计算机科学2001Vol.28№.12

基于代价的 XML 文档关系模式的生成*)

The Cost-Based Generation of Relational Model for XML Documents

高 军 唐世渭 杨冬青 王腾蛟

(北京大学计算机科学技术系 北京100871)

(北京大学视觉与听觉信息处理国家重点实验室 北京100871)

Abstract It is one of feasible ways to use the fully-developed database management system to manage the XML data. How to generate the corresponding relational schema is a key problem to reach that target. Here described is an approach to generate the relational schema according to the structure of DTD and the analysis of user query for XML data. In the view of the drawbacks of approach put forward by Jauavel Shanmugasundaram, this approach takes both the storage cost and query cost into consideration.

Keywords DTD graph, XML, Query cost model, Relational schema

1. 引言

作为目前信息表示和交换的标准,XML 得到越来越广泛的应用。对 XML 进行管理,传统的方法是利用文件系统。目前,传统的关系数据库在市场上仍占有主流地位,如何利用关系数据库来管理 XML 数据成为现实的问题。由于 XML 文档本质上是基于图模式的半结构化数据,而目前商用数据库管理系统管理的是基于关系模式的结构化数据,利用关系数据库来管理 XML 数据可能带来非常高的存贮代价和查询代价,因此利用关系数据库来管理 XML 数据必须解决两种异构模式之间的转换问题,生成合理的关系模式。

对于上述问题,工业界和学术界从不同的角度进行了研究,文[6]Oracle 8i 利用关系数据库引擎来完成 XML 文档的简单查询,但是,Oracle 8i 的当前版本要求手工生成 XML 文档所对应的关系模式,增加了使用人员的负担。

文[1]中提出利用关系数据库管理半结构化数据的一种解决方案,XML 文档采用边表的形式进行存贮,产生出的结构简单,适应于任何形式的 XML 文档实例。但是该方法没有利用现有的 DTD,而且并没有解决关于存贮代价和查询代价高的问题。

文[3]中提出了在不存在 DTD 约束 XML 文档情况下的处理方式。该方法首先进行的是模式提取,生成大于某一支持度的关系模式,对于不符合所生成关系模式中的 XML 数据,在关系数据库中采用文[1]中的方法直接存贮半结构化数据,最终得到的模式是混合模式。上述结果导致利用关系数据库管理 XML 实现复杂,代价高。在文[3]得出的结论中,该方法只适应于结构化程度高的 XML 文档。

文[2]提出了一种遵循 DTD 的 XML 文档实例关系模式的生成方式。该方法利用 DTD 描述构造 DTD 图,基于 DTD 图的分析,提出了三种生成对应关系模式的方法,Basic,Shared,Hybrid。上述方法较好地解决了利用关系数据库管理 XML 数据的存贮问题。但是,上述方法在生成模式时只考虑了 XML 文档的存储代价,未考虑 XML 文档的查询代价。

我们目前承担了国家重点基础研究发展规划资助的"网络环境下海量信息组织与处理的理论与方法研究"中的"面向内容的海量信息集成、分析处理与服务"的研究课题,COMMIX(Content-Oriented Massive inforMation Integration based on XML)是课题组开发的原型系统。其功能是集成网络环境下海量的、异构的信息源,为用户提供统一的视图和各类信息服务。

COMMIX 系统中信息集成层,需要对 XML 查询结果进行物化保存。本文介绍了 COMMIX 中提出的一种新的关系数据库管理半结构化数据的一种方法。在研究了目前不同解决方法的结论和存在的不足后,

*)本文研究得到国家重点基础研究发展规划(973)资助,编号 G1999032705。高 军 博士研究生,研究方向为数据库与信息系统。唐世渭 教授,博士生导师,研究方向为数据库与信息系统。杨冬青 教授,博士生导师,研究方向为数据库与信息系统。王腾蛟 博士研究生,研究方向为数据库与信息系统。

对于文[2]中提出的方法做出改进。利用本文提出的局部查询代价比较模型，根据获取的用户查询模式，生成考虑用户查询代价的关系模式。

2. 相关概念

2.1 简化 DTD

DTD 相当于对应的 XML 文档的模式。其基本的单位是元素，元素之间可以利用下列操作符作连接：'*'(多个元素，可以为空)，'+'(至少一个元素)，'?'(最多有一个元素)，'|'(或)。

DTD 所描述的实际上是一种嵌套的关系模式，在 DTD 的元素定义中，子元素之间存在顺序，相同的子元素可能多次出现。但是在关系模式中不能描述重复出现的属性；在 DTD 定义中子元素可能有多个层次，但是在传统关系模式中不能描述嵌套结构，而对应该元素的关系可能包含所有的子元素信息，所以，利用关系模式表示 DTD 需要去掉其元素定义中子元素的层次和重复的子元素，根据文[2]中定义的下列规则，我们令'%'表示'*'或者'?'，则化简规则如下：

(e1,e2)*->e1*,e2*,(e1,e2)?->e1?,e2?,(e1|e2)->e1?,e2?;
e1*%->e1* e1??->e1*;
…e1*,…e1%…->e1*,…,…e1%,…e1*…->e1*,…e1?,…e1?…->e1*,……e1,…e1…->e1*.

例1 (!ELEMENT a((b|c|e)?,(e?|(f?,(b,b)*))*)利用上述规则化简为(!ELEMENT a(b*,c?,e*,f*))。关系模式描述上述元素时，对于子元素的'?'操作符的处理和'*'操作符的处理是不同的。对于带有'?'操作符的子元素，可以作为关系模式中一个允许为空的属性存在。对于带有'*'操作符的子元素，在传统的非嵌套关系中不能以属性的方式加以表达，只能单独对应关系，通过外键和父元素建立的关系作连接。

2.2 构造 DTD 扩展树

根据文[2]得到 DTD 图的形式化描述：

定义1 DTD 为有向图 G=(N,E)，其中集合 N 对应简化后的 DTD 中的元素，包括操作符'*'，'?'，集合 E∈N*N，表示 DTD 图中的边集，描述了 DTD 种元素的结构关系，如果 n∈N，则 in(n)表示元素 n 的入度，out(n)表示 n 的出度；如果 n∈N，m∈N，存在 e∈E，e=(n,m)，则 f(n,m)成立，f(n,m)表明元素 n 是元素 m 的父亲。

例2 DTD 描述(部分)

```
(!element ADDRESS(street,zip))
(!element FACTORY (owner,address))
(! element RESTAURANT (address, owner,
          waiter*))
(!element OWNER (name,age))
```

例3 例2中所对应的 DTD 图(图1)

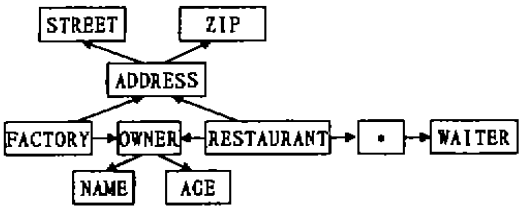


图1

在 DTD 图中不存在根的概念，如果 XML 文档符合某个 DTD，则所对应的 XML 文档可以以该 DTD 图中任何元素节点为根。

如何存贮上述 DTD 所对应的 XML 文档，文[2]中给出了三种方法：①Basic 方法，DTD 中的任何一个元素都建立一个关系与之对应，每一个元素的关系定义包含其元素数中的所有元素；②Shared 方法。特定的元素建立对应的关系，通过外键建立关系之间的联系；③Hybrid 方法，关系的建立上和 Shared 一致，但是在关系中的属性上不同。

从空间代价的角度来看，采用 Basic 的方法建立关系模式是不可接受的，但是对于查询代价，文[2]中的实验数据表明，采用 Shared 方法，可能导致大量的连接操作，采用 Basic 方法，可能需要大量的 UNION 操作，很难说明何种关系模式的查询代价低。

上述的实验数据表明，要生成一个合理的关系模式，不但应当考虑其存储代价，而且应当考虑具体的应用环境。

定义2 扩展的 DTD 图为有向图 G=(N,E,M)，其中 N 集合和 R 集合与定义1中一致，如果 DTD 所对应的关系模式为 S，则 M 表示元素所对应关系的所包含的属性。

半结构化数据的查询一般采用路径查询的形式，查询结果为路径中的节点信息。如果利用关系来管理上述数据，需要将路径查询转换到关系查询上。上述要求，表明在 DTD 中应当记录元素和关系模式的对应关系，上述对应关系在关系模式建立的过程中加以确定。

3. 局部查询代价比较模型

半结构化数据存在共同的特点[7]，数据是可以看成带标记有向图的形式。目前 XML 查询有多种查询语言，它们的具体表现形式不同，但主要是利用正则路径表达式来表达查询。利用关系数据库管理 XML 文档需要将上述的路径查询表达式转化到具体的关系查询上。本文以 XML-QL[4]为例，找到半结构化查询所涉及的关系查询，在关系模式生成阶段，如果仅仅根据

DTD 结构无法得出最优模式时，就根据不同模式所对应的查询代价来选择关系模式。

3.1 XML-QL 描述

文[4]中 XML-QL 查询语言的查询路径语法生成形式如：

```
.....
Where::=`where' condition(`,',condition)
Condition::=pattern bindings * `in' datastore
Pattern:=starttagpattern pattern * endtag
Starttagpattern::=`<'regularexpression attribute * `>'
Regularexpression::=regularexpression ` * ' | regularexpression `+' | regularexpression `.' regularexpression | regularexpression `|' regualrexpression |(var)|(id)
Attribute::=(id)`='(`"'(STRING)`"'|(VAR))
.....
```

上述所选的查询语言的产生式和 XML 的路径表达式有关。

上述定义中，路径表达式可以表达路径标签变量(VAR)、路径闭包 * 和常规的路径表达式。在路径表达式中，存在4种连接符：'*'表示路径重复多次，可以为空；'+'表明路径至少重复一次；'.'表明路径的常规连接；'|'表明路径的选择。

上述的路径表达式，必须转化到确定的路径上，才能对应到相应的关系中。由于存在路径标签变量和路径闭包的运算，根据路径表达式获取路径的空间复杂性不是多项式的，但是，路径表达式所表达的路径中的中间元素节点最多为 DTD 中的元素数目，路径中中间元素节点的获取也是多项式时间能够完成的。

3.2 局部查询代价比较模型

在原有 DTD 中或在中间过程中产生的入度大于1、出度为0的元素 n，根据文[2]中的方法，可以单独对应关系，也可以将该元素作为每个父元素对应关系的属性，或采用这二种方法混合的办法。

定义3 对于用户的查询语句 Q，查询路径表达式为 p，DTD 图中的元素为 n，在 DTD 图中构造元素 n 的树时，如果在路径表达式 p 的表达路径的中间元素集合是 n 的元素树中元素集合的子集，则称上述查询 Q 相对于元素 n 为全局查询；如果中间元素集合不是元素树中元素集合，但交集不为空，则称查询 Q 相对于元素 n 为特定路径查询。

如果元素 n 作为其父元素对应关系的属性，则特定路径的查询效率高；但是上述元素的全局查询，则需要所有父元素对应关系查询结果 UNION 操作；而且同一个元素数据，可能在不同的父元素所对应关系中产生多个副本，关系的更新操作困难。

如果采用混合的方法，主要问题是所有的数据至少保存两次，造成存贮空间的浪费，降低了处理的效率，增加了更新的复杂性，增加了关系模式中对应关系的数目，但是元素 n 的查询，包括全局查询和特定路径的查询效率有提高。从关系数据库来管理 XML 数据的空间存贮代价和数据更新的角度看，不采用混合存贮的方法。

如果对于元素 n 单独建立对应关系，则该元素的全局维护和全局查询效率提高，但是特定路径查询，必须采用该元素对应关系和其特定路径的父元素所对应关系作连接的方式，查询的效率低。而且，上述元素单独建立关系，可能在最终关系模式中产生大量的关系。

在确定上述元素所对应的关系模式时，仅仅依靠 DTD 图中的结构是不够的。如果获取用户的查询方式，分析用户的查询，在几种可能的关系模式中，可选择查询代价相对较小的模式。

本文中提出局部查询代价比较模型，主要确定 DTD 图中入度大于1、出度等于0的元素 n 所对应的关系模式。在本文中，确定标准是根据用户的查询模式在不同的关系模式下的估算代价。

DTD 图是有向图，根据路径表达式产生确定的路径集合的空间复杂性不是多项式的，但是在有向图两个节点之间的路径中中间节点个数最多是 DTD 图的元素数目，上述中间节点的获取，也可以在多项式时间完成。所以，在本文，判断特定查询是否是全局查询，是根据路径中中间元素集合和元素 n 形成的元素树元素集合的关系来确定的。

算法1：局部查询代价比较

输入：DTD 中元素 n，输入查询语句集 $Q=\{(q_1,w_1),(q_2,w_2),\cdots,(q_n,w_n)\}$，其中，$q_i$ 是输入的查询语句，w_i 是查询语句 q_i 的权重，cost_join 是 JOIN 代价常数，cost_union 是 UNION 代价常数。

输出：$COST_R(n)$ 和 $COST_A(n)$，其中 $COST_R(n)$ 表示当元素 n 对应关系时的查询代价，$COST_A(n)$ 表示当元素 n 对应属性时的查询代价。

具体步骤：

1. 在 DTD 图中构造以元素 n 为根的树；
2. 任何一个查询语句 q_i，构造其中间路径节点集合；
3. 如果上述集合是元素 n 形成的树中的节点集合的子集，则上述查询为全局查询；
4. 如果上述集合和元素 n 形成的树中的节点集合的交集不为空，并且判断3不成立，则上述查询为特定路径查询；
5. 如果相对于元素 n 为全局查询，则 $COST_A(n)=COST_A(n)+w_i * cost\_union * in(n).out(n)$ 表示元素 n 的出度，相当于其父元素对应表的个数；
6. 如果相对于元素 n 为特定路径查询，则 $COST_R(n)=COST_R(n)+w_i * cost\_join$。

3.3 代价比较模型讨论

代价比较模型的提出，是为了在关系模式的生成阶段量化地完成查询代价的估计。仅仅依靠 XML 查询语句不可能完成对应的关系模型中的代价准确分析。本文中的代价模型，是一个局部的估算模型，在算法实现复杂性不高的前提下，主要能够确定 DTD 的某个具体元素的处理方式。

查询语句组和其权值的获取，可以通过用户查询要求的追踪记录统计完成。如在电子商务网站中，可以通过 Web 日志文件得知用户的常用查询要求和同类查询出现频率。

如果在DTD中存在n个元素，则以某一个元素为根形成的树最多含有n个元素。在查询语句中，中间路径元素的获取，是图扫描的过程，复杂性为$O(n^2)$，利用线性时间完成全局查询还是特定路径的查询，如果存在p条查询语句，则全部的运算复杂性为$O(pn^2)$。

上述的代价比较模型，只是近似结果，在代价计算过程中，只考虑了关系运算中的时间复杂性较高的JOIN运算和UNION运算，而且代价的计算是局部查询的代价估计。

4. 用DTD和查询语句组生成相应的关系模式

4.1 根据化简DTD产生最小的收缩图

产生最小收缩图的目的是在DTD图中确定建立对应关系的元素，DTD图中的元素节点，当入度$in(n)=0$时，表明该元素n不可能经过其它的元素到达，所以入度为0的元素一定产生对应关系；当入度$in(n)=1$时，表明存在且仅存在一个元素m，$f(m,n)$成立，则元素n，考虑存贮代价，将元素n作为其父元素生成关系的属性；上述的元素的处理，仅依靠DTD的结构即可完成。当$in(n)>=1$时，存在多种处理方式，利用第3节中采用的代价比较模型来选择查询估算代价小的关系模式。

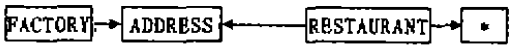
定义4 如果不存在节点n，m，$in(m)=1$，$f(m,n)$，并且所有节点k，如果$out(k)=0$，则对于所输入的查询语句组Q，$COST_R(k)>COST_A(k)$，则称该DTD图为最小收缩图。

算法2：根据DTD图得到DTD最小收缩图

输入：DTD图，相应的XML文档实例，查询语句集合Q
输出：DTD的最小收缩图
1. 找到在DTD图中元素n的集合N，其中$out(n)=0$并且n不是操作符元素。
2. 上述N中任何元素n，如果$in(n)=1$，则找到唯一的节点m，使得$f(m,n)$，否则，转STEP6，
3. 如果父节点m不是操作符节点，该节点标记删除，在其父节点中的M中加入该节点信息，$M[m]=M[m]\cup\{n\}$，
4. 如果父节点为'?'，找到祖父节点k，在其祖父节点中加入该属性$M[k]=M[k]\cup\{n\}$，该节点n和其父节点标记删除。
5. 如果父节点为'*'，根据DTD图的构造，可知其祖父节点k不可能为操作符节点，找到其父节点m，加入该节点的属性$M[m]=M[m]\cup\{n\}$，该节点n和其父节点标记删除，转STEP7。
6. 计算元素n的局部查询代价，如果$COST_R(n)<COST_A(n)$，则对于元素n建立关系与之对应，如果$COST_R(n)>COST_A(n)$，对于任意$m\in H$，$f(m,n)$，$M[m]=M[m]\cup\{n\}$。
7. 重复步骤1，如果不存在满足条件的节点，则退出。

其中，第3、4、5步是入度为0、出度为1的元素的处理，第6步是入度为0出度大于1的元素的处理。

例3 可能的最小收缩图（和所输入的查询有关）



4.2 根据最小收缩图生成关系模式

算法3：根据最小收缩图建立关系

输入：DTD最小收缩图。
输出：DTD所对应的关系。
1. 任意非操作符号元素节点n，生成一关系R，并产生其唯一的关键字id(n)。
2. 根据元素节点的M集合，确定元素节点所对应的关系的属性。
3. 节点n在收缩图中该节点的父节点m，在节点n中所对应的关系中追加节点m所对应关系的关键属性id(m)，在关系模式中建立两个关系之间的外键联系。
4. '*'符号节点，生成一关系，如果父节点为f，子节点为s，则生成关系R{id(f),id(s)}。

其中第2步建立相应的关系，在第3，4步建立不同类型关系的连接。

例4 例3所对应的关系模式可能为（和输入的含有权值的查询语句组有关）：

ADDRESS(**addressid**,street,zip,factoryid,restaurantid)
FACTORY (**factoryid**, owner.isroot, owner.name, owner.age,production)
RESTAURANT (**restaurantid**, owner.name, owner.age, food)
RESTAURANT.WAITER(**waiter**,**restaurantid**)

如果输入的查询中存在大量的对于OWNER的特定路径和查询，则根据局部查询代价比较模型，确定OWNER作为其父元素FACTORY和RESTAURANT的属性。如果输入的查询中存在大量的对于ADDRESS的直接查询，则同样根据局部查询代价模型，确定ADDRESS元素产生单独的关系。

4.3 整体方法流程

我们的方法和文[2]中方法的主要区别是，本文考虑了用户的查询模式，利用局部查询代价模型，对于可能的关系模式进行筛选，得出的关系模式不但考虑了管理XML的存贮代价，而且考虑了查询代价。

由上述的算法可知，完成DTD的化简，生成最小收缩图，查询代价的判断，生成关系模型，其算法的复杂性都是多项式时间可以完成的。

参考文献

1 Florescu D, Kossman D. A Performance Evaluation of Alternative Mapping Schemes for Storing XML Data in a Relational Database Rapport de Recherche No. 3680 INRIA, Rocquencourt, France, May 1999
2 Shanmugasundaram J, et al. Relational Database for Querying XML Documents: Limitation and Opportunities. In: Proc. of the 1999 VLDB Conference, Sep. 1999
3 Fernandez D M, Suciu D. Storing Semistructured Data with STORED. In: Proc. of the ACM SIGMOD Conference, Philadelphia, Pennslyvania, May 1999
4 Fernandez D M, et al. XML-Q1: A Query Language for XML. http://www.w3.org/TR/NOTE-xml-ql
5 McHugh J, et al. A Database Management System for Semistructured Data SIGMOD Record, 1997, 26(3): 54～66
6 Corporation O. XML Support in Oracle 8 and beyond. Technical white paper. http://www.oracle.com/xml/documents
7 Buneman P. Semistructured data. In: Proc. of the 16th ACM SIGACT SIGMOD SIGART sym on principles of Database systems(PODS'97). 1997. 99～108