第41卷 第6期 西 安 交 通 大 学 学 报 Vol.41 №6
2007年6月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Jun. 2007

基于本体的多源异构XML数据近似查询方法

田磊，覃征，衡星辰，邵利平
（西安交通大学电子与信息工程学院，710049，西安）

摘要：提出了一种基于本体的面向多源异构可扩展置标语言(XML)的近似查询方法.该方法将传统的基于查询模式树松弛的近似查询策略与基于本体的XML数据集成思想相融合，首先建立文档类型定义结构与全局模式的映射模型(PTO)，再将PTO与近似查询领域的松弛操作、打分机制相结合，提出了一种新的XML近似查询算法——OAXQ. 这样，用户在全局模式下的查询语言不再是Xpath查询表达式，而是对象查询语言的一个简单子集，松弛的产生不再依靠查询模式树的变换，而是通过一系列映射规则转化而成.汽车外型的智能设计实验表明，OAXQ算法的查询速度比单纯基于查询模式树松弛策略的静态有序选择算法平均提高了20倍，查询准确度提高了2%～24%.

关键词：可扩展置标语言；文档类型定义；松弛；本体
中图分类号：TP312 **文献标识码**：A **文章编号**：0253-987X(2007)06-0702-05

Approximate Query Approach Based on Ontology for Multi-Source and Heterogeneous XML Data

Tian Lei, Qin Zheng, Heng Xingchen, Shao Liping
(School of of Electronics and Information Engineering, Xi′an Jiaotong University, Xi′an 710049, China)

Abstract: An ontology-based approach to effectively process approximate query for multi-source and heterogeneous XML data is proposed, in which the traditional approximate XML query strategy using tree pattern relaxation is integrated with the idea of ontology-based XML data integration. Firstly, a mapping model(PTO) from document type definition(DTD) structure to global schema is constructed, then PTO is combined with relaxation and scoring mechanism and a novel algorithm—OAXQ is presented. Thus, user query language based on global schema becomes a simple subset of object query language instead of Xpath expressions, and relaxations are accomplished by a series of conversion of mapping rules instead of transform of query tree patterns. Experiments of intelligent design of automobile shape show that comparing with SSO algorithm the query speed and precision is increased by 20 times and 2%-24% respectively by OAXQ

Keywords: extensible markup language; document type definition; relaxation; ontology

近年来，伴随着可扩展置标语言(XML)成为Internet上发布和交换数据的标准，针对XML半结构化数据的查询问题逐渐成为当今国内外信息检索领域的研究热点.目前，有关XML近似查询算法主要是基于查询重写策略[1-3]、计划松弛策略[4-6]和数据松弛策略[7].其中，比较有代表性的算法包括动态有序罚分算法(DPO)[2-3]、静态有序选择算法(SSO)[4,6]等.随着查询数据规模的扩大，遵循多类的文档类型定义(DTD)或Schema的XML文档查询将逐步成为主流，但却没有很好地解决大规模多源异构XML文档集的查询问题.

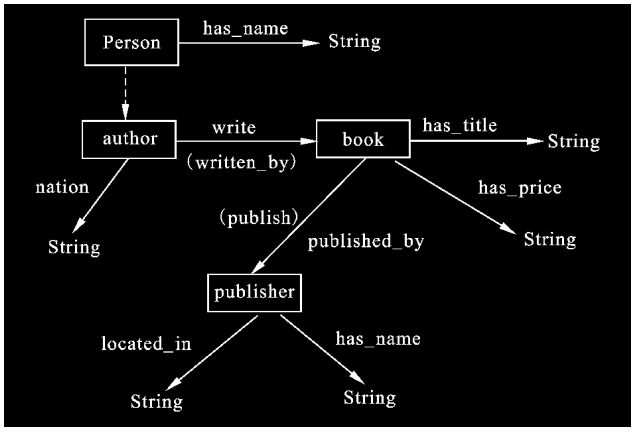
本文设计、构造了适用于多源异构XML数据

收稿日期：2006-10-16. 作者简介：田磊(1983～)，男，硕士生；覃征(联系人)，男，教授，博士生导师. 基金项目：国家重点基础研究发展规划资助项目(2004CB719401).

近似查询的映射模型(PTO),并在此基础上提出了生成 Top-K[8] 个查询结果的新算法——OAXQ,它能较好地解决多源异构 XML 文档集的近似查询问题.

1　基于本体的 XML 数据集成

给出某一特定领域的若干 XML 异构数据源,每种结构的 XML 文档遵循一类 DTD,XML 数据集成能够使用户以统一的模式查询这些异构数据源中的信息.这是通过定义全局模式与各个数据源结构之间的映射来实现的,因为用户的查询是基于全局模式提出的.文献[9]提出了一种基于本体的 XML 数据集成方案,该方案用本体描述全局模式,用路径-路径的映射将全局模式与数据源模式关联起来.下面以图 1(一个特定领域的 XML 数据集成系统的全局模式)为例来说明文献[9]中的方案.在本例中,有 S_1 和 S_2 2 个 XML 数据源,它们的 DTD 及映射规则集如图 2 所示.



方框:概念;实线箭头:角色或属性;虚线箭头:继承(isa)关系;String:属性的类型;圆括号:逆角色

图　1　全局模式

假设用户在全局模式下提出查询 Q_1,查询作家 J. K. 罗琳的《哈利波特》出版地,则有

Q_1:select c
from book a, a. has_title b,
　a. published_by. located_in c,
　a. written_by. has_name d
where b="哈利波特"and d="J. K. 罗琳"

具体到数据源 S_1,用户原始查询语句将被转化为查询 $Q_1(a)$,即

$Q_1(a)$:select c
from url1/book a,
　a. /title b, a. /publisher. /location c,
　a. /info. /author. /@ name d

```
<!ELEMENT Book(title, info, publisher)>
<!ELEMENT Title#PCDATA>
<!ELEMENT info(price, author+)>
<!ELEMENT price#PCDATA>
<!ELEMENT author#PCDATA>
<!ELEMENT publisher(location)>
<!ATTLIST  publisher name CDATA#REQUIRED>
<!ELEMENT location#PCDATA>
R1:url1/book as u1 -> book
R2:u1/title as u2 -> has_title
R3:u1/info/price as u3 -> has_price
R4:u1/publisher as u4 -> published_by
R5:u4/location as u5 -> located_in
R6:u1/info/author as u6 -> written_by
R7:u6/@name as u7 -> has_name
```

(a)数据源 S_1

```
<!ELEMENT author(book+)>
<!ATTLIST author name CDATA#REQUIRED>
<!ELEMENT book(info)>
<!ATTLIST book title CDATA # REQUIRED>
<!ELEMENT info(price, publisher)>
<!ELEMENT price#PCDATA>
<!ELEMENT publisher(location)>
<!ATTLIST publisher name CDATA # REQUIRED>
<!ELEMENT location # PCDATA>
R1:url2/author as u1 -> author
R2:u1/book as u2 -> write
R3:u2/@title as u3 -> has_title
R4:u2/info/price as u4 -> has_price
R5:u2/info/publisher as u5 -> published_by
R6:u5/location as u6 -> located_in
R7:u5/@name as u7 -> has_name
```

(b)数据源 S_2

图 2　DTD 及映射规则

where b="哈利波特"and d="J. K. 罗琳"

显然,$Q_1(a)$很容易转化成 XQuery 表达式.

2　基于本体的 XML 近似查询

2.1　映射模型

考虑文献[9]中提到的映射规则,其左侧是一条 Xpath 路径,右侧是全局模式中相关的角色,本文设计、构造了适用于 XML 近似查询的映射模型 PTO,其结构为

$$R:(v_1,v_2)\rightarrow c_1.\text{role},\quad c_2.\text{inverse(role)}\mid c_3.\text{attribute}$$

式中:(v_1,v_2)是 DTD 蕴涵的二元结构化谓词,它表示父子关系或祖先后代关系;c_1. role 是一条包含单一角色的模式路径[9];c_2. inverse(role)是与 c_1. role 互补的逆角色模式路径,其中 c_2 是 c_1. role 代表的

概念，inverse(role)表示 role 的逆角色[9]；c_3. attribute 由概念 c_3 和单一属性组成；符号“|”是或者的意思，表示对于每一条规则，角色和属性只能出现一种. 具体到数据源 S_1 的 DTD，构造其基于模型 PTO 的映射规则集如图 3 所示.

```
R1: (book, title) ->book.has_title
R2: (book, price) ->book.has_price
R3: (book, author) ->book.written_by, author. write
R4: (book, publisher) ->book.published_by, publisher. publish
R5: (publisher, location) ->publisher.located_in
R6: (publisher, name) ->publisher.has_name
```

图 3 对数据源 S_1 构造的 PTO 映射规则集

2.2 近似查询算法 OAXQ

文献[6]利用了传统的基于查询模式树松弛的近似查询策略，并且给出了基于该策略的静态有序选择(SSO)算法. 与文献[6]的区别在于：本文引入了基于本体的 XML 数据集成思想来处理多源异构 XML 数据的近似查询，用户在全局模式下提出查询，使用的语言不再是 Xpath 查询表达式，是对象查询语言(OQL)的一个简单子集，松弛也不再依靠查询模式树的变换来产生，而是通过一系列映射规则转化来完成.

定义 1 给定某一特定领域的 XML 文档集，若其中包含多个数据源，并且不同数据源中的 XML 文档遵循不同种类的 DTD，则该 XML 文档集中的数据是多源异构的.

定义 2 用户在全局模式下的查询语句，通过一系列基于模型 PTO 的映射规则转化为某一特定 DTD 蕴涵的二元结构化谓词集，则用户查询语句结构化松弛为该 DTD.

定义 3 给定一条包含单一角色的模式路径 c_1. role，与其互补的逆角色模式路径为 c_2. inverse(role)，它们对应相同的模式路径谓词频，即

$$f_t(\mathrm{pr}) = \sharp \mathrm{pr}/(\sharp c_1 * \sharp c_2)$$

式中：pr 为模式路径对应的二元结构化谓词，有 4 种形式，分别为 pc(c_1,c_2)、pc(c_2,c_1)、ad(c_1,c_2)和 ad(c_2,c_1)，其中 pc 代表父子关系，ad 代表祖先后代关系. 同时，用 $\sharp i$ 代表元素 i 在整个 XML 文档集中出现的次数.

定义 4 给定一条包含单一角色的模式路径 c_1. role，与其互补的逆角色模式路径为 c_2. inverse(role)，它们对应相同的模式路径逆文档频，即

$$f_{id}(\mathrm{pr}) = \lg(|D|/|\{n \in D: \mathrm{match}(\mathrm{pr},n)\}|)$$

式中：$|D|$为文档集的总文档数；n 为 XML 文档集 D 中的某个文档；pr 为模式路径对应的二元结构化谓词；match(pr,n)代表 pr 在文档 n 中可匹配；$|\{n \in D: \mathrm{match}(\mathrm{pr},n)\}|$为在 D 中可以匹配 pr 的文档数.

定义 5 给定一条包含单一角色的模式路径 c_1. role，与其互补的逆角色模式路径为 c_2. inverse(role)，它们对应相同的模式路径罚分，即

$$F(c_1.\mathrm{role}) = F(c_2.\mathrm{inverse}(\mathrm{role})) = f_t(\mathrm{pr}) * f_{id}(\mathrm{pr})$$

定义 6 给定一条映射规则 r：$(v_1,v_2) \rightarrow c_1$. role，$c_2$. inverse(role)，其对应的罚分为

$$d(r) = [2 - \mathscr{H}(v_1, c_1) - \mathscr{H}(v_2, c_2)]\, f_t(\mathrm{pr})$$

式中：$\mathscr{H}(i,j) \in [0, 1]$代表概念 i 与概念 j 的语义相似度，它由专家指定.

定义 7 给定一个全局模式 G，R 为 G 到某一特定 DTD 的映射规则集，P 为用户查询语句映射到该 DTD 后丢掉的模式路径集，则该 DTD 相对于用户查询语句的结构化松弛得分为

$$M_S = M - \sum_{p \in P} F(p) - \sum_{r \in R} d(r)$$

式中：M 表示完全匹配得分，由用户指定；$F(p)$表示模式路径 p 的罚分；$d(r)$表示映射规则 r 的罚分.

定义 8 给定一个包含值节点 value 的二元结构化谓词(v_1,value)和 D 中的一个含有节点 v_1 的 XML 文档 n，则 n 因无法精确匹配该值节点谓词而产生的匹配罚分为

$$M_v = [1 - \max\{\mathscr{H}(\mathrm{value}, v_2)\}](|v_1|/|n|)$$

式中：$|v_1|$表示文档 n 中包含 v_1 节点的个数；$|n|$表示文档 n 的总节点数；v_2 满足$\{v_2 \in n |\ \mathrm{pc}(v_1,v_2)\}$，而 $\mathscr{H}(\mathrm{value},v_2)$的含义在定义 6 中给出.

定义 9 给定某一特定 DTD 相对于用户查询语句的结构化松弛得分 M_S，V 是用户查询语句中包含的值节点谓词集，则遵循该 DTD 的 XML 文档 n 的近似得分为

$$M_a = M_S - \sum_{v \in V} M_v$$

其中 M_v 在定义 8 中给出.

至此，给出基于映射模型 PTO 的多源异构 XML 数据近似查询算法 OAXQ，其步骤描述如下.

算法 1 XML 近似查询算法 OAXQ.

输入：DTD 聚簇的多源异构 XML 文档集，对应于每一类 DTD 的 PTO 映射规则集.

输出：Top-K 个近似查询结果.

步骤 1：使用 PTO 映射规则集将全局模式下的用户查询语句转化为具体的针对每一类 DTD 的结

构化谓词集和值节点谓词集，并在此过程中计算每一类 DTD 对应于用户查询语句的结构化松弛得分，下转步骤 2.

步骤 2：将 DTD 按结构化松弛得分由高到低排序，首先对得分最高的 DTD 聚簇的 XML 数据源文档进行精确匹配，而在精确匹配过程中通过计算值节点谓词匹配罚分得到该 DTD 聚簇的每一个 XML 文档的近似得分，并将查询结果放入集合 S 中按近似得分由高到低排序，下转步骤 3.

步骤 3：按顺序用下一个 DTD 的结构化松弛得分对集合 S 进行截取，如果集合 S 中近似得分值超出该 DTD 结构化松弛得分的 XML 文档数大于 K，返回前 K 个结果，结束；否则，下转步骤 4.

步骤 4：对该 DTD 聚簇的每一个 XML 文档进行精确匹配，将新得到的查询结果按近似得分由高到低的顺序插入到集合 S 之中，上转步骤 3.

3　实验及结果分析

3.1　实验环境

实验配置为：Pentium Ⅳ 1.6 GHz，256 MB 内存，Windows XP 操作系统. 原型系统是按国家“九七三”智能化汽车外型特征搜索要求设计的，并采用 VC++6.0 实现后台汽车外型部件的 XML 文档查询. 在实验中，使用的 XML 样本数据由自行开发的样本发生器按照不同种类的 DTD（汽车外型种类）生成，样本库中的 XML 文档根据 DTD 按类聚簇.

3.2　实验结果及分析

将 OAXQ 算法与 SSO 算法[6]进行查询速度和查询准确度的比较. 查询速度主要反映查询算法的效率，查询准确度主要反映用户对查询结果的满意度.

若指定 XML 样本总文档数为 1 000，返回结果数 $K=50$（Top-K 中的 $K=50$），当 XML 文档库中不同种类 DTD 的个数 $x=1,5,10,15$ 时，SSO 算法和 OAXQ 算法的查询时间如表 1 所示. 结果表明，OAXQ 算法的查询效率明显优于 SSO 算法，并且随着 DTD 种类的增加其优势更加显著. 当全部 XML 文档都遵循单一 DTD 时，SSO 算法和 OAXQ 算法的查询时间相差不大，因为 OAXQ 算法利用 DTD 在结构上对 XML 文档聚簇的性质来优化精确匹配，省去了对大量二元结构化谓词的联接操作，使得查询时间比较少；随着 DTD 种类的增加，SSO 算法需要对整个文档库进行遍历，尤其当结构上完全匹配的 XML 文档数达不到 K 时，遍历可能会进行多次，而 OAXQ 却利用 DTD 对 XML 文档聚簇的特点，优先考虑结构化松弛得分高的文档区域，避免了对整个文档库进行盲目遍历，从而速度优势更为显著.

表 1　SSO、OAXQ 算法的查询时间比较

算法	查询时间/ms			
	$x=1$	$x=5$	$x=10$	$x=50$
SSO	93	1 547	1 602	8 797
OAXQ	63	71	79	94

若指定 XML 样本总文档数为 500，返回结果数 $K=20$（Top-K 中的 $K=20$），当 XML 文档库中的 $x=1,5,10,15$ 时，SSO 算法和 OAXQ 算法的查询准确度如图 4 所示. 结果表明，OAXQ 算法在查询准确度上稍优于 SSO 算法，并且随着 DTD 种类的增加，优势逐步明显. 影响查询准确度的关键是，XML 近似查询中的松弛操作和打分机制，SSO 算法依赖于闭包谓词罚分驱动的查询模式树松弛，其中谓词罚分基于整个 XML 文档库的统计值，而 OAXQ 算法基于映射模型 PTO，扩展了传统的 $f_t * f_{id}$ 方法来定义模式路径罚分，从而使松弛操作在语义层次上完成，语义上的查询结果准确度更高. 由于每一类 DTD 都对应有各自的映射规则集，所以 OAXQ 算法的查询准确度基本不受 XML 文档库中的 DTD 个数的影响.

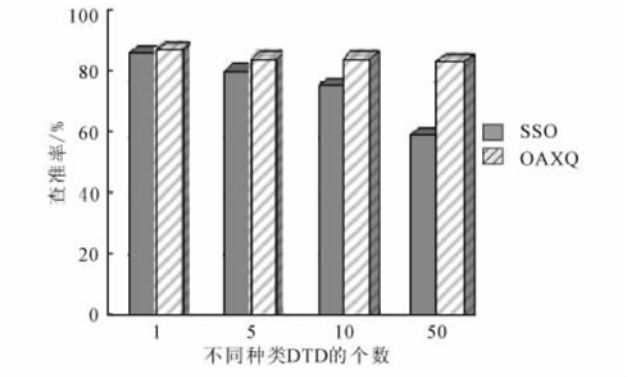


图 4　查询准确度比较

4　结　论

本文提出了一种基于本体的面向多源异构 XML 数据的近似查询方法. 首先设计、构造了适用于 XML 近似查询的映射模型，接着将映射模型与 XML 近似查询领域的松弛操作、打分机制相结合，进而提出了一种新的 XML 近似查询算法 OAXQ. 经国家“九七三”汽车外型智能化设计项目的实验表

明，OAXQ算法在查寻准确度、查询速度这2个方面明显优于单纯基于查询模式树松弛的静态有序选择算法SSO[6]. 下一步将开展XML数据挖掘研究，挖掘XML文档库内部的语义信息，并在专家的适当参与下，半自动地生成全局模式本体图，并基于统计信息建立语义相似词库，以辅助计算映射规则罚分和值节点谓词匹配罚分.

参考文献：

[1] Chinenyanga T T, Kushmerick N. Expressive and efficient ranked querying of XML data [C]// 4th International Workshop on the Web and Databases. New York: ACM Press, 2001:1-6.

[2] Delobel C, Rousset M. C. A uniform approach for querying large tree-structured data through a mediated schema [EB/OL]. [2006-02-10]. http://gemo.futurs.inria.fr/publications/GEMO-PUBLI/display-abstract.php? id=241.

[3] Schlieder T. Similarity search in XML data using cost-based query transformations [C]// 4th International Workshop on the Web and Databases. New York: ACM Press, 2001:19-24.

[4] Amer-Yahia S, Cho S, Srivastava D. Tree pattern relaxation [C]// 8th International Conference on Extending Database Technology. Berlin: Springer-Verlag, 2002:496-513.

[5] Kilpelainen P. Tree matching problems with applications to structured text databases, A-1992-6 [R]. Helsinki, Finland: University of Helsinki, 1992.

[6] Amer-Yahia S, Lakshmanan L V, Pandit S. FleXpath: flexible structure and fulltext querying for XML [C]// The ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2004: 83-94.

[7] Damiani E. The APPROXML tool demonstration [C]// 8th International Conference on Extending Database Technology. Berlin: Springer-Verlag, 2002:753-755.

[8] Marian A, Amer-Yahia S, Koudas N, et al. Adaptive processing of Top-*K* queries in XML [C]// 21st IEEE International Conference on Data Engineering. Los Alamitos, USA: IEEE Computer Society, 2002:162-173.

[9] Amann B, Beeri C, Fundulaki I, et al. Querying XML sources using an ontology-based mediator [C]// 10th International Conference on Cooperative Information Systems. Berlin: Springer-Verlag, 2002:429-448.

（编辑 苗凌）