用于 XML 模式和 DTD 规范化设计的层次模式设计*

吴永辉[1,2+]

[1](复旦大学 计算机科学与工程系,上海 200433)

[2](中国科学院 软件研究所 计算机科学重点实验室,北京 100080)

Hierarchical Schemas Design for XML Schemas and DTDs Normalization Design

WU Yong-Hui[1,2+]

[1](Department of Computer Science and Engineering, Fudan University, Shanghai 200433, China)

[2](Key Laboratory of Computer Science, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: +86-21-55664413, Fax: 86-21-65642820, E-mail: slwu@fudan.edu.cn, http://www.cs.fudan.edu.cn

Received 2003-05-21; Accepted 2003-11-27

Wu YH. Hierarchical schemas design for XML schemas and DTDs normalization design. *Journal of Software*, 2004,15(7):1099~1106.

http://www.jos.org.cn/1000-9825/15/1099.htm

Abstract: Normalization design of XML Schemas and DTDs (document type definitions) is to produce a set of XML schemas or DTDs that can well represent data dependencies and eliminate redundancies. Now there are a few researches on it, and the existing researches are still at its initial stage. Provost proposed the idea of applying the theory of relational database to XML schemas normalization design. This idea has not been put into practice. The paper shows algorithms of hierarchical schemas design for XML schemas and DTDs normalization design based on Provost's idea. Firstly the paper analyzes hierarchy decomposition based on Provost's idea. Then it presents an algorithm producing a decomposition tree to eliminate redundant schemas. Finally it shows an algorithm of hierarchical schemas design for XML schemas and DTDs normalization design to get over deficiencies for Provost's idea. With respect to other researches on normalization design for XML schemas and DTDs, the set of full and embedded MVDs in hierarchical schemas produced by these algorithms are implied by the given set of MVDs (multivalued dependencies), and the hierarchical schemas eliminate redundant ones and satisfy the lossless join property.

Key words: XML schema; document type definition (DTD); hierarchical schema; multivalued dependency (MVD); attribute; simpletype element

摘 要: XML 模式和 DTD(document type definition)规范化设计是给出一个很好地表示数据间依赖关系并消除了冗余的 XML 模式或 DTD 的集合.目前在这一方面开展的研究还不多,而且才刚起步.Provost 提出将关系数

* Supported by the National Natural Science Foundation of China under Grant No.69933010 (国家自然科学基金); the Foundation of Laboratory of Computer Science, Institute of Software, the Chinese Academy of Sciences under Grant No.SYSKF0306 (中国科学院软件研究所计算机科学实验室开放课题基金)

作者简介: 吴永辉(1966－),男,浙江温岭人,博士,副教授,主要研究领域为数据库,数字图书馆.

据库理论应用于 XML 模式规范化设计的思想,这一思想还没有付诸实施.在 Provost 思想的基础上给出用于 XML 模式和 DTD 规范化设计的层次模式设计的算法.首先分析了基于 Provost 思想的层次分解;然后给出用于消除冗余模式的分解树设计算法;最后给出用于 XML 模式和 DTD 规范化设计的层次模式设计算法,这一算法克服了 Provost 思想的缺陷.相应于其他 XML 模式和 DTD 规范化设计的研究,在算法产生的层次模式中,完全 MVD(multivalued dependency)和嵌入 MVD 的集合由给出的 MVD 集合导出;并且产生的层次模式具有消除冗余模式和满足无损联接的特性.

关键词: XML 模式;文档类型定义(DTD);层次模式;多值依赖(MVD);属性;简单类型元素

中图法分类号: TP393 **文献标识码**: A

XML 已经成为网上数据表示和交换的主要工具,而建立一个 XML 应用的第一步通常是设计 XML 模式或 DTD(document type definition),给出能很好地表示数据间依赖关系并消除了冗余的层次结构.然而,不同于关系模式规范化设计,目前 XML 模式和 DTD 设计缺乏理论基础.

对于 XML 模式和 DTD 规范化设计,现在开展的研究还不多,而且才刚起步,主要工作有:Provost 提出将关系数据库理论应用于 XML 模式规范化设计的思想[1],这一思想还没有付诸实施;Arenas 和 Libkin 给出 XML 函数依赖的概念,定义 XML 模式的范式 XNF,并提出将一个任意的 DTD 转化为一个符合 XNF 的 DTD 的算法,这一算法是通过移动属性和建立新的元素类型,以实现 DTD 的规范化设计[2];Ling 和 Lee 等人定义半结构化模式的范式 S3-NF,NF-SS 和 ORA-SS,给出将一个半结构化模式转化为一个满足某个范式的 XML 模式的算法,这一算法是通过规范化规则对半结构化模式进行重新构造,以实现 XML 模式的规范化设计[3~6].

因为 XML 文档与面向对象数据库有着一些相似的数据特性;XML 模式和 DTD 是嵌套的层次结构,并且层次间的数据依赖通常用多值依赖(multivalued dependency,简称 MVD)表示.所以面向对象模式规范化设计、嵌套关系模式规范化设计和基于 MVD 集合的关系模式规范化设计与 XML 模式和 DTD 规范化设计存在着一定的关联.目前在面向对象模式规范化设计方面开展的研究也不多,有代表性的工作有:Tari 等人提出全局依赖和对象范式的定义,给出全局依赖的推导公理,并给出规范化规则和导出对象范式的算法[7];Mok 等人则给出了一个实用的基于嵌套范式(NNF)的 OODB 设计过程:将面向对象数据模型用超图表示,并将超图转化为 NNF[8].此外,我们也对复杂对象模式的规范化设计进行了比较深入的研究,定义复杂对象模式的冗余和范式,给出并验证复杂对象模式规范化设计算法,并分析对象依赖集合的性质以及对规范化设计的影响[9,10].而嵌套关系模式规范化设计的理论基础是 Roth 和 Korth 提出的能在进行嵌套化和非嵌套化操作中保持嵌套关系特性的划分范式(PNF)[11,12]和 Ozsoyoglu 和 Yuan 提出的具有消除冗余和保持依赖特性的嵌套范式(NNF)[13,14].在关系数据库理论中,存在基于 MVD 集合产生 4NF 的算法[15].

目前的 XML 模式和 DTD 规范化设计的研究存在着一些共同特点:以函数依赖(FD)表示数据间的依赖关系;规范化设计算法通过对原模式的重构,产生一个符合某一范式的模式.尽管这样能够消除某些冗余,但也带来一些问题:首先,规范化设计算法对原模式进行重新构造,会导致模式的原有语义发生改变;其次,在有些情况下,仅用一个模式可能无法很好地表示数据依赖和消除冗余,需要将原来的一个模式分解为几个模式;第三,XML 结构复杂,仅用 FD 不能充分地表示 XML 模式和 DTD 中数据间的语义关系,特别是层次间的数据依赖通常用 MVD 表示.此外,不仅在现有的 XML 模式和 DTD 规范化设计的研究中,而且在比较成熟的嵌套关系模式规范化设计的研究成果中,都没有对冗余进行专门的研究和分类.因而目前的 XML 模式和 DTD 规范化设计没有很好地消除冗余和体现数据间的依赖关系.

Provost 提出将关系数据库理论应用于 XML 模式规范化设计的思想[1].由于 XML 数据模型复杂,将关系数据库理论应用于 XML 模式和 DTD 的设计会导致一些问题.本文分析基于 Provost 思想的层次分解,并给出用于 XML 模式和 DTD 规范化设计的层次模式设计算法.

1 层次分解和表示 XML 模式和 DTD 的层次模式

XML 模式和 DTD 是层次结构.由于层次间的数据依赖用 MVD 表示[11,12],根据 Provost 的思想[1],基于 MVD 的层次分解定义如下:

定义 1(层次分解). 设 $O,X_0,X_1,X_2,\ldots,X_k$ 是由简单类型元素(不包含属性和其他元素的元素)和属性组成的集合 ,$X_0\subset O,X_i\cap X_j=\varnothing(0\le i,j\le k,i\ne j)$. 如果在 $O\cup X_1\cup X_2\cup\ldots\cup X_k$ 上 MVD$X_0\rightarrow\rightarrow X_1|X_2|\ldots|X_k$ 成立,则称 $\{X_0X_1\cap O, X_0X_2\cap O,\ldots,X_0X_k\cap O,O-X_1\cup X_2\cup\ldots\cup X_k\}$为在 O 上相应于 $X_0\rightarrow\rightarrow X_1|X_2|\ldots|X_k$ 的层次分解.

基于 Provost 的思想[1]和定义 1,给出简单类型元素和属性的集合 O 和 O 上的 MVD 集合 M,层次分解的过程如下:

输入:简单类型元素和属性的集合 O 和 O 上的 MVD 集合 M;

步骤:

/*步骤 1:初始化*/

产生 M 的划分$\{M_1,M_2,\ldots,M_m\}$,使得在每个 M_i 中的 MVD 有相同的 LHS;设 $M_i=\{X_{i0}\rightarrow\rightarrow X_{i1}\,|\,X_{i2}\,|\ldots|\,X_{ik_i}\}$, $1\le i\le m$;

设图 G 由一个标号为集合 O 的结点构成;

Z:=$\{O\}$;

/*步骤 2:层次分解*/

For i:=1 To m Do

For 每个 $z\in Z$ Do

If $X_{i0}\subset z$ Then

{将 z 对应的结点的标号改为 X_{i0};

For j:=1 To k_i Do

If ($X_{i0}\subset X_{i0}X_{ij}\cap z$) Then

{在 G 中加入一个标号为 $X_{i0}X_{ij}\cap z$ 结点 v;

以标号为 X_{i0} 的结点为起点,结点 v 为终点作弧;

Z:=$Z\cup\{X_{i0}X_{ij}\cap z\}$;

};

If $(z-X_{i1}\cup X_{i2}\cup\ldots\cup X_{ik_i}\ne\varnothing)$ Then

{在 G 中加入一个标号为 $z-X_{i1}\cup X_{i2}\cup\ldots\cup X_{ik_i}$ 的结点 v';

以标号为 X_{i0} 的结点为起点,结点 v'为终点作弧;

$Z:=Z\cup\{z-X_{i1}\cup X_{i2}\cup\ldots\cup X_{ik_i}\}$;

};

Z:=$Z-\{z\}$;

};

输出:表示层次模式的图 G.

我们参照嵌套范式的定义[14],用层次模式来表示图 G 所示的层次分解结果,定义如下:

定义 2(层次模式). 设 O 是一个简单类型元素和属性的集合,层次模式 $H(O)$表示为有向无环图 $DAG=(N,E,f)$;f 是从结点集 N 到 O 的幂集的一个内射,以表示结点的标号,对于弧集 E 中的弧$\langle u,v\rangle$,$f(u)\rightarrow\rightarrow d(v)$,其中 $d(v)=f(v)\cup f(v_{d1})\cup\ldots\cup f(v_{dm}),v_{d1},\ldots,v_{dm}$ 是 v 的所有后继.

显然,对于由层次分解产生的层次模式,如果 $v_1,v_2,\ldots,v_n$ 是层次模式中的所有结点,则 $f(v_1)\cup f(v_2)\cup\ldots\cup f(v_n)=O$;并且层次模式中的弧表示的完全多值依赖和嵌入多值依赖的集合由 M 导出.

在层次模式中,层次模式树定义如下.

定义 3(层次模式树). 在层次模式 $H(O)$中,以入度为 0 的结点为根,并包含该结点所有后继的有根树被称为

层次模式树.

一棵层次模式树相应于一个 XML 模式或 DTD.首先,自底向上地逐层在每个结点标号对应的集合中删去在其父结点标号对应的集合中也包含的简单类型元素和属性;然后,再自底向上地逐层为每个结点赋名:如果是叶结点,并且结点标号只包含一个简单类型元素或属性,则直接用该简单类型元素或属性命名该结点.否则,为该结点赋一个复杂类型元素名,该元素包含该结点标号中的简单类型元素和属性及其子结点的名称.这样,一棵层次模式树就可以转化一个 XML 模式或 DTD 的树的表示形式.

层次模式的定义没有涉及冗余问题.在模式中,最常见和最明显的冗余是冗余模式:相同内容以相同的结构重复出现.对此,我们定义层次模式中的冗余模式如下:

定义 4(层次模式中的冗余模式). 设 $H(O)$是层次模式,如果 $H(O)$中存在两棵互不相交的,并且包含出度为 0 结点的有根子树 T_A 和 T_B,使得 T_A 包含的简单类型元素和属性的集合是 T_B 的子集,而且 T_B 表示的 MVD 是 T_A 表示的 MVD 的嵌入 MVD,则我们称在层次模式 $H(O)$中存在冗余模式.

如果在给出的 MVD 集合 M 中存在这样的两个 MVD $X\rightarrow\rightarrow Y$ 和 $Z\rightarrow\rightarrow W$,$Z$ 相应于 $X\rightarrow\rightarrow Y$ 是可分解的(即 $X\subset Z$,$Z\cap Y\neq\varnothing$,$Z-X-Y\neq\varnothing$),并且 $W\subset Y$,则相应于 $Z\rightarrow\rightarrow W$ 的分解会产生两个包含 Z 的简单类型元素和属性集合,而后相应于 $X\rightarrow\rightarrow Y$ 进行分解,在产生的集合族中会有一个集合是另一个集合的子集.

因此,如果按 Provost 的思想[1]进行层次模式设计会产生冗余模式;而产生冗余模式的原因是,在给出的 MVD 集合中,存在这样的两个 MVD:$X\rightarrow\rightarrow Y$ 和 $Z\rightarrow\rightarrow W$,$Z$ 相应于 $X\rightarrow\rightarrow Y$ 是可分解的,并且 $W\subset Y$.

此外,层次分解还存在其他问题:首先,产生的层次模式仅由一棵层次模式树组成,即所有的元素和属性都包含在一个 XML 模式或 DTD 文件中,这使得产生的 XML 结构比较混乱;其次,产生的层次模式的结构基于 M 的划分的序列,不同的序列导致不同的层次模式.

2 消除冗余模式的分解树设计

设 O 是简单类型元素和属性的集合,M 是 O 上的 MVD 集合.

引理 1. 如果 $X\rightarrow\rightarrow Y\in M$,$Z\rightarrow\rightarrow W\in M$,而且 $Y\in DEP(X)$,$W\in DEP(Z)$,并且相应于 $X\rightarrow\rightarrow Y$,$Z$ 是可分解的,则

(1) $W\cap Y$ 或者是 W,或者是$\varnothing$;

(2) $X(Y\cap Z)\rightarrow\rightarrow Y\cap W$.

证明:

(1) 假设存在这样的 $W\in DEP(Z)$,使得 $W\cap Y\subset W$ 并且 $W\cap Y\neq\varnothing$.因为 Z 相应于 $X\rightarrow\rightarrow Y$ 是可分解的,所以 $X\subset Z$;则 $Z\rightarrow\rightarrow Y$;因为 $Z\rightarrow\rightarrow W$,所以 $Z\rightarrow\rightarrow Y\cap W$;由假设,$W\cap Y\subset W$ 并且 $W\cap Y\neq\varnothing$,所以 $W\notin DEP(Z)$.这与 $W\in DEP(Z)$矛盾,所以 $W\cap Y$ 不是 W 就是$\varnothing$.

(2) 因为 $X\rightarrow\rightarrow Y$,所以 $X(Y\cap Z)\rightarrow\rightarrow Y-Z|O-(Y-Z)$.因为 $Z\subset O-(Y-Z)$,所以 $O-(Y-Z)\rightarrow\rightarrow W$,则 $X(Y\cap Z)\rightarrow\rightarrow W-(O-(Y-Z))$.因为 $W\in DEP(Z)$,所以 $Z\cap W=\varnothing$,则 $W-(O-(Y-Z))=W\cap Y$,所以 $X(Y\cap Z)\rightarrow\rightarrow Y\cap W$. □

由引理 1,可推导出如下推论:

推论 1. 如果 $Y\in DEP(X)$,$W\in DEP(Z)$,并且 Z 相应于 $X\rightarrow\rightarrow Y$ 是可分解的;如果 $W\cap Y=W$,则 $W\in DEP(X(Y\cap Z))$.

定理 1. 对于任意的 MVD 集合 M,存在与 M 等价的 MVD 集合 M',在 M'中不同时存在这样的两个 MVD:$X\rightarrow\rightarrow Y$ 和 $Z\rightarrow\rightarrow W$,相应于 $X\rightarrow\rightarrow Y$,$Z$ 是可分解的,并且 $W\subset Y$.

证明:首先,可以产生与 M 等价的 MVD 集合 M_1,使得在 M_1 中的任何 $X\rightarrow\rightarrow Y$,$Y\in DEP(X)$成立.

设 $X\rightarrow\rightarrow Y\in M_1$,$Z\rightarrow\rightarrow W\in M_1$,并且相应于 $X\rightarrow\rightarrow Y$,$Z$ 是可分解的,并且 $W\subset Y$.因为 $Y\cap Z\neq\varnothing$,并且 $Y-Z\neq\varnothing$,所以由推论 1,$W\in DEP(X(Y\cap Z))$.在 M_1 中用 $X(Y\cap Z)\rightarrow\rightarrow W$ 取代 $Z\rightarrow\rightarrow W$,设产生的 MVD 集合为 M_2,则 $M_1^+=M_2^+$.

对 M_2 重复上述过程.因为$|X(Y\cap Z)|<|Z|$,所以这一过程会结束.设 M'是过程结束时产生的 MVD 集合,则 $M'^+=M_1^+=M^+$,并且 M'中不同时存在这样的两个 MVD:$X\rightarrow\rightarrow Y$ 和 $Z\rightarrow\rightarrow W$,相应于 $X\rightarrow\rightarrow Y$,$Z$ 是可分解的,$W\subset Y$.□

在给出用于消除冗余模式的分解树设计算法之前,首先给出函数 TEST,用于判别两个 MVD g_i 和 g_j 是否会导致冗余模式.

Boolean Function TEST(i,j)

{设 g_i 是 $X\rightarrow\rightarrow Y$,$g_j$ 是 $Z\rightarrow\rightarrow W$;
If (Z 相应于 $X\rightarrow\rightarrow Y$ 是可分解的,并且 $W\subset Y$) Then
Return True
Else
Return False;
}

在算法 1 中,$DT(O,M)$表示分解树,$label(n)$为 $DT(O,M)$中结点 n 的标号,表示结点 n 对应的简单类型元素和属性的集合.

算法 1. 消除冗余模式的分解树设计.

输入:简单类型元素和属性的集合 O 和 O 上的 MVD 集合 M;

步骤:

/*步骤 1:对 M 应用定理 1;*/

产生与 M 等价的 MVD 集合 M_1:对于在 M_1 中的任何 $X\rightarrow\rightarrow Y$,$Y\in DEP(X)$;

设 M_1 为$\{g_1,g_2,\ldots,g_k\}$,队列 Q 为$\varnothing$;

For i:=1 To k Do
For j:=1 To k Do
If $TEST(i,j)$ Then 将(i,j)加入到 Q 中;
Repeat
{将 Q 的队头(i,j)移出;
在 M_1 中用 $LHS(g_i)\cup(RHS(g_i)\cap LHS(g_j))\rightarrow\rightarrow RHS(g_j)$取代 g_j;
/*LHS 和 RHS 分别表示 MVD 的左式和右式*/
For l:=1 To k Do
{If $TEST(l,j)$ Then 将(l,j)加入到 Q 中;
If $TEST(j,l)$ Then 将(j,l)加入到 Q 中;
};
}
Until Q 为$\varnothing$;

/*步骤 2:划分和排序;*/

产生 M_1 的划分$\{G_1,G_2,\ldots,G_m\}$,使得在每个 G_i 中的 MVD 有相同的 LHS.设 X_i 为 G_i 中 MVD 的 LHS,$1\le i\le m$;并且如果 $X_i\subset X_j$,则 $1\le i<j\le m$;

/*步骤 3:基于 MVD 集合 M_1 产生分解树;*/

设 $DT(O,M)$在初始时为一个标号为集合 O 的结点;

For i:=1 To m Do
For 每个在 $DT(O,M)$中出度为 0 的结点 n Do
If $label(n)$相应于 M_1 中的 G_i 是可分解的 Then
{Y:=$label(n)$;
$label(n)\longleftarrow X_i$;
设 Z 是 G_i 中的所有 RHS 的并集;
For 每个在$\{X_i(Y\cap W)|Y\cap W\neq\varnothing$并且 $X_i\rightarrow\rightarrow W$ 在 G_i 中$\}\cup\{Y-Z\}$中的元素 X' Do
在 $DT(O,M)$中加入一个新结点 n'和弧$\langle n,n'\rangle$,$label(n')\longleftarrow X'$;
};

输出:$DT(O,M)$.

由算法 1 可知,产生的分解树 $DT(O,M)$是一棵有根树.

由算法 1 可直接推导出引理 2 成立.

引理 2. 如果 n_1 和 n_2 是 $DT(O,M)$中两个不同的结点,并且 $label(n_1)\subset label(n_2)$,则 n_1 是 n_2 的祖先.

引理 3. 如果 n_1 和 n_2 是 $DT(O,M)$中两个不同的结点,则 $label(n_1)\neq label(n_2)$.

证明:假设在 $DT(O,M)$中有两个不同的结点 n_1 和 n_2,$label(n_1)=label(n_2)=X$.根据算法 1,在 n_1 和 n_2 之间不可能存在祖先和后代的关系.设 $L(n)$是集合族$\{Y|Y=label(m),m$ 是以 n 为根的子树的树叶$\}$中全体元素组成的广义并集,则 $X\subseteq L(n_1)\cap L(n_2)$.设 n 是 n_1 和 n_2 的最小的公共祖先,则由算法 1 可知,$L(n_1)\cap L(n_2)\subseteq label(n)$,所以 $X\subseteq label(n)$.因为 n_1 和 n_2 的祖先 n 的标号不可能也为 X,所以 $label(n)\neq X$.如果 $X\subset label(n)$,由引理 2 可知,n_1 和 n_2 是 n 的祖先,所以导致矛盾,则 $X\not\subset label(n)$.因此 $X\subseteq label(n)$不成立.所以在 $DT(O, M)$中不存在这样的两个不同的结点 n_1 和 n_2,使得 $label(n_1)=label(n_2)$. □

由引理 2 和引理 3,定理 2 成立.

定理 2. 如果 n_1 和 n_2 是算法 1 产生的 $DT(O,M)$两个不同的叶结点,则 $label(n_1)\not\subseteq label(n_2)$.

由定理 2,算法 1 产生的 $DT(O,M)$消除了冗余模式,但所有的元素和属性都包含在一棵有根树中,结构比较复杂,需要进一步的处理.

3 消除冗余模式并具有无损联接特性的层次模式设计

在算法 1 的基础上,算法 2 产生层次模式.

算法 2. 消除冗余模式并具有无损联接特性的层次模式设计.

输入:算法 1 产生的 $DT(O,M)$;

步骤:

/*步骤 1:初始化;*/

设 r 是 $DT(O,M)$的根,$CHILDREN(n)$表示 $DT(O,M)$中内结点 n 的孩子的集合;

$LEAVES$←——$\{m|m$ 是在 $DT(O,M)$中的叶结点$\}$;

$H(O)$←——$\{H(m)|m$ 在 $LEAVES$ 中,设此时 $H(m)$是一个标号为 $label(m)$的结点$\}$;

/*步骤 2:产生层次模式;*/

Repeat

{从 $DT(O,M)$中选择一个内结点 n,使得 $CHILDREN(n)\subseteq LEAVES$,令 $H(n)$是一个标号为 $label(n)$的结点;

将 $H(n)$加入到 $H(O)$中;

$LEAVES$←—— $LEAVES$-$CHILDREN(n)$;

For 每个在 $CHILDREN(n)$中的 m DO

If $H(m)$中有标号为 X 的结点,使得 $X\supset label(n)$ Then

{以 $H(m)$中满足 $X\supset label(n)$的最小的 X 对应的顶点为终点,以标号为 $label(n)$的顶点为起点作弧,以此将 $H(m)$合并到 $H(n)$中;

$H(O)$←——$H(O)-\{H(m)\}$;

}

Else

在 $H(n)$中构造标号为 $L(m)$的新结点 $v(L(m)$的定义见引理 3 的证明),并以 v 为终点,标号为 $label(n)$的顶点为起点作弧;

$LEAVES$←—— $LEAVES\cup\{n\}$;

}

Until $LEAVES=\{r\}$;

输出:层次模式 $H(O)$.

由算法 2,产生的层次模式 $H(O)$表示为一个弱连通的有向无环图,且 $H(O)$表示的数据依赖由给出的 MVD 集合 M 导出.

因为嵌套关系可以通过非嵌套化操作转换为 1NF 关系,所以每个 XML 模式或 DTD 相应的 XML 文档可以通过非嵌套化操作转换为一个关系实例,所以要证明算法 2 产生的层次模式满足无损联接的特性,就要证明相应的关系模式具有无损联接的特性.

定理 3. 算法 2 产生的层次模式满足无损联接的特性.

证明:因为在算法 2 产生的弱连通图中没有环,所以对弱连通图中每个入度为 0 的结点 X,设以 X 为根的层次模式树标记为 T_X,$D(X)$表示 T_X 对应的 XML 文档,$R(X)$表示 $D(X)$通过非嵌套化操作产生的关系实例.

首先证明,在层次模式中的任何层次模式树 T_X 对应的 XML 模式或 DTD 满足无损联接.设 $X_1,X_2,\ldots,X_n$ 为 T_X 的叶结点,$T(X_i)$表示 T_X 中从根结点 X 到叶结点 X_i 的有向路,$R(X_i)$表示 $T(X_i)$在 $D(X)$中对应的部分通过非嵌套化操作产生的关系实例,$1\le i\le n$.所以要证明 T_X 对应的 XML 模式或 DTD 满足无损联接,就是要证明 $R(X_1)\bowtie R(X_2)\bowtie\ldots\bowtie R(X_n)=R(X)$.

设 T_X 的高度为 h,T_X 中高度为 $h-1$ 的内结点为 $Z_1,Z_2,\ldots,Z_k$,其中 Z_i 的子女结点为 $X_{i1},\ldots,X_{ik_i}$,$1\le i\le k$;根据层次模式中的依赖关系,$f(Z_i)\rightarrow\rightarrow d(X_{i1})|\ldots\ldots|d(X_{ik_i})$;设 $T(Z_i)$表示 T_X 中从根结点 X 到结点 Z_i 的有向路和 Z_i 的所有后继构成的有根子树,$R(Z_i)$表示 $T(Z_i)$在 $D(X)$中对应的部分通过非嵌套化操作产生的关系实例,由 MVD-JD 规则,$R(X_{i1})\bowtie\ldots\bowtie R(X_{ik_i})=R(Z_i)$.因此,通过逐层向上的归纳,最后可以推导出在 T_X 中,$R(X)$满足无损联接,即 $R(X_1)\bowtie R(X_2)\bowtie\ldots\bowtie R(X_n)=R(X)$.

然后,在产生的弱连通图中任取包含共同子树的两棵层次模式树 T_X 和 T_Y.设共同子树为 T_Z,结点 z 为 T_Z 的树根.根据层次模式中的依赖关系,可以推出 $f(X)\rightarrow\rightarrow d(X),f(Y)\rightarrow\rightarrow d(Y)$,因为由算法 2,$f(x)\subset d(z),f(y)\subset d(z)$,所以由增广律,$d(z)\rightarrow\rightarrow d(X)|d(Y)$,则由 MVD-JD 规则,$R(X)\bowtie R(Y)=R(XY)$.类似地,对弱连通图中包含共同子树的子图进行归纳,可以证明,算法 2 产生的层次模式满足无损联接. □

定理 4. 算法 2 产生的层次模式 $H(O)$消除了冗余模式.

由定理 2,对于算法 2 产生的弱连通的有向无环图中任意两个出度为 0 的结点 v_1 和 v_2,$f(v_1)\not\subseteq f(v_2)$.所以定理 4 成立.

所以,算法 2 产生的层次模式中的数据依赖由给出的简单类型元素和属性集合上的 MVD 集合 M 导出,并具有消除冗余模式和保持无损联接的特性.

4 结 语

相应于其他 XML 模式和 DTD 规范化设计的研究,本文给出的算法基于简单类型元素和属性集合 O 和 O 上的 MVD 集合 M,产生用于表示 XML 模式和 DTD 的层次模式.在产生的层次模式中,完全 MVD 和嵌入 MVD 的集合由 M 导出,并且产生的层次模式具有消除冗余模式和满足无损联接的特性.所以,产生的层次模式能更好地保持了数据依赖关系,并且消除了冗余模式.本文的工作对于基于 Web 的信息系统开发有一定意义.在以后的工作中,我们将对 XML 模式的冗余及其导致的异常进行更深入的分析,并以此定义范式,完善规范化设计算法.

References:

[1] Provost W. Normalizing XML. http:// www.xml.com/pub/a/2002/11/13/normalizing.html

[2] Arenas M, Leonid L. A normal form for XML documents. In: Lucian P, ed. Proc. of the 21st ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (PODS 2002). Madison: ACM Press, 2002. 85~96. http://www.cs.toronto.edu/~marenas/

[3] Wu X, Ling TW, Lee SY, Lee ML. NF-SS: A normal form for semistructured schema. In: Tjoa AM, ed. Proc. of the Int'l Workshop on Data Semantics in Web Information Systems 2001. Yokohama: IEEE Computer Society, 2001. http://www.comp.nus.edu.sg/~lingtw

[4] Wu X, Ling TW, Lee SY, Lee ML, Gillian D. Designing semistructured database using ORA-SS model. In: Tamer M, ed. Proc. of the 2nd Int'l Conf. on Web Information System Engineering. Kyoto: IEEE Computer Society, 2001. 17~28. http://www.comp.nus.edu.sg/~lingtw

[5] Lee ML, Ling TW, Wai LL. Designing functional dependencies for XML. In: Simonas S, ed. Proc. of the VIII. Conf. on Extending Database Technology (EDBT 2002). Prague: Springer-Verlag, 2002. 124~141. http://www.comp.nus.edu.sg/~lingtw

[6] Lee SY, Lee ML, Ling TW, Kalinichenko LA. Designing good semi-structured databases and conceptual modeling. In: Jacky A, ed. LNCS 1728. Paris: Springer-Verlag, 1999. 131~145. http://www.comp.nus.edu.sg/~lingtw

[7] Tari Z, Stokes J, Spaccapietra S. Object normal forms and dependency constraints for object-oriented schemata. ACM Trans. on Database Systems, 1997,22(4):513~569.

[8] Mok WY, Ng YK, Embley DW. Using NNF to transform conceptual data models to object-oriented database designs. Data & Knowledge Engineering, 1998,24(3):313~336.

[9] Wu Y, Zhou A. Research on properties of a set of object dependencies. Journal of Computer Research and Development, 2001, 38(12):1491~1498 (in Chinese with English abstract).

[10] Wu Y, Zhou A. The normal object scheme forest with respect to conflict-free dependencies. Journal of Software, 2002,13 (8):1488~1493 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/1488.pdf

[11] Roth MA, Korth HF. The design of non-1NF relational databases into nested normal form. In: Yuan L, ed. Proc. of the 1987 ACM SIGMOD Conf. San Francisco: ACM Press, 1987. 143~159.

[12] Roth MA, Korth HF, Silberschatz A. Extended algebra and calculus for nested relational databases. ACM Trans. on Database System, 1988,13(4):389~417.

[13] Ozsoyoglu ZM, Yuan L. On the normalization in nested relational databases. In: LNCS 361. New York: Springer-Verlag, 1987. 243~271.

[14] Ozsoyoglu ZM, Yuan L. A new normal form for nested relations. ACM Trans. on Database System, 1987,12(1):111~136.

[15] David M. The Theory of Relational Databases. New York: Computer Science Press, 1983.

附中文参考文献:

[9] 吴永辉,周傲英.对象依赖集合的性质的研究.计算机研究与发展,2001,38(12):1491~1498.

[10] 吴永辉,周傲英.相应于无冲突依赖的规范化对象模式森林.软件学报,2002,13(8):1488~1493. http://www.jos.org.cn/1000-9825/13/1488.pdf

****

庆祝《软件学报》创刊 15 周年纪念活动预告

2004 年 9 月 16 日 北京

http://www.jos.org.cn

2004 年迎来了《软件学报》创刊 15 周年，为纪念《软件学报》所走过的 15 年风雨历程，并感谢长期以来支持《软件学报》工作的老一辈科学家、新一代科学家、历届编委会委员、广大的作者、读者以及主办单位、各级主管单位的领导等，我们将于 2004 年 9 月 16 日在北京举办纪念活动，并同时出版纪念专刊。纪念专刊将安排在《软件学报》2004 年第 9 期和第 10 期上发表。

现将有关事项预告如下：

一、专刊及大会报告内容

1．当前国际国内计算机软件(各学科方向)的发展动态和热点问题

2．有关《软件学报》的题词及照片等

二、重要日期

活动时间：2004 年 9 月 16 日

三、其他

1. 活动形式：纪念活动、表彰突出贡献人和优秀审稿人、大会报告、专题讲座、编委会；

2. 报告人：邀请国外著名大学学者及科学家和国内计算机专家做专题报告；

3. 参加活动人员：编委、特邀嘉宾、被表彰人员；

4. 详细议程待确定之后发布在《软件学报》网站上。