一种非归并不确定XML小枝模式查询算法

刘立新 张晓琳 吕 庆 张换香 褚艳华
(内蒙古科技大学信息工程学院 包头 014010)

摘 要 针对目前不确定XML小枝模式查询需要存储大量中间结果和归并中间结果的情况,提出一种非归并不确定XML小枝模式查询算法ProTwigList。该算法查询之前通过Tag+Level流进行剪枝,以减少待处理节点的数目;并扩展了区间编码来对剪枝后剩余的普通节点进行编码,用一定规则对分布节点进行标识;查询时采用公共分布节点路径的方法处理分布结点,最后结合最低公共祖先节点的概率计算查询结果的概率值。理论分析和实验结果证明了ProTwigList算法的查询效率。

关键词 不确定XML,P-文档,分布节点,区间编码,小枝模式
中图法分类号 TP311 **文献标识码** A

Matching Twig Patterns in Uncertain XML without Merging

LIU Li-xin ZHANG Xiao-lin LV Qing ZHANG Huan-xiang CHU Yan-hua
(Information Engineering School, Inner Mongolia University of Science and Technology, Baotou 014010, China)

Abstract In order to avoid storing large amounts of intermediate results and merging those intermediate results when matching twig patterns in uncertain XML, this paper proposed the algorithm ProTwigList. First, it uses Tag+Level stream pruning to reduce the number of nodes to be processed. Second, it extends the range encoding to encode the remaining ordinary nodes, and adopts rules to mark distributed nodes. Third, it uses the path of common distributed node to deal with distributed nodes. Finally, it utilizes the probability of the lowest common ancestor node to calculate the probability of every result. The theoretical analysis and experimental results prove the efficiency of ProTwigList.

Keywords Uncertain XML, P-document, Distributed node, Rang encoding, Twig pattern

1 概述

随着数据采集和处理技术的进步,人们对数据不确定性的认识也逐步深入。在经济、军事、物流、金融、电信等领域中,不确定性数据普遍存在,且扮演着关键角色[1]。不确定性数据分为离散型和连续型,离散型不确定数据是一个离散值,连续型不确定数据通常用概率密度函数表示[2,3]。概率XML数据是近年来提出的一种新的不确定数据的表示方法,是不确定XML数据的主要形式,目前应用概率XML管理不确定数据已经取得一系列的研究成果[4]。

小枝模式查询(又称Twig Pattern查询)是概率XML数据查询处理的关键步骤之一。执行小枝模式查询之后可以进一步对节点的值进行处理,离散型数据可以直接取出其值,连续型数据则进一步可以用积分法或者蒙特卡洛等方法处理[2,3]。文献[5]从查询语义的角度研究了概率XML数据小枝模式查询方式,根据查询执行方式的不同或者返回结果的情况将查询语义分为3种:布尔查询语义、完全语义查询和不完全语义查询。文献[6]扩展了区间编码后使用整体小枝模式的方法进行查询,但算法需要存储中间结果和归并中间结果两步,且在归并中间结果时存在不能正确归并的情况。文献[7]算法扩展了Dewey编码查找大于一定概率阈值的小枝模式,采用Tag+Probability的方式剪掉小于概率阈值的数据,并在归并时进一步利用过滤策略删除小于概率阈值的结果,但此算法仍然是一种归并算法。文献[8]扩展了Dewey编码查找概率值最大的Top-K小枝模式,并按概率大小对元素流进行排序,来提高查询效率。

本文研究非归并的不确定XML小枝模式查询算法:(1)使用Tag+Level流的方法剪去无用的数据流,以减少待处理节点的数目,从而节省处理时间。(2)扩展了区域编码使之适用于连续不确定XML数据,并通过对分布节点的标识正确处理分布节点。(3)提出一种非归并的不确定XML小枝模式查询算法:ProTwigList算法,并在经典数据集上进行测试,对ProTwigList算法的查询效率进行验证。

2 背景知识

2.1 数据模型

P-文档模型由普通节点和分布式节点两种类型的节点构成。每一个普通节点都有一个标签和一个唯一的标识符,普

到稿日期:2012-07-05 返修日期:2012-10-19 本文受国家自然科学基金(61163015),内蒙古科技大学创新基金(2011NCL024,2010NC041)资助。

刘立新(1983-),女,硕士,助教,主要研究方向为不确定数据管理,E-mail:99liulixin@163.com;张晓琳(1966-),女,博士,教授,主要研究方向为数据库理论与技术;吕 庆(1986-),男,硕士生,主要研究方向为数据库理论与技术。

通节点可能出现在样本空间的文件中，分布式节点仅用于定义产生随机文件的概率过程。分布式节点有：独立类型节点 ind[9]，独立节点在 P-文档树中出现的概率是独立的，不受其他节点的影响；互斥类型节点 mux[9]，互斥节点在 P-文档树中只能出现一个其他节点不出现的节点，或者全都不出现；支持连续不确定数据的连续分布节点 cont，连续分布节点仅出现在叶子上，任何可表达的连续分布都可以出现在 cont 节点中，用 cont 节点存储概率密度函数以及它们的参数。

P-文档模型将概率值附加到文档树的边上，各节点的概率依赖于其祖先的概率。一条边 e 的权重，记为 $\Pr(e)$，它表示分布节点选择孩子节点的概率。P-文档中的一个节点 V_i，在文档中出现的概率等于从根节点到节点 V_i 经过的所有的边的概率的乘积。

图 1 为一个包含 cont，mux，ind 3 种类型的分布节点的 P-文档，可表示为 $\text{PrXML}^{\text{cont,mux,ind}}$，其中叶子节点 B_1，B_2，B_3，B_4，B_5 是离散型的，叶子节点 C_1，C_2 是连续型的。叶子节点 B_1 出现的概率 $\Pr(B_1)=0.2*0.9=0.18$。

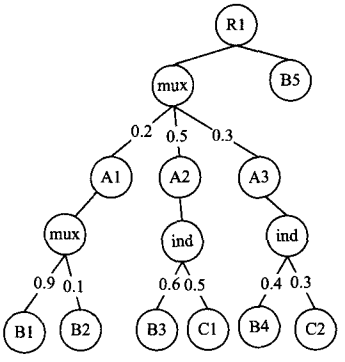

图 1 P-文档

2.2 不确定 XML 小枝模式查询

通常，XPath 查询语句包含孩子轴(/)、后裔轴(//)和谓词节点，表示成 XP{/,//,[]}，其经常用树的形式来表达，称为小枝模式，Xpath 查询也称小枝模式查询。与 XML 小枝模式查询不同的是：不确定 XML 小枝模式查询不仅需要在包含分布节点的概率 XML 数据中找出满足小枝模式出现的所有匹配结果，还需要计算出每个匹配结果出现的概率值。

为描述方便，本文中以带下标的大写字母表示 P-文档中的元素节点，以大写字母表示小枝模式中的查询节点。

定义 1 给定一个小枝模式查询 $Q=(V,E)$ 和一个 P-文档 $D=(V_D,E_D)$，其中 V 和 E 分别表示小枝模式中的节点集合和边集合，V_D 和 E_D 分别表示 P-文档中的节点集合和边集合。小枝模式匹配就是在 D 上匹配 Q 的过程，是 Q 中的查询节点到 D 中的元素节点的映射过程，满足以下两个条件：

(1)对于每一个查询节点 $X\in V$，V_D 中的元素节点 X_i 与 X 具有相同的标签名，且满足 X 的谓词约束。

(2)V_D 中的元素节点 X_i 和 Y_j 之间的关系必须满足 V 中相应查询节点 X 和 Y 之间的结构约束关系，如 P-C 关系、A-D 关系。

图 2 是 XPath 为 $R[//C]/A/B$ 所对应的小枝模式 Q'，Q' 在图 1 的 P-文档上的一个查询结果是 (R_1,A_2,B_3,C_1)，此结果出现的概率为：$0.5*0.6*0.5=0.15$。

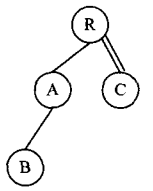

图 2 小枝模式 Q'

2.3 Tag+Level 流

Tag+Level 流最早在文献[10]中被提出。查询前，本文采用此方法去处理 P-文档中的普通节点，减少了待处理节点的数目，从而节省了 I/O 处理时间。

定义 2 P-文档中，一个 Tag+Level 流 X_v^L 是指在 P-文档树中具有相同标签名和树层次的元素节点的集合，L 表示层次值。X_v^L 中的元素节点是按照扩展的区间编码中的 Start 值升序排列的。

由定义 2 可知一个查询节点 U 的 Tag+Level 数据流可能有多个。例如 Q' 在图 1 的 P-文档上匹配时，查询节点 B 的 Tag+Level 流有：$X_B^2=\{B_5\}$，$X_B^3=\{B_1,B_2,B_3,B_4\}$。

引理 1[10] 对于查询节点 U 的一个 Tag+ Level 流 X_U^L，若 $\forall U_i\in X_U^L$，

(1)查询节点 U 的孩子节点集合是 $children(U)$，元素节点 U_i 在 $children(U)$ 的所有 Tag+ Level 流中找不到孩子节点(P-C 关系)或后代节点(A-D 关系)；

(2)查询节点 U 的父亲/祖先节点集合是 $parent(U)$，元素节点 U_i 在 $parent(U)$ 的所有 Tag+ Level 流中找不到父亲节点(P-C 关系)或祖先节点(A-D 关系)；

则 X_U^L 被剪掉。

3 非归并不确定 XML 小枝模式查询算法

3.1 扩展的区域编码

编码技术使得通过任意两个节点的编码都能够判断两个节点的关系，提高了查询效率。本文在处理 P-文档时，扩展了区域编码对普通节点进行编码；并用 0 起始的偶数标识独立节点，用 1 起始的奇数标识互斥结点。

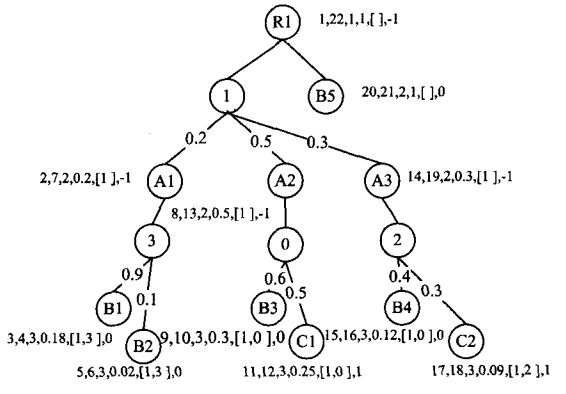

图 3 编码结果

扩展的区域编码由 6 元组组成(Start，End，Level，Pr，[Path]，Flag)。Start，End 是前序遍历 P-文档得到的两个整数值。Level 表示这个元素在 P-文档树中的深度。Pr 表示由根节点到此节点经过的路径上概率值的乘积，称为路径概率。Path 记录由根节点到此节点经过的分布节点。Flag 主要用于区分离散型节点和连续型节点，当叶子节点是离散型时

Flag=0，当叶子节点是连续型时 Flag=1，非叶子节点 Flag=−1。各个节点间关系的判断与普通区域编码相同。

图 3 给出了图 1 中 P-文档编码结果。编码过程中如果某节点没有经过分布节点，则[Path]项为空。

3.2 处理分布节点

本文使用扩展的区域编码中的[Path]存储某节点经过的分布节点路径，查询时根据所有叶子节点的公共分布路径来处理分布节点。假设某小枝模式有 n 个叶子节点 $T_1, T_2, \cdots, T_n$，在 P-文档中查询时，根据相对应的 n 个叶子节点存储的[Path]路径求出其公共分布节点路径：(1)若公共分布节点路径以奇数结尾，说明它们是同一个互斥节点的孩子，则不能同时出现。(2)若公共分布节点路径以偶数结尾，说明它们是同一个独立节点的孩子，则可以同时出现。

例如 Q' 在图 1 的 P-文档匹配时，叶子节点 B_1 和 C_1 经过的分布节点路径分别为：B_1. Path=1,3 和 C_1. Path=1,0，其共同经过的最后一个公共分布节点是 1，说明它们是标记为 1 的互斥节点的孩子，不可能同时出现。而叶子节点 B_3 和 C_1 经过的分布节点路径分别为：B_3. Path=1,0 和 C_1. Path=1,0，它们共同经过的最后一个公共分布节点是 0，说明它们是标记为 0 的独立节点的孩子，有可能同时出现。

3.3 计算查询结果的概率

每一个小枝匹配都是以一定概率值出现的，此概率值可以通过存储的 Pr 计算而来。扩展的区域编码中 Pr 存储根节点到此结点的路径概率。对应于有 n 个叶子节点 $T_1, T_2, \cdots, T_n$ 的小枝查询模式，n 个叶子节点的最低公共祖先 LCA (Lowest Common Ancestor) 表示为 $LCA(T_1, T_2, \cdots, T_n)$，(1)若最低公共祖先是一个互斥结点，此小枝模式的概率是 0；(2)否则，小枝模式的概率为：$\Pr(Q)=\Pr(T_1)*\Pr(T_2)*\cdots*\Pr(T_n)/(\Pr(LCA(T_1,T_2,\cdots,T_n)))^{n-1}$。

3.4 非归并 ProTwigList 算法

对于查询树中的每一个节点，ProTwigList 算法简单维持一个队列，利用队列以及各队列中的元素指针来保存完整的小枝模式匹配结果。构造队列时把可能参与结果的节点存入队列中，在输出结果时根据分布节点的性质得到符合要求的查询结果并计算每个结果的概率值。

算法 1 ProTwigList

输入：查询树 Q 和 P-文档中各节点对应的 Tag+Level 流；

输出：所有的查询结果及其概率；

```
1. Tag+Level streamPruning;//无用标签流剪枝
2. initialize stack S as empty;
3. initialize all list as empty,and the length of each list is initialized as 0;
4. while(not all Tag+Level =∅)
5.    let Ve be the node,whose type and level equal
6.    to the first node in the preorder traversal;
7.    let e be the smallest top element in X^L_e;
8.    Remove e from X^L_e;
9.    toList(S,Q,reg(e));
10.    for each child of Ve in Q,Vp,do
11.       e.start_Vp←length(L_Vp)+1;
12.    push(S,e);
13. toList(S,Q,(∞,∞));
14. output Result according to the lists and evaluate the probabilities;
Procedure toList(S,Q,r)
15. While(S≠∅ and r⊄reg(top(S))) do
16.    f←pop(S);
17.    Let f's type be Vf;
18.    for each child of Vf in Q,Vk,do
19.       f.end_Vk←length(L_Vk);
20.    append f into list if f.start_Vk≤f.end_Vk for
21.    every f's child,Vk;every f's child,Vk;
```

算法的 4−12 行是执行循环，直到所有的 X^L_v 为空，其中 5−8 行是根据先序遍历 P-文档树的顺序从 Tag+Level 流中选取一个节点。9 行调用 toList 函数，此函数首先判断某节点与栈顶元素节点是否符合祖先后代关系，若不满足则弹出栈顶元素并判断弹出的元素是否加入到相应的队列。13 行弹出最后的栈中的所有元素节点并判断是否加入到相应的队列。最后，输出所有查询结果并计算其相应的概率值。

4 算法复杂度分析

假定通过引理 1 的剪枝后，P-文档中剩余的普通节点数量为 $|T_{left}|$，小枝模式中共有 N 个查询节点，且查询节点的最大度为 D。

时间复杂度：被引理 1 剪枝后剩余 $|T_{left}|$ 个普通节点，构造队列时每一个节点都要经历一次入/出栈操作，最坏的情况下每一个节点最多需要经历 D 次区间编码的比较来决定其孩子节点是否入栈，此阶段的时间复杂度是 $O(D.|T_{left}|)$。输出结果时，假定最终结果解的个数是 R，小枝模式 N 个节点每一个节点都要输出，故此时的时间复杂度是 $O(N.|R|)$。所以，ProTwigList 算法的时间复杂度是 $O(D.|T_{left}|+N.|R|)$。

空间复杂度：构造队列时最坏情况下的时间复杂度是 $O(D.|T_{left}|)$，此时剪枝后剩余的 $|T_{left}|$ 个节点最后都存入队列中。在输出查询结果时空间复杂度仍然是 $O(D.|T_{left}|)$。所以，ProTwigList 算法的空间复杂度是 $O(D.|T_{left}|)$。

5 实验结果和分析

为准确分析 ProTwigList 算法的性能，本文同时实现了文献[7]中的算法 Holistic Twig in PXML。实验的硬件环境为 CPU Inter(R) Core i3(3.2GHz)，RAM 为 4G，操作系统为 64 位的 Windows 7 旗舰版 SP-1，实验工具为 Eclipse SDK 3.7.1，JDK 6.0。

实验数据集合采用真实数据集 DBLP[11]，在数据集上随机插入分布节点，并分别进行了 5 个不同的小枝模式查询测试，实验用到的查询用例见表 1。

表 1 实验中用到的小枝模式查询用列表

编号	小枝模式
Q1	dblp//article[title]//year
Q2	dblp//aiticle[author]//title
Q3	dblp//masterthesis[title]//year
Q4	dblp//article[author][//title]//url
Q5	dblp//masterthesis[//title]//author

实验 1 文档大小固定(文档大小为 44.3M)，小枝查询模式变化(分别执行 5 个查询 Q1−Q5)，对比两个算法的查询效率，实验结果如图 4 所示。实验结果表明在相同的文档下，分别执行不同的查询语句，ProTwigList 算法的查询效率

(下转第 228 页)

gorithm[J]. Information Sciences,2008,178(22):4205-4227
[3] Liu Jing-wei, Xu Mei-zhi. Kernelized fuzzy attribute C-means clustering algorithm [J]. Fuzzy Sets and Systems, 2008, 159 (18):2428-2445
[4] Graves D, Pedrycz W. Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study[J]. Fuzzy sets and systems,2010,161(4):522-543
[5] Li M J, Ng M K, Cheung Y-M, et al. Agglomerative fuzzy K-Means clustering algorithm with selection of number of clusters [J]. IEEE Transactions on Knowledge and Data Engineering, 2008,20(11):1519-1534
[6] Bobrowski L, Bezdek J C. c-means clustering with the L_1 and L_∞ norms[J]. IEEE Trans on System, Man and Cybernetics, 1991, 21(3):545-554
[7] Hathaway R J, Bezdek J C, Hu Y. Generalized fuzzy c-means clustering strategies using Lp norm distances[J]. IEEE Trans on Fuzzy Systems,2000,8(5):576-582
[8] Weinberger K Q, Saul L K. Distance metric learning for large margin nearest neighbor classification[J]. J of Machine Learning Research,2009,10(2):207-244
[9] 叶斌,胡修林,张蕴玉,等. 基于 3D Zernike 矩的三维地形匹配算法及性能分析[J]. 宇航学报,2007,28(5):1241-1245
[10] Zhang D, ChenS. A comment on alternative c-means clustering algorithms[J]. Pattern Recognition,2004,37(2):173-174
[11] Bezdek J C. A convergence theorem for the fuzzy ISODATA clustering algorithms [J]. IEEE Transactions on Pattern Anal Machine Intel,1980,2(1):1-8
[12] 钱国红,黄德才. 基于 3D 角度编码的量子遗传算法[J]. 计算机科学,2012(8):242-245
[13] 张愿章. Young 不等式的证明及应用[J]. 河南科学,2004,22 (1):23-29
[14] 西奥多里德斯,等. 模式识别[M]. 北京:电子工业出版社,2006
[15] 张葛祥,李娜,金炜东,等. 一种新量子遗传算法及其应用[J]. 电子学报,2004,32(3):476-479
[16] Yang Yi-ming. An evaluation of statistical apporachs to text categorization[J]. Journal of Information Retrieval, 1999, 1(1/2): 67-88

(上接第 200 页)

都高于 Holistic Twig 算法,这主要因为 ProTwigList 算法不需要归并中间结果,且在查询前使用 Tag+Level 流剪枝能提前减少需要处理的节点数。

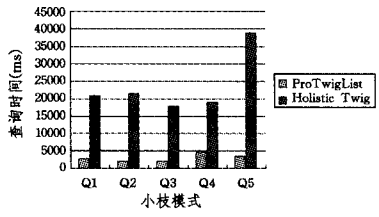

图 4 查询比较(1)

实验 2 文档大小变化,小枝模式固定(以 Q1 为例),对比两算法的查询效率,实验结果如图 5 所示。实验结果表明随着文档大小的逐渐增加,两算法的运行时间都会增加,但 ProTwigList 算法仍优于 Holistic Twig 算法,同样证明了 ProTwigList 算法的查询效率。

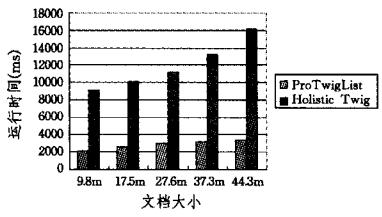

图 5 查询比较(2)

结束语 本文提出了一种非归并不确定 XML 小枝模式查询算法:ProTwigList 算法,该算法提前减少待处理节点的数目,小枝模式查询时使用队列存储可能参与结果的元素节点,输出查询结果并根据最低公共祖先节点计算各结果的概率值。理论分析和实验结果证明了 ProTwigList 算法的性能。下一步将针对包含 NOT、OR 等谓词的不确定 XML 复杂小枝模式查询算法展开研究。

参 考 文 献

[1] 周傲英,金澈清,王国仁,等. 不确定性数据管理技术研究综述[J]. 计算机学报,2009,32(1):1-16
[2] 李文凤,彭智勇,李德毅. 不确定性 Top-k 查询处理[J]. 软件学报,2012,26(3):1-19
[3] Li Jian, Deshpande A. Ranking Continuous Probabilistic Datasets[C]// Proceeding of the 36th International Conference on Very Large Data Bases. Singapore: VLDB Endowment,2010:13-17
[4] Nierman A, Jagadish H V. ProTDB: Probabilistic data in xml [C]// Proceeding of the 28th Very Large Data Bases. Hong Kong: VLDB Endowment,2002:29-41
[5] Kimelfeld B, Sagiv Y. Matching Twigs in Probabilistic XML[C]// Proceeding of the 33th International Conference on Very Large Data Bases. Vienna: VLDB Endowment,2007:23-28
[6] Li Ya-wen, Wang Guo-ren, Xin Juan-chang. Holistically Twig Matching in Probabilistic XML[C]// Proceedings of the 25th International Conference on Data Engineering. Shanghai: IEEE, 2009:1649-1656
[7] Liu Si-qi, Wang Guo-ren. Boosting Twig Joins in Probabilistic XML[C]// Proceeding of the 22ed International Conference on Database and Expert Systems Applications. France: Springer-Verlag,2011:51-58
[8] Ning Bo, Liu Cheng-fei, Yu J X, et al. Matching Top-k Answers of Twig Patterns in Probabilistic XML[C]// Proceeding of the 15th International Conference on Database Systems for Advanced Applications. Tsukuba: Springer-Verlag,2010:125-139
[9] Nierman A, Jagadish H V. Probabilistic data in XML[C]// Proceeding of the 28th international conference on Very Large Data Bases. Hong Kong: Springer-Verlag,2002:646-657
[10] Lu Jia-heng, Meng Xiao-feng, Ling T W. Indexing and querying XML using extended Dewey labeling scheme [J]. Data & Knowledge Engineering,2011,70(1):35-39