能小于可用内存的大小。首先根据集合 A 和 D 的大小以及可用内存的大小来确定子集合的个数 m，再依据划分原则将 A 和 D 分别划分为 m 个子集合 Ai 和 Di，$1 \le i \le m$。对所得到的每一对子集合 Ai 和 Di，如果有任意一个可以完全装入内存，则可以对其调用内存的结构连接方法；如果两个都大于可用内存，则需要对它们进行进一步的划分，直到有一个可以装入内存中为止。

3.1 改进的划分方法:

经过划分处理，原来的结构连接问题转化为多个子集合对之间的结构连接。这时候影响算法效率的主要因素是所选择的内存结构连接算法的 CPU 代价。在各种内存连接算法中，最基本的是循环嵌套算法，循环嵌套的代价相当高，为 $O(|Ai|.|Di|)$，因此并不可取。

基于二分法的 XMl 结构连接技术：前提是对原来的区域划分作一下改进，要求划分满足每对 Ai 和 Di 能同时装入内存。对调入内存的 Ai 和 Di 采用二分法，逐步递归，将区域细分，划分到 Ai 或者 Di 至多只含有一个元素为止。然后根据祖先后代节点的判断规则，Ai 中的元素 a 和 Di 中的元素 d，查找满足 a.start<d.start<a.end 的 a 和 d，产生连接结果，这样，效率得到了提高。

例 图 1 描述了两个输入集合 A 和 D 的元素编码情况，元素编码范围如图中虚线所示。当 k=2 时：

$A1=\{a_1, a_2, a_3\}$;$A2=\{a_1, a_3, a_4\}$;$D1=\{d_1, d_2, d_3\}$;$D2=\{d_4\}$

当 k=4 时：

$A1=\{a_1, a_2,\}$; $A2=\{a_1, a_3\}$; $A3=\{a_1, a_3\}$;$A4=\{a_1, a_4\}$

$D1=\{d_1\}$;$D2=\{d_2, d_3\}$；$D3=\{\}$；$D4=\{d_4\}$

因为 $D3=\varnothing$

所以递归结束

通过比对，得到最后结果$A \blacktriangleright D=\{(a_1,d_1), (a_2,d_1), (a_1,d_2), (a_1,d_3), (a_3,d_2), (a_3,d_3)\}$。

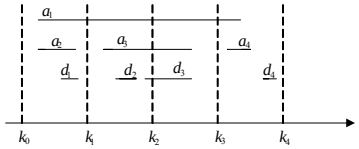

图 1 结构连接的一个实例

3.2 内存中的结构连接算法

基于上述二分法，设计了具体的递归求解算法 TwoDivideJoin 算法，具体步骤如下所示。

Algorithm TwoDivideJoin(*A*,*D*, Start, End)

Input: *A*: ancestor set, *D*: descendant set, Start: region start, End: region end

Description:

```
1     initialize: A1,A2,D1,D2 as subset
2       Mid=(End-Start)/2
3       for each element i in A and element j in D do
4             if  i.start<Mid  then
5         add  i  to A1
6             if  i.end>Mid  then
7         add  i  to A2
8             if  j.start >=Start&&j.start<=Mid
9         add  j  to D1
10          if j.start >Mid
11        add  j  to D2
12      end for
13      if  the number of element in A1 or D1 >=1
14      TwoDivideJoin(A1,D1, Start, Mid)
15    if  the number of element in A2 or D2 >=1
TwoDivideJoin(A2,D2, Mid, End)
```

首先解析试验 XML 文档，将每个元素进行编码(start,end)，随机生成集合 A、D。再进行二分，将 A、D 划分为 $A1$、$A2$ 和 $D1$、$D2$，然后将 $A1$、$A2$ 和 $D1$、$D2$ 继续二分，直到 Ai 和 Di 至多有一个元素为止。

4 实验结果和分析

为了评价算法的性能，笔者做了大量的实验，并对其进行分析，选择了循环嵌套算法和本文提出的 TwoDivideJoin 进行比较，在人工数据集和真实数据集上进行了实验。实验的输入集合 A 和 D 都是无序的，也没有索引存在。

4.1 实验设置

所有的算法都用 C# 实现，实验都在 Pentium 4 1.7GHz，512MB RAM，60GB 硬盘的笔记本电脑上运行，底层操作系统是 Windows XP，实验文档使用 XMark 生成。

表 1 两种算法的实验对比

	元素点数/个	*A* 元素数/个	*D* 元素数/个	区域划分算法耗时/s	二分法耗时/s
k=1/3	6 000	1 500	4 500	0.203	0.047
	12 000	3 000	9 000	0.813	0.141
	24 000	6 000	18 000	3.593	0.563
	48 000	12 000	36 000	25.625	2.25
	100 000	25 000	75 000	51.141	8.953
k=1/1	6 000	1 500	4 500	0.375	0.046 9
	12 000	3 000	9 000	1.063	0.141
	24 000	6 000	18 000	4.118	0.625
	48 000	12 000	36 000	33.656	2.344
	100 000	50 000	50 000	67.859	9.234

基于区域划分的 XML 结构连接算法与基于二分法的结构连接算法实验对比如表 1 所示，其中 k=*A* 元素数/*D* 元素数。

(下转第 66 页)