XML 上的函数依赖推理*

谈子敬+, 庞引明, 施伯乐

(复旦大学 计算机与信息技术系,上海 200433)

Reasoning about Functional Dependency for XML

TAN Zi-Jing+, PANG Yin-Ming, SHI Bai-Le

(Department of Computer and Information Technology, Fudan University, Shanghai 200433, China)

+ Corresponding author: Phn: 86-21-55059446, Fax: 86-21-65642219, E-mail: zijingtan@sina.com

http://www.fudan.edu.cn/dep/xinxi.html

Received 2002-05-16; Accepted 2002-11-04

Tan ZJ, Pang YM, Shi BL. Reasoning about functional dependency for XML. *Journal of Software*, 2003,14(9): 1564~1570.

http://www.jos.org.cn/1000-9825/14/1564.htm

Abstract: XML is extended with functional dependencies, which are fundamental to semantic specification. Based on DTD, tree model for XML and path expressions, definitions for node value equality and node set are given. The definition for functional dependency, logical implication and path closure are studied; the satisfiability of functional dependency in a given DTD is proved. A sound and complete set of inference rules is presented, and an algorithm to calculate closure for XML paths is proved.

Key words: functional dependency; XML; logical implication; inference rule

摘 要: 函数依赖是数据语义的重要组成部分,这一概念被引入到 XML 的领域中.基于 DTD,XML 的树模型和路径表达式,对节点的值相等和路径节点集作了定义.在此基础上,研究了 XML 中函数依赖、逻辑蕴涵和路径闭包的概念,证明了函数依赖在给定 DTD 上的可满足性.提出了一个 XML 上函数依赖的正确和完备的推理规则集,并给出了一个用于计算路径闭包的算法.

关键词: 函数依赖;可扩展标记语言;逻辑蕴涵;推理规则

中图法分类号: TP311 **文献标识码**: A

XML 具有跨平台、简单易用等特性,在很短的时间内就获得了广泛的应用,已成为一种通用的数据交换格式.XML 文档很容易表达来自不同源的数据,但是它所提供的语义信息却相对有限.

例 1:下面是一个有关学校情况的 DTD 定义.学校包含一组课程,每门课程有课程号、开始时间和参加这门课的学生列表.学生的信息包括学号和住址,地址由所处的区和街道构成.

〈!ELEMENT school (course*)〉

〈!ELEMENT course (time,student_list)〉

* Supported by the National Natural Science Foundation of China under Grant No.69933010 (国家自然科学基金)

第一作者简介: 谈子敬(1975－),男,上海人,博士生,主要研究领域为数据库理论,XML.

〈!ATTLIST course cno ID #REQUIRED〉

〈!ELEMENT student_list (student*)〉

〈!ELEMENT student (sno,address)〉

〈!ELEMENT address (district,street)〉

我们希望可以表述模式中所具有的以下约束:

(1) 课程号是课程的键;

(2) 学号在整个学校里是惟一的,可以决定学生住址;

(3) 每个学生同一时刻只可以上一门课;

(4) 学生的住址可以决定所处街道;

(5) 学生住址的街道可以决定区.

在上述要求中,约束存在于 XML 文档的树型结构中.在课程里,课程号是惟一标识,可以决定参与的学生.而在学生信息里,学号又是标识,这样的依赖关系是嵌套的.参与约束的不只包括属性类型,还包括元素类型;元素自身可以有复杂的构造,而且可以以集合的方式参与约束(学生列表).所表达的约束不仅仅是键,而且更广泛.这非但超出了传统关系数据库的范畴,也超出了已有的 DTD 和 XML schema 的表述能力.我们希望可以为 XML 提供表述类似约束的能力,引入关系数据库中的概念,称其为 XML 中的函数依赖.函数依赖是数据语义的重要组成部分,对于键的定义、完整性约束、查询优化和数据集成等都有非常重要的作用.

本文第 1 节简介在关系数据库和 XML 领域中相关的研究成果.第 2 节给出形式化的 DTD,XML 文档和路径表达式定义.第 3 节定义 XML 中的函数依赖和逻辑蕴含,证明函数依赖在 XML 文档上的可满足性,并给出一个正确和完备的推理规则集.第 4 节讨论路径闭包的计算.第 5 节对未来的研究方向作出一些展望.

1 相关工作

函数依赖的概念源于关系数据库领域,有一些深入的研究[1,2],还包括一些复杂的模型,如嵌套的函数依赖[3].这些都属于传统的数据库范畴,不适用于像 XML 这种树状结构和路径导航的半结构化文档模型.在半结构化数据领域中,研究集中在路径约束[4~6].这主要是由于半结构化数据的图状结构,导致很多导航路径相互包含.而 XML 文档是一棵树,树上的每一个节点都有由根到达它的惟一路径.XML[7]及其相关的规范仍然处在完善的过程中,DTD,XML data[8]和 XML schema[9]等提供了基本的键定义能力.XPath[10]是一种用于在 XML 文档中进行导航和查询的路径表达语言.

在 XML 的领域中,针对于完整性约束(主要是键),也有一些相应的研究成果.给出 DTD 和一组完整性约束定义,不能判定是否存在同时符合该 DTD 和完整性约束的 XML 文档.即使进一步限制为简单属性,类似的判定也是 NP 完全的,这是文献[11,12]所研究的主要问题.文献[13]在 DTD 的基础上,扩展了键的定义能力,重点是相关键(relative key).文献[14]通过分析两类特殊的路径表达式,脱离模式定义,在文档一级讨论了相应的键的逻辑蕴涵.本文对函数依赖的讨论是基于 DTD 的模式定义所展开的,而且键只是函数依赖的一个特例.文献[15]将 XML 文档映射到关系模式中,讨论了 XML 的范式,给出了一个将任意 DTD 转化为符合范式形式的无损算法.与文献[15]不同,本文主要研究的是函数依赖及其相应的推理规则集,而且在模型上并不寻求 XML 向关系模式的映射.

2 DTD,XML 文档的树模型和路径表达式

2.1 DTD

首先给出 DTD 的形式化定义,这是在文献[7,12]的基础上修改后得到的.

定义 1. DTD 是一个五元组(E,A,M,N,r):

- E 是有限的元素类型集合;
- A 是有限的属性类型集合,且属性和元素不同名;

• 对$\forall e\in E$,$M(e)$称为 e 的元素类型定义,$M(e)=S$(字符串),或者是一个正规表达式α;

$\alpha::=\varepsilon|e'|\alpha|\alpha,\alpha|\alpha^*$.其中$\varepsilon$表示空,$e'\in E$,“|”,“,”和“*”分别代表或、连接和闭包,$\alpha$的字母表是 E 的子集.

• 对$\forall e\in E$,$N(e)$称为 e 的子属性定义,$N(e)\subseteq A$;

• $r\in E$,称为根元素类型;

$\forall e\in E(e\neq r)$,e 一定和 r 相关,其含义是或者 e 出现在 $M(r)$中,或者$\exists e'\in E$,使得 e 出现在 $M(e')$,且 e'和 r 相关;

• $\forall a\in A$,$\exists e\in E$,使 $a\in N(e)$;

• $\forall e\in E$,r 不出现在 $M(e)$中.

若 $e,e'\in E$,$e'\in M(e)$,且在 $M(e)$中不包含 e'的自连接,则称 e'相对于 e 是惟一的.比如 sno 出现在 M(student)的字母表中,且不含自连接.在这种情况下,每一个 student 元素都只包含惟一的 sno 元素.

2.2 XML文档的树模型

定义 2. 符合给定 DTD(E,A,M,N,r)的 XML 的树模型:

• 模型中的基本元素是节点和字符串.下面使用 v 代表节点,V 表示节点集,是有限的.NULL 代表空串,∅表示空集,符号[]表示列表(有序);

• 对$\forall v$,定义函数 name(v)$\in E\cup A$,根据 name(v),进一步定义两个 V 的子集:

$$Ve=\{v|v\in V,\text{name}(v)\in E\},$$

$$Va=\{v|v\in V,\text{name}(v)\in A\};$$

• 对$\forall v\in Ve$,定义函数 subelem(v)=[$v_1,v_2,\ldots,v_n$]($v_i\in Ve$).

若 name(v)=e,$M(e)=\alpha$,则 name(v_1),name(v_2),…,name(v_n)属于α定义的正规集;

对$\forall v\in Ve$,定义函数 attr(v)={$v_1,v_2,\ldots,v_m$}($v_i\in Va$).若 name(v_i)=a_i,name(v)=e,则 $a_i\in N(e)$;

• 对$\forall v$,定义函数 value(v)$\in\{S\}$;

• 有且仅有一个特殊的节点,记为根节点 root,name(root)=r;

• 函数满足以下要求:

$\forall v_1\neq$root,$\exists v\in V$,$v_1\in$subelem(v)$\cup$attr(v),且 v 惟一,

$\forall v_1,v_2\in$attr(v)($v_1\neq v_2$),name(v_1)$\neq$name(v_2),

$\forall v_1\in$attr(v),$\forall v_2\in$subelem(v),name(v_1)$\neq$name(v_2).

“相等”是函数依赖中的一个重要概念,在 XML 文档树上,我们定义如下.

定义 3. 节点值相等.当满足以下条件时,称节点 v 和 v'值相等,记为 $v\equiv v'$.

(1) name(v)=name(v'),value(v)=value(v');

(2) 若 $v\in Ve$,subelem(v)=[$v_1,v_2,\ldots,v_n$],subelem(v')=[$v'_1,v'_2,\ldots,v'_n$],则 $v_i\equiv v'_i$ 对于 $i\in[1,n]$都成立;

(3) 若 $v\in Ve$,attr(v)={$v_1,v_2,\ldots,v_m$},attr(v')={$v'_1,v'_2,\ldots,v'_m$},则对$\forall v_i$,$\exists v'_j\equiv v_i$,反之亦然($i,j\in[1,m]$).

2.3 路径表达式

在 XML 树上,若从 v 到 v_1 有树上的一条边,我们称 v 是 v_1 的父节点,v_1 是 v 的子节点.若存在序列($v_0,v_1,\ldots,v_n$),其中 v_i 是 v_{i-1} 的子节点,则称 v_0 是 v_n 的祖先节点.将 v_{i-1} 到 v_i 的边依次连接,构成树上的惟一的从 v_0 到 v_n 的路径,记为 v_0–v_n.DTD 可以决定符合它的 XML 文档上的路径形式,这就是路径表达式的概念.DTD 上的一个路径表达式可能对应文档中的若干条路径.也就是说,给定起始节点,经由路径表达式所能到达的节点是一个集合.

定义 4. D 上的路径表达式 P 为 $\rho_1/\rho_2.../\rho_n$ 的形式.其中 $\rho_i\in E\cup A$,且对$\forall i\in[2,n]$,$\rho_i\in M(\rho_{i-1})$的字母表$\cup N(\rho_{i-1})$.若 $\rho_1=r$,则称 P 为根路径表达式.

定义 5. 路径的节点集.设有 D 上的路径表达式 $P=\rho_1/\rho_2.../\rho_n$ 和符合 D 的 XML 树上的元素节点 v,当 $\rho_1\in M(\text{name}(v))$的字母表$\cup N(\text{name}(v))$时,使用记号$\langle v\{P\}\rangle$来表示路径的节点集.其意义为:若树上有节点序列($v_0,v_1,...,v_n$)($v_i$ 是 v_{i-1} 的子节点,$v_0=v$),且 name(v_i)=ρ_i($i\in[1,n]$),则 $v_n\in\langle v\{P\}\rangle$.特别地,若 $\rho_1=r$,则使用记号$\langle P\rangle$.

定义 6. 路径表达式的连接.P,Q 都是 D 上的路径表达式,P 为 $\rho_1/\rho_2.../\rho_n$,Q 为 $\rho'_1/\rho'_2.../\rho'_m$,则当 $\rho'_1\in M(\rho_n)$的字母表$\cup N(\rho_n)$时,$\rho_1/.../\rho_n/\rho'_1/.../\rho'_m$ 也是路径表达式,称为 P 和 Q 的连接,记为 P/Q.设 $R=P/Q$,则我们称 R 包含 P,P

为 R 的前缀.借用集合的符号,记为 $P\subseteq R$.

定义 7. 设 $\rho_1/.../\rho_n/\rho'_1/.../\rho'_m$ 是路径表达式,令 $P=\rho_1/\rho_2.../\rho_n$,$Q=\rho'_1/\rho'_2.../\rho'_m$.当 ρ'_i 相对于 ρ'_{i-1} 是惟一的($i\in[2,m]$),且 ρ'_1 相对于 ρ_n 也惟一时,称 P 惟一决定 Q.

定义 8. 设$\langle v\{P\}\rangle=\{v_1,v_2,\ldots,v_n\}$,$\langle v'\{Q\}\rangle=\{v'_1,v'_2,\ldots,v'_m\}$.当$\exists v_i\equiv v'_j(i\in[1,n],j\in[1,m])$时,称$\langle v\{P\}\rangle$和$\langle v'\{Q\}\rangle$交不空,记为$\langle v\{P\}\rangle\cap\langle v'\{Q\}\rangle\neq\varnothing$.

可以看到,如果节点集中只包含单个元素,交不空就退化为前面节点值相等的概念.一般情况下,交不空并不是严格意义上的相等,但是考虑到XML文档本身的半结构化特性,交不空比严格的相等更符合我们模型的要求.出于统一性考虑,我们也使用符号"≡",并定义$\langle v\{P\}\rangle\equiv\langle v'\{Q\}\rangle$成立当且仅当$\langle v\{P\}\rangle\cap\langle v'\{Q\}\rangle\neq\varnothing$.

3 函数依赖

3.1 函数依赖

定义 9. XML 文档上的函数依赖.

给定 $D=(E,A,M,N,r)$,D 上的函数依赖形为 $R(Q_1,Q_2,...,Q_n\rightarrow P_1,P_2,...,P_k)$.$R,R/Q_1,...,R/Q_n,R/P_1,...,R/P_k$ 都是 D 上的路径表达式,且 R 为根路径表达式.

若在符合 D 的 XML 树 X 上,有以下条件之一成立:

(1) $\langle R\rangle=\varnothing$,

(2) $\forall v_1,v_2\in\langle R\rangle$,如果$\langle v_1\{Q_i\}\rangle\equiv\langle v_2\{Q_i\}\rangle$对于 $i\in[1,...,n]$都成立,则有$\langle v_1\{P_j\}\rangle\equiv\langle v_2\{P_j\}\rangle$对于 $j\in[1,...,k]$都成立,就称 $R(Q_1,Q_2,...,Q_n\rightarrow P_1,P_2,...,P_k)$在 X 上成立,记为 $X\models R(Q_1,Q_2,...,Q_n\rightarrow P_1,P_2,...,P_k)$.称 R 为目标路径,$Q_1,...,Q_n$ 为头部路径,$P_1,...,P_k$ 为体部路径.

函数依赖的定义有两个要素,即成立的范围和相关的属性集.关系模式是平面的,因而所有的函数依赖都是"全局"的,每个属性也都是简单类型.XML 文档是一个带有层次结构的树模型,我们使用$\langle R\rangle$标识 X 上的若干棵子树(根 $v\in\langle R\rangle$),类比于关系数据库,它决定了函数依赖成立的范围.路径表达式$(Q_1,Q_2,...,Q_n)$和$(P_1,P_2,...,P_k)$则分别定义了属性集,它们都是相对于 R 的.

使用函数依赖,可以分别表示例 1 中的约束如下:

(1) school/course(cno→time,student_list)

(2) school/course/student_list/student(sno→address)

(3) school/course(time,student_list/student/sno→cno)

(4) school/course/student_list/student(address→address/street)

(5) school/course/student_list/student/address(street→district)

3.2 逻辑蕴涵

定义 10. 逻辑蕴涵.给定 $D=(E,A,M,N,r)$和一个 D 上的函数依赖集Σ,若任意使Σ成立的符合 D 的 XML 文档 X 必然使函数依赖 $R(Q_1,Q_2,...,Q_n\rightarrow P)$成立,则称Σ逻辑蕴涵 $R(Q_1,Q_2,...,Q_n\rightarrow P)$,记为$\Sigma\Rightarrow R(Q_1,Q_2,...,Q_n\rightarrow P)$.

定义 11. 被Σ逻辑蕴涵的函数依赖全体构成的集合,称为Σ的闭包,记为Σ^+,即$\Sigma^+=\{R(Q_1,Q_2,...,Q_n\rightarrow P)|\Sigma\Rightarrow R(Q_1,Q_2,...,Q_n\rightarrow P)\}$.

3.3 推理规则集

为了解决逻辑蕴含的判定问题,方法之一是从一组已知的函数依赖成立,推导出其他函数依赖成立,这就需要一个推理规则集.推理规则集首先必须是正确的,即推导出的函数依赖确实是成立的;其次推理规则集应该是完备的,即可以推导出所有成立的函数依赖.下面我们给出 XML 上函数依赖的推理规则集.

(1) 若在文档 X 上,$\langle R\rangle$为空,或只包含单个节点,则有 $R(Q_1,Q_2,...,Q_n\rightarrow P)$.

(2) 若$\exists i\in[1,n]$,使 Q_i 是 P 的前缀,则有 $R(Q_1,Q_2,...,Q_n\rightarrow P)$.

(3) 若有 $R(Q_1,Q_2,...,Q_n\rightarrow P_1,P_2,...,P_k)$,且 R/Q'是 D 上的路径表达式,则有 $R(Q_1,Q_2,...,Q_n,Q'\rightarrow P_1,P_2,...,P_k,Q')$.

特别地,若$\exists i\in[1,n]$,使 $Q_i=Q'$,则有 $R(Q_1,Q_2,...,Q_n\rightarrow P_1,P_2,...,P_k,Q')$.

(4) 若有 $R(Q_1,...,Q_n\rightarrow P_1,...,P_k)$,且有 $R(P_1,...,P_k\rightarrow Q'_1,...,Q'_m)$,则有 $R(Q_1,...,Q_n\rightarrow Q'_1,...,Q'_m)$.

(5) 若有 $R(Q_1,...,Q_n\rightarrow P_1,...,P_k)$,且 $R=R'/R_0$,则有 $R'(R_0/Q_1,...,R_0/Q_n\rightarrow R_0/P_1,...,R_0/P_k)$.

(6) 若有 $R'(R_0/Q_1,...,R_0/Q_n\rightarrow R_0/P_1,...,R_0/P_k)$,且 R'惟一决定 R_0.设 $R=R'/R_0$,则有 $R(Q_1,...,Q_n\rightarrow P_1,...,P_k)$.

定理 1. 推理规则(1)~规则(6)是正确的(对Σ使用推理规则所推出的函数依赖在Σ^+中).

证明:可以很容易地通过函数依赖的定义得到证明. □

推理规则(2)~规则(4)是自反性、增广性和传递性[1]在 XML 树上的自然扩展.推理规则(1)反映了函数依赖和成立范围的关系.如果在 XML 树上,所考查的范围 R 为空,或只包含单个节点,则以 R 为目标路径的函数依赖显然成立.推理规则(5)、规则(6)表示的是不同范围内函数依赖之间的关系,由于 XML 树的层次结构特征,一个作用域上的函数依赖可以推广到另一个作用域上.由于树的特点,每个节点都只有惟一的父节点,所以自下而上总是成立的,但是自上而下就必须要求目标路径存在一对一的关系(见推理规则(6)).

例 2:假设修改前面的 DTD 定义,让一个学生可以有多个住址.

此时若已知 school/course/student_list/student(address/street→address/district)成立,是不能推导出 school/course/student_list/student/address(street→district)成立的.因为当一个学生所具有的多个地址存在同街道但不同区的情况时,仍然可能使前面的函数依赖成立,但后面的函数依赖就不会成立.

作为推理规则的应用,可以很容易地得到下面类似于关系模式中的分解和合并定理.

定理 2(合并). 若对$\forall j\in[1,k]$,有 $X\models R(Q\rightarrow P_j)$,则有 $X\models R(Q\rightarrow P_1,P_2,...,P_k)$.

定理 3(分解). 若 $X\models R(Q\rightarrow P_1,P_2,...,P_k)$,则对$\forall j\in[1,k]$,$X\models R(Q\rightarrow P_j)$.

判定可满足性的含义是,给定 D 和 D 上的函数依赖集Σ,是否存在同时符合 D 和满足Σ的 XML 文档.在关系数据库中,相关问题是平凡的.在本文所讨论的函数依赖范畴内,可满足性是可以判定的.

算法 1. 给定 D 和 D 上的函数依赖集Σ,构造文档 X 如下:

(1) name(root)=r,将 root 放入待处理队列 L.

(2) 从 L 中取出节点,记为 v,进行以下操作,直至 L 为空.

(a) 若 M(name(v))=S,设 value(v)="1",否则设 value(v)=NULL

(b) 为属于 M(name(v))字母表的元素依次构造元素节点 $v'\in$subelem(v)

i. 若 M(name(v))=e',name(v')=e'

ii. 若 M(name(v))=$e|e'$,name(v')=e

iii. 若 M(name(v))=e,e',构造两个节点 v',v'',name(v')=e,name(v'')=e'

iv. 若 M(name(v))=e*,name(v')=e

v. 将构造的新元素节点放入队列 L

(c) 为每个 $a\in N$(name(v))构造属性节点 $v'\in$attr(v),name(v')=a,value(v')="1"

定理 4. 给定 D 和 D 上的一个函数依赖集Σ,存在同时符合 D 和满足Σ的文档 X.

证明:显然算法 1 生成的文档符合 DTD,且满足Σ. □

定义 12. 给定 $D=(E,A,M,N,r)$,$Q_1,Q_2,...,Q_n$ 相对于Σ,$R(R/Q_i$ 是 D 上的路径表达式,R 为根路径表达式)的闭包是一个路径表达式的集合,记为$(Q_1,Q_2,...,Q_n,\Sigma,R)^+$.$(Q_1,Q_2,...,Q_n,\Sigma,R)^+=\{P|R(Q_1,Q_2,...,Q_n\rightarrow P)$可以使用推理规则(1)~规则(6)推出$\}$.

定理 5. 推理规则(1)~规则(6)是完备的(所有Σ^+中的函数依赖都可以通过对Σ使用推理规则而推出).

证明:使用反证法.假设$\exists R(Q_1,Q_2,\ldots,Q_n\rightarrow P)\in\Sigma^+$,但是不能通过对$\Sigma$使用推理规则而推出.为了证明推理规则是完备的,需要构造出一棵符合 D 的 XML 树 X,使Σ在 X 上成立,同时使 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$在 X 上不成立,这样就与 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)\in\Sigma^+$矛盾.

(1) 构造 X

首先使用算法 1 的方法构造初始 X.设 $R=\rho_0/\rho_1/...\rho_{k-1}/\rho_k.../\rho_n(\rho_0=r)$,其中 ρ_k 相对于 ρ_{k-1} 不是惟一的,而 ρ_i 相对于 ρ_{i-1} 都是惟一的($i\in[k+1,n]$).显然,这样的 ρ_k 是存在的,否则在任意符合 D 的 X 上,$\langle R\rangle$都只包含单个节点,则根

据推理规则 1,有 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$,这与假设矛盾.

令 $R_0=\rho_0/\rho_1/\ldots\rho_{k-1}$,$R=R_0/\rho_k/R_1$.若在 R_0 上,ρ_j 相对于 ρ_{j-1} 都是惟一的($j\in[1,k-1]$),则如图 1 所示;否则如图 2 所示,其中 ρ_t 在 R_0 上下标最大且不惟一(ρ_{k-1} 相对于 ρ_t 是惟一的).在定理生成的初始 X 上可能没有虚线部分,则我们对其进行扩展,原则是和左侧完全对称,包括节点值.由前面的讨论易知,这样扩展后得到的 XML 树仍然是符合 DTD 的,且Σ也依然成立.在以节点 z'为根的树上(name=ρ_k 的右侧子树),对所有值不空的节点进行如下操作:首先将所有节点的 value 值置为“0”,然后对任意节点 u',若 $z'-u'\in(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$,则将 value($u'$)置为“1”.根据推理规则(2)和推理规则(4),可知以 u'为根的子树上所有非空节点值都会被置为“1”.若 u 和 u'是在左右子树上对称的节点,仅在此时有 $u\equiv u'$成立.

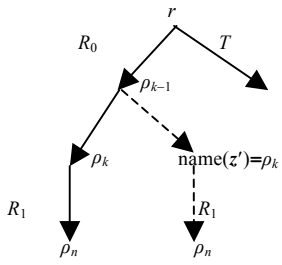

Fig.1 Tree constructed (Situation 1)

图 1 构造出的文档树(情况 1)

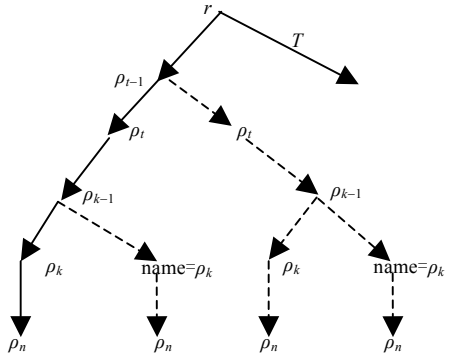

Fig.2 Tree constructed (Situation 2)

图 2 构造出的文档树(情况 2)

(2) 判定Σ在 X 上成立,取$\forall S(T_1,\ldots,T_k\rightarrow O)\in\Sigma$.我们对 S 和 R 的最长公共前缀 S_0 进行讨论:

(a) 当 S 和 R 没有公共前缀时,如图 1 中的 T,则显然对 X 的修正没有影响到 $S(T_1,\ldots,T_k\rightarrow O)$的成立;

(b) 当 $S_0\subseteq R_0$或当 $R_0=S_0$时,在图 1 中,对 X的修正不产生影响.而在图 2 中,若 $S\subseteq R_0$,设$\langle S\rangle=\{v_1,v_2\}$,则显然 $v_1\equiv v_2$成立,函数依赖也必然成立;

(c) 当 $S=R_0/\rho_k$时,若对$\forall i\in[1,k]$,都有 $T_i\in(R_1/Q_1,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$,则由定义有 $R_0/\rho_k(R_1/Q_1,\ldots,R_1/Q_n\rightarrow T_i)$成立.由推理规则(4)可知,$R_0/\rho_k(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n\rightarrow O)$成立,则 $O\in(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$.根据 X的构造,此时 $S(T_1,\ldots,T_k\rightarrow O)$成立.

若$\exists i\in[1,k]$,$T_i\notin(R_1/Q_1,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$.设$\langle S\rangle=\{v_1,v_2\}$,此时$\langle v_1\{T_i\}\rangle\equiv\langle v_2\{T_i\}\rangle$不成立,由函数依赖定义,$S(T_1,\ldots,T_k\rightarrow O)$成立;

(d) 当 $R_0/\rho_k\subseteq S_0$时,设 $S=R_0/\rho_k/S'$.由已知 $S(T_1,\ldots,T_k\rightarrow O)$,根据推理规则(5),有 $R_0/\rho_k(S'/T_1,\ldots,S'/T_k\rightarrow S'/O)$.此时对 S'/T_i,$i\in[1,k]$进行类似于前面(c)中的讨论,易知在文档 X 上,有 $S(T_1,\ldots,T_k\rightarrow O)$成立.

由 $S(T_1,\ldots,T_k\rightarrow O)$的任意性可知,$\Sigma$在 X 上成立.

(3) 判定 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$在 X 上不成立

若从Σ可以推出 $R_0/\rho_k(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n\rightarrow R_1/P)$,则因为已知 ρ_i 相对于 ρ_{i-1} 都是惟一的($i\in[k+1,n]$),由推理规则(6),必有 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$,与假设矛盾.所以可知 $R_1/P\notin(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$.而对$\forall i\in[1,n]$,$R_1/Q_i\in(R_1/Q_1,R_1/Q_2,\ldots,R_1/Q_n,\Sigma,R_0/\rho_k)^+$,所以 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$在 X 上不成立.

(4) 导出矛盾

构造的 X,使Σ成立,同时使 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$不成立.这与 $R(Q_1,Q_2,\ldots,Q_n\rightarrow P)\in\Sigma^+$矛盾,所以假设不成立.

综合可得,推理规则(1)~规则(6)是完备的. □

4 讨 论

为了判定是否有$\Sigma\Rightarrow R(Q_1,Q_2,\ldots,Q_n\rightarrow P)$成立,依据$(Q_1,Q_2,\ldots,Q_n,\Sigma,R)^+$的定义,只需要判定是否 $P\in(Q_1,Q_2,\ldots,Q_n,\Sigma,R)^+$.下面给出一个路径闭包的算法.

算法 2. 求 $Q_1,Q_2,...,Q_n$ 相对于Σ,R 的闭包$(Q_1,Q_2,...,Q_n,\Sigma,R)^+$.

输入:$\Sigma,Q_1,Q_2,...,Q_n,R,D(R/Q_i$ 是 D 上的路径表达式,R 为根路径表达式).

输出:$(Q_1,Q_2,...,Q_n,\Sigma,R)^+$.

方法:

(1) 处理Σ中的每个函数依赖,得到Σ',初始置Σ'为空

a. 对 $S(T_1,...,T_m\rightarrow O)$,若 $S=R/R_0$,将 $R(R_0/T_1,...,R_0/T_m\rightarrow R_0/O)$放入$\Sigma'$

b. 对 $S(R_0/T_1,...,R_0/T_m\rightarrow R_0/O)$,若 $R=S/R_0$,且 S 惟一决定 R_0,将 $R(T_1,...,T_m\rightarrow O)$放入$\Sigma'$

c. 对 $R(T_1,...,T_m\rightarrow O)$,将 $R(T_1,...,T_m\rightarrow O)$放入$\Sigma'$

(2) 根据下列规则构造序列 $X(0),X(1),...$

a. $X(0)=\{Q_1,Q_2,...,Q_n\}$

b. $X(i+1)= X(i)\cup\{P|R(P_1,P_2,...,P_k\rightarrow Z)\in\Sigma'\wedge P\subseteq Z\wedge P_j\in X(i)(j\in[1,\ldots,k])\}$

直到某个 i,有 $X(i+1)=X(i)$,算法中止.$X(i)$就是$(Q_1,Q_2,...,Q_n,\Sigma,R)^+$.

5 小 结

函数依赖语意的引入对于 XML 文档是非常重要的.我们下一步的工作包括扩展已有的 XML 模式定义方法,以增加对函数依赖的支持,并利用函数依赖的信息进行查询优化和数据集成的研究.

References:

[1] Abiteboul S, Hull R, Vianu V. Foundations of Databases. Boston, MA: Addison-Wesley, 1995.

[2] Ramakrishnan R, Gehrke J. Database Management Systems. NY: McGraw-Hill Higher Education, 2000.

[3] Hara CS, Davidson SB. Reasoning about nested functional dependencies. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Philadelphia: ACM Pess, 1999. 91~100.

[4] Buneman P, Fan WF, Weinstein S. Path constraints on semistructured and structured data. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Seattle: ACM Press, 1998. 129~138.

[5] Abiteboul S, Vianu V. Regular path queries with constraints. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Tucson: ACM Press, 1997. 122~133.

[6] Buneman P, Fan WF, Weinstein S. Path constraints in semistructured databases. Journal of Computer and System Sciences, 2000, 61(2):146~193.

[7] Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation, 2000. http://www.w3.org/TR/REC-xml.

[8] XML-Data. W3C Note, 1998. http://www.w3.org/TR/1998/NOTE-XML-data/.

[9] XML schema part 1: Structures. W3C Recommendation, 2001. http://www.w3.org/TR/xmlschema-1/.

[10] XML path language (XPath). W3C Recommendation, 1999. http://www.w3.org/TR/xpath.

[11] Fan WF, Libkin L. On XML integrity constraints in the presence of dtds. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Santa Barbara: ACM Press, 2001. 114~125.

[12] Arenas M, Fan WF, Libkin L. On verifying consistency of XML specifications. In: Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Madison: ACM Press, 2002. 259~270.

[13] Buneman P, Davidson SB, Fan WF, Hara CS, Tan WC. Keys for XML. In: Proceedings of the 10th International World Wide Web Conference. Hong Kong: ACM Press, 2001. 201~210.

[14] Buneman P, Davidson SB, Fan WF, Hara CS, Tan WC. Reasoning about keys for XML. In: Ghelli G, Grahne G, eds. Database Programming Languages, the 8th International Workshop. Lecture Notes in Computer Science 2397, Frascati: Springer-Verlag, 2001. 133~148.

[15] Arenas M, Libkin L. A normal form for XML documents. In: Lucian P, ed. Proceedings of the ACM Symposium on Principles of Database Systems (PODS). Madison: ACM Press, 2002. 85~96.