

# 存在 XML 强多值依赖的 XML Schema 规范化研究

殷丽凤<sup>1</sup> 郝忠孝<sup>1,2</sup>

(哈尔滨理工大学计算机科学与技术学院 哈尔滨 150080)<sup>1</sup>

(哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)<sup>2</sup>

**摘 要** 为了解决不完全信息环境下 XML 模式设计中 XML 文档的数据冗余和操作异常,研究了不完全信息环境下存在 XML 强多值依赖的 XML Schema 规范化问题;提出了 XML Schema、符合 XML Schema 的不完全 XML 文档树等概念;基于子树信息等价和子树信息相容的概念给出了 XML 强多值依赖的定义;提出了弱键路径和 XML 强多值依赖弱范式的定义;通过实例分析了在 XML Schema 中 XML 强多值依赖引起数据冗余的原因,提出了转换规则,给出了规范化算法。研究成果可较好地处理 XML 文档中出现大量不完全信息时的数据冗余问题,实现不完全信息环境下更好的 XML Schema 设计。

**关键词** 不完全信息,子树信息等价,子树信息相容,XML 强多值依赖,XML 强多值依赖弱范式

**中图法分类号** TP311.13 **文献标识码** A

## Research on Normalization of XML Schema with XML Strong MVD

YIN Li-feng<sup>1</sup> HAO Zhong-xiao<sup>1,2</sup>

(College of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China)<sup>1</sup>

(College of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)<sup>2</sup>

**Abstract** For solving data redundancy and abnormal manipulation for XML documents in schema design under incomplete information circumstances, the normalization of XML Schema existing XML strong multivalued dependencies under incomplete information circumstances was researched. The concepts of XML Schema and incomplete XML document tree according with XML Schema were formalized. Based on the equivalence of the sub-trees' information and the consistent of the sub-trees' information, XML strong multivalued dependencies' definition was given. Weak keys and XML strong multivalued dependencies weak normal form were given. The reasons of data redundancies aroused by XML Strong multivalued dependencies in XML Schema via the instance were analyzed, the transforming rules and the arithmetic of normalization were presented. The production in this work can deal with data redundancies in XML documents coming forth a great deal incomplete information and achieve better XML Schema design under incomplete information circumstances.

**Keywords** Incomplete information, Equivalence of the sub-trees' information, Consistent of the sub-trees' information, XML strong multivalued dependencies, XML strong multivalued dependencies weak normal form

## 1 引言

XML 规范化理论是 XML 数据库理论中的重要问题之一。M W Vincent 和 Liu Jixue 等人<sup>[1-3]</sup>基于 XML 文档提出了 XML 多值依赖定义,研究了相应的范式理论,并且在不完全信息环境下提出了 XML 强函数依赖和相应的范式理论,但没有说明不完全信息的具体语义。目前,DTD 和 XML Schema 是 XML 主流模式,有许多作者已经对基于 DTD 的 XML 规范化进行了研究。文献[4-6]根据数据约束 XML 函数依赖提出了基于 DTD 的范式和相应的规范化算法。文献[7]根据数据约束 XML 多值依赖提出了 DTD 的规范化算法;这些理论研究为基于 DTD 的 XML 规范化理论奠定了基

础。但 DTD 存在一些缺陷,如不支持数据类型、较差的扩展性等。W3C 针对 DTD 的缺陷设计了 XML 另一种模式 XML Schema,2001 年 5 月 W3C 正式推荐它为 XML 模式的标准。

XML Schema 符合 XML 规范,可以直接用 XML 的 API (如 DOM, SAX) 进行解析,不但支持命名空间、丰富的数据类型和属性组等,而且支持不完全信息。由于客观世界存在大量不完全信息,XML Schema 能够更好地描述现实客观世界,因此基于 XML Schema 的规范化理论更具有重要的现实应用意义。但 XML Schema 只可以表示不完全信息,却没有说明不完全信息的具体语义。为了说明不完全信息的语义,本文将文献[8]中的存在型不完全信息的语义(即存在型不完全信息必然对应着某个或几个实值)引入 XML Schema 中。由

到稿日期:2009-03-05 返修日期:2009-05-05 本文受黑龙江省自然科学基金(F200702)资助。

殷丽凤 女,博士研究生,主要研究领域为不完全信息环境下 XML 数据库理论和主动 XML 数据库理论,E-mail:yinlife0598@sina.com;郝忠孝男,教授,博士生导师,主要研究领域为数据库系统与理论。

于存在型不完全信息的引入,XML文档中的函数依赖、键、多值依赖的定义会失去原来的意义,因此不能用完全信息环境下的XML规范化理论解决不完全信息环境下的相应问题。对于不完全信息环境下存在XML多值依赖的XML Schema规范化研究,目前还没有查到相关文献。本文提出了相关理论,解决了相关问题。

作者对不完全信息环境下的XML数据库规范化理论做了一些工作,对XML强函数依赖<sup>[9]</sup>和XML强闭包依赖<sup>[10]</sup>进行了研究。基于存在型不完全信息的语义给出了XML Schema、符合XML Schema的不完全XML文档树、子树信息等价、子树信息相容等定义,提出了XML强多值依赖的概念。通过实例分析了在XML Schema中XML强多值依赖引起数据冗余的原因,提出了转换规则,给出了规范化算法,并且对算法的时间复杂度进行了分析,实现了更好的XML Schema设计。

## 2 基本概念

本文中, $\varphi$ 为存在型不完全信息,简记为不完全信息。由于XML Schema的复杂类型元素是对XML Schema的结构进行约束,简单类型元素和属性才能体现出数据信息,因此本文限制不完全信息体现在简单类型元素和属性中。为了便于研究问题,对复杂类型元素的子元素限定为不包含混合类型,下面给出基本概念。

**定义1(XML Schema)** XML Schema为一个九元组 $S=(CE, SE, A, D, CP, SP, R, Type, r)$ 。其中(1) $CE$ 表示复杂类型元素的有限集合;(2) $SE$ 为简单类型元素的有限集合;(3) $A$ 为属性的有限集合;(4) $D$ 为数据的标识;(5) $CP$ 表示从复杂类型元素到子元素的映射, $\forall \tau \in CE, CP(\tau)$ 表示正则表达式 $\alpha = \epsilon \mid \tau' \mid \alpha, \alpha \mid \alpha^*$ ,其中 $\epsilon$ 表示空的序列, $\tau' \in CE \cup SE$ ,“ $\mid$ ”表示连接,“ $*$ ”表示克林闭包, $\alpha \subseteq CE \cup SE$ ;(6) $SP$ 表示从简单类型元素到 $D$ 的映射;(7) $R$ 表示从复杂类型元素到属性的映射, $\forall \tau \in CE, R(\tau)$ 表示正则表达式 $\alpha = \epsilon \mid @a \mid \beta, \beta$ ,其中 $\epsilon$ 表示空的序列, $@a \in A$ ,“ $\mid$ ”表示连接, $\beta \subseteq A$ ;(8) $Type$ 表示 $CE \cup SE \cup A \cup D$ 所对应的数据类型和数据约束;(9) $r$ 表示Schema根元素, $r \in CE, \forall \tau \in CE, r$ 不会出现在 $CP(\tau)$ 中。

**定义2(XML Schema 路径)** 给定一个XML Schema  $S$ , 字符串 $p = w_1/w_2/\dots/w_n$ ,若 $w_1 = r, CP(w_1) \in CE$ ,其中 $i \in [1, n-2], w_n \in R(w_{n-1})$ ,称 $p$ 为 $S$ 中的属性类型路径;若 $w_1 = r, CP(w_1) \in CE \cup SE$ ,其中 $i \in [1, n-2], w_n = D$ ,称 $p$ 为 $S$ 中的值类型路径;若 $w_1 = r, CP(w_1) \in CE$ ,其中 $i \in [1, n-2], w_n \in CE$ ,称 $p$ 为 $S$ 中的复杂元素类型路径;若 $w_1 = r, CP(w_1) \in CE$ ,其中 $i \in [1, n-2], w_n \in SE$ ,称 $p$ 为 $S$ 中的简单元素类型路径。 $Dpaths(S)$ 表示 $S$ 上值类型路径集合; $Apaths(S)$ 表示 $S$ 上属性类型路径集合;由 $Dpaths(S)$ 和 $Apaths(S)$ 所构成的路径集合称为全路径集合,记作 $DAPaths(S)$ ; $Paths(S)$ 表示 $S$ 上所有类型路径的集合。 $parent(w_i)$ 表示路径 $w_1/w_2/\dots/w_i$ 的父路径 $w_1/w_2/\dots/w_{i-1}$ 的最后标识,即 $parent(w_i) = w_{i-1}$ 。

**定义3(符合XML Schema的不完全XML文档树)** 给定XML Schema为 $S=(CE, SE, A, D, CP, SP, R, Type, r)$ ,称七元组 $T=(V, lab, cele, sele, att, val, v_r)$ 为符合 $S$ 的不完全XML文档树。其中,(1) $V$ 表示节点的集合;(2) $lab$ 表示从 $V$ 到 $CE \cup SE \cup A \cup D$ 的映射,若 $lab(v) \in CE$ 称 $v$ 为复杂元素类型节点,若 $lab(v) \in SE$ 称 $v$ 为简单元素类型节点,若 $lab(v) \in A$ 称 $v$ 为属性类型节点,若 $lab(v) = D$ 称 $v$ 为值类型节点;(3) $cele$ 表示从复杂元素类型节点到复杂元素类型节点或简单元素类型节点的映射,若 $v \in CE, cele(v) = [v_1, \dots, v_n]$ ,满足 $lab(v_i) \in CP(lab(v))$ ,其中 $i \in [1, n]$ ;(4) $sele$ 表示从简单类型元素节点到值类型节点的映射,若 $v \in SE, sele(v) = v'$ ,满足 $lab(v') = D$ ;(5) $att$ 表示从复杂元素类型节点到属性类型节点的映射,若 $v \in CE, att(v) = [v_1, \dots, v_m]$ ,满足 $lab(v_i) \in R(lab(v))$ ,其中 $i \in [1, m]$ ;(6)若 $v$ 为值类型节点和属性类型节点, $val$ 返回相应类型的数据且满足相应的数据约束,若数据约束允许不完全信息,则相应的数据可能为 $\varphi$ ,即 $lab(v) \in A$ 或 $lab(v) = D, val(v)$ 返回 $v$ 的值且满足相应的 $Type(lab(v))$ ,若 $v \in CE \cup SE$ ,则 $val(v) = v$ ;(7) $v_r$ 为文档树的根节点且 $lab(v_r) = r$ 。

全XML文档树。其中,(1) $V$ 表示节点的集合;(2) $lab$ 表示从 $V$ 到 $CE \cup SE \cup A \cup D$ 的映射,若 $lab(v) \in CE$ 称 $v$ 为复杂元素类型节点,若 $lab(v) \in SE$ 称 $v$ 为简单元素类型节点,若 $lab(v) \in A$ 称 $v$ 为属性类型节点,若 $lab(v) = D$ 称 $v$ 为值类型节点;(3) $cele$ 表示从复杂元素类型节点到复杂元素类型节点或简单元素类型节点的映射,若 $v \in CE, cele(v) = [v_1, \dots, v_n]$ ,满足 $lab(v_i) \in CP(lab(v))$ ,其中 $i \in [1, n]$ ;(4) $sele$ 表示从简单类型元素节点到值类型节点的映射,若 $v \in SE, sele(v) = v'$ ,满足 $lab(v') = D$ ;(5) $att$ 表示从复杂元素类型节点到属性类型节点的映射,若 $v \in CE, att(v) = [v_1, \dots, v_m]$ ,满足 $lab(v_i) \in R(lab(v))$ ,其中 $i \in [1, m]$ ;(6)若 $v$ 为值类型节点和属性类型节点, $val$ 返回相应类型的数据且满足相应的数据约束,若数据约束允许不完全信息,则相应的数据可能为 $\varphi$ ,即 $lab(v) \in A$ 或 $lab(v) = D, val(v)$ 返回 $v$ 的值且满足相应的 $Type(lab(v))$ ,若 $v \in CE \cup SE$ ,则 $val(v) = v$ ;(7) $v_r$ 为文档树的根节点且 $lab(v_r) = r$ 。

**定义4(路径节点集)** 设 $T$ 为符合 $S$ 的不完全XML文档树, $p \in Paths(S), p = w_1/w_2/\dots/w_n, n = v_1, v_2, \dots, v_n$ 为 $T$ 中的节点序列。若 $lab(v_1) = w_1, \dots, lab(v_n) = w_n$ ,称 $n$ 为通过路径 $p$ 的路径节点集, $[[p]]$ 表示 $T$ 中通过路径 $p$ 的路径节点集的最后节点的集合。

**定义5( $T$ 的一对一拆分)** 设XML Schema为 $S, T$ 为符合 $S$ 的不完全XML文档树, $DAPaths(S)$ 为 $S$ 上的全路径集合, $T' \subseteq T$ ,若 $\forall p \in DAPaths(S)$ ,则 $\exists s \in T', s$ 为通过 $p$ 的路径节点集且 $s$ 唯一,称 $T'$ 与 $DAPaths(S)$ 为一对一关系,记作 $T' \leftrightarrow DAPaths(S)$ 。称每个与 $DAPaths(S)$ 为一对一关系的 $T'$ 为 $T$ 关于 $DAPaths(S)$ 的一个一对一拆分。本文限定符合 $S$ 的不完全XML文档树 $T$ 为关于 $DAPaths(S)$ 的所有一对一拆分所构成的子树的并集。若 $P \subseteq DAPaths(S)$ ,存在唯一的 $T'' \subseteq T'$ 且 $T''$ 与 $P$ 为一对一关系,把 $T''$ 记作 $T'|_P$ 。

**定义6(子树信息等价)** 设XML Schema为 $S, T$ 为符合 $S$ 的不完全XML文档树, $P = \{p_1, \dots, p_n\} \subseteq DAPaths(S), T_1 \subseteq T, T_2 \subseteq T, T_1 \leftrightarrow P, T_2 \leftrightarrow P, \forall v_{1i}, v_{2i} \in [[p_i]] (i \in [1, n], v_{1i} \in T_1, v_{2i} \in T_2)$ ,若满足下面的条件之一,则称子树信息等价,记作 $T_1 \doteq_m T_2$ ;否则称子树信息不等价,记作 $T_1 \not\doteq_m T_2$ 。

1)  $val(v_{1i})$ 和 $val(v_{2i})$ 均为完全信息,则 $val(v_{1i}) = val(v_{2i})$ ;

2)  $val(v_{1i})$ 和 $val(v_{2i})$ 都为不完全信息,它们的语义信息相同(对应的实值个数相等、取值范围相同)。进行不完全信息代换时, $val(v_{1i})$ 和 $val(v_{2i})$ 代换为同一完全信息。

**定义7(子树信息相容)** 设 $T, S, P, T_1, T_2, v_{1i}, v_{2i}$ 的意义同定义6。若 $v_{1i}, v_{2i}$ 满足下面的条件之一,则称子树信息相容,记作 $T_1 \doteq_n T_2$ ;否则称子树信息不相容,记作 $T_1 \not\doteq_n T_2$ 。

1)  $T_1 \doteq_n T_2$ ;

2) 若 $val(v_{1i})$ 为不完全信息, $val(v_{2i})$ 为完全信息,至少存在一个不完全信息代换过程,使 $val(v_{1i}) = val(v_{2i})$ 成立;

3) 若 $val(v_{1i})$ 和 $val(v_{2i})$ 都为不完全信息,且它们限定的代换范围的交集不是空的,则至少有一种不完全信息代换过程,使 $val(v_{1i}) = val(v_{2i})$ 成立。

**定义8(XML强多值依赖)** 给定XML Schema  $S$ ,在 $S$ 上的XML强多值依赖(记作XSMVD)具体表现形式为

$X \xrightarrow{S} Y|Z$ , 其中  $X \subseteq DAPaths(S), Y \subseteq DAPaths(S), Z = DAPaths(S) - (X \cup Y)$ 。符合  $S$  的不完全 XML 文档树为  $T$ , 在  $T$  中的任意两个子树  $T_1, T_2, T_1 \leftrightarrow DAPaths(S), T_2 \leftrightarrow DAPaths(S)$  满足  $T_1|_X \doteq_m T_2|_X$ , 则在  $T$  中必存在子树  $T_3 (T_3 \leftrightarrow DAPaths(S))$  且可以与  $T_1, T_2$  相同满足如下条件:

- 1)  $T_3|_X \doteq_m T_1|_X$  或  $T_3|_X \doteq_m T_2|_X$ ;
- 2)  $T_3|_{[Y-X]} \doteq_m T_1|_{[Y-X]}$  且  $T_3|_Z \doteq_m T_2|_Z$ 。

由于子树  $T_1, T_2$  的对称性, 一定存在另一个子树  $T_4 (T_4 \leftrightarrow DAPaths(S))$  且可以与  $T_1, T_2$  相同满足以下条件:

- 1)  $T_4|_X \doteq_m T_2|_X$  或  $T_4|_X \doteq_m T_1|_X$ ;
- 2)  $T_4|_{[Y-X]} \doteq_m T_2|_{[Y-X]}$  且  $T_4|_Z \doteq_m T_1|_Z$ 。

则称 XSMVD:  $X \xrightarrow{S} Y|Z$  在  $T$  上成立, 或者称  $T$  满足 XSMVD:  $X \xrightarrow{S} Y|Z$ 。

### 3 XSMVD 弱范式

本节首先给出不完全信息环境下弱键和 XSMVD 弱范式的定义, 通过实例分析不完全信息环境下设计不好的 XML Schema 产生数据冗余的原因, 给出消除数据冗余的转换规则和规范化算法。

**定义 9(弱键路径)** 设 XML Schema  $S=(CE, SE, A, D, CP, SP, R, Type, r), DAPaths(S)$  为  $S$  上的全路径集合,  $X \subseteq DAPaths(S), T$  为符合  $S$  的不完全 XML 文档树。若对于任意两个子树  $T_1, T_2 \in T$  且  $T_1 \leftrightarrow DAPaths(S), T_2 \leftrightarrow DAPaths(S)$  满足  $T_1|_X \neq_m T_2|_X, X$  为单个路径, 称  $X$  为弱键路径,  $X$  为多个路径的集合, 称  $X$  为弱键路径集。

**定义 10(XSMVD 弱范式)** 设  $S$  和  $T$  的意义同定义 9,  $\Sigma$  为  $S$  上的 XSMVD 集合,  $T$  满足  $\Sigma$ 。任意  $X \xrightarrow{S} Y \in \Sigma$ , 其中  $X, Y \subseteq DAPaths(S)$ , 若  $X$  都为弱键路径(集), 称此  $S$  满足 XSMVD 弱范式。

**例 1** 下面给出一个学校课程安排(Schedules)情况的 XML Schema  $S_1$ , 如图 1 所示。其中图边上的标记“1”和“\*”表示复杂元素对应的子元素个数以及属性的个数, “\*”表示一个或多个。

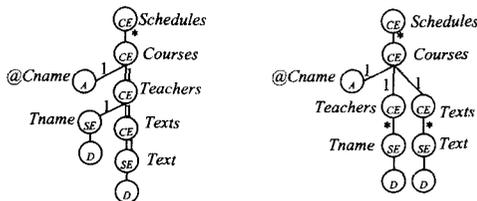


图 1 XML Schema  $S_1$       图 2 XML Schema  $S_2$

这一模式  $S_1$  表示存储的课程(Courses)由哪些老师(Teachers)讲授以及对对应哪些教材(Texts), 其中 Cname, Tname 和 Text 分别表示课程名、老师名和教材名。  $T_1$  为符合  $S_1$  的不完全 XML 文档树, 如图 3 所示。在  $T_1$  中具体安排哪门课程, 目前是不确定的, 但知道为{“数据库原理”, “数据库原理与设计”, “数据库原理与应用”}中的一门。具体由哪些老师讲授, 目前也是不确定的, 但知道可能由{“Johe”, “Mary”, “Smith”, “Joan”}中的两个老师讲授。不完全信息的语义为  $\varphi_1 \doteq \varphi_3 = \{“数据库原理”, “数据库原理与设计”\}, \varphi_2 \doteq \varphi_4 = \{“数据库原理”, “数据库原理与应用”\}。 \varphi_5 \doteq \varphi_6 = \{“Johe”, “Mary”\}, \varphi_7 \doteq \varphi_8 = \{“Smith”, “Joan”\}。由 XSMVD 的$

定义得 XML 强多值依赖  $Schedules, Courses, @Cname \xrightarrow{S} Schedules, Courses, Teachers, Tname, D | Schedules, Courses, Teachers, Texts, Text, D$  在  $T_1$  上成立。但在  $T_1$  中  $Schedules, Courses, @Cname$  不为弱键路径, 所以  $S_1$  不满足 XSMVD 弱范式, 在  $T_1$  中存在大量冗余数据。

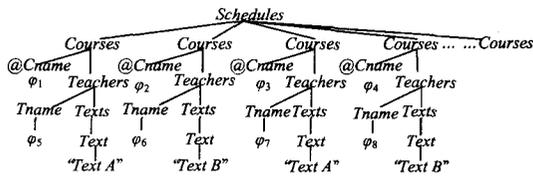


图 3 符合  $S_1$  的不完全 XML 文档树  $T_1$

可以看出, 在  $T_1$  中存在的冗余数据是由于  $S_1$  在设计上存在问题。在  $S_1$  中, 教材(Texts)与课程(Courses)有直接关系而与老师(Teachers)没有直接关系, 而  $S_1$  中把两个没有直接联系的实体嵌套在了一起, 即教材(Texts)嵌套在了老师(Teachers)的下面; 其次, 一门课程(Cname)可以由多名老师(Tname)讲授且对应多门教材(Text), 而在  $S_1$  中一门课程(Cname)只能由一名老师(Tname)讲授且对应一门教材(Text), 与原本所表达的意义不符。根据上述原因可以对  $S_1$  进行转换, 把与课程(Courses)有直接关系的教材(Texts)嵌套在课程(Courses)的下面, 更改  $S_1$  为一门可以由多名老师(Tname)讲授且对应多门教材(Text)的课程(Cname), 用“\*”表示这种一对多的关系。  $S_1$  修改后对应的 XML Schema  $S_2$  如图 2 所示。符合  $S_2$  的不完全 XML 文档树  $T_2$  如图 4 所示, 其中, 不完全信息的语义为  $\varphi_1 = \{“数据库原理”, “数据库原理与设计”\}, \varphi_2 = \{“数据库原理”, “数据库原理与应用”\}, \varphi_3 = \{“Johe”, “Mary”\}, \varphi_4 = \{“Smith”, “Joan”\}。在  $T_2$  中路径  $Schedules, Courses, @Cname$  为弱键路径,  $S_2$  满足 XSMVD 弱范式的, 确在  $T_2$  中减少了冗余数据。$

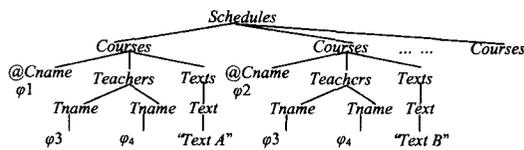


图 4 符合  $S_2$  的不完全 XML 文档树  $T_2$

若在  $T_1$  中课程的名字是确定的, 其它情况相同, 所对应的  $T_2$  会没有冗余数据。从而说明当 XML Schema 满足 XSMVD 弱范式, XML 文档出现大量不完全信息时, 可能仍存在数据冗余。在 XML 文档中不完全信息量比较少的情况下, 会消除数据冗余。

根据上面的实例分析, 若 XSMVD  $X \xrightarrow{S} Y|Z$  中的路径(集合)  $Z$  中存在复杂元素  $B$  嵌套在路径(集合)  $Y$  中复杂元素  $A$  的下面, 而  $B$  与  $A$  没有直接关系, 而与  $A$  的父亲元素有直接关系, 为了减少数据冗余, 则移动  $B$  所在的子树为 Parent ( $A$ ) 的子元素; 若实体之间的语义为一对多的关系, 则由原来的语义关系“1”修改为“\*” (表示一个或多个)。下面根据不同的情况, 给出相应的转换规则。

若 XSMVD:  $X \xrightarrow{S} Y|Z (Z = DAPaths(S) - (X \cup Y))$  中的  $Y$  和  $Z$  都为单个路径, 此转换情形如图 5 所示, 其中  $Tree_x$  表示由路径集合  $X$  所构成的子树。

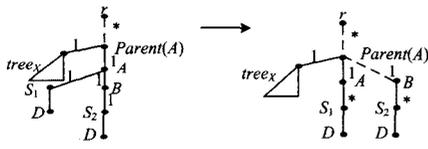


图5 转换情形 1

根据转换情形 1,下面给出其转换规则。

转换规则 1 移动属性或简单元素所在的子树。

设  $S$  转换为  $S'=(CE',SE',A',D',CP',SP',R',Type',r')$ ,其中

$CE' := CE; SE' := SE; A' := A; D' := D; r' := r; R' := R;$

$CP'(Parent(A)) := CP(Parent(A)) \cup B;$

$CP'(A) := (CP(A) - B)^*; CP'(B) := (CP(B))^*;$

$Type'$  与  $Type$  的意义相同;  $SP'$  与  $SP$  的意义相同;

$R'$  与  $R$  的意义相同;

$\Sigma' := \Sigma - tree_x \xrightarrow{S} r/\dots/Parent(A)/A/S_1/D \mid r/\dots/$

$Parent(A)/A/B/S_2/D \cup tree_x \xrightarrow{S} r/\dots/Parent(A)/A/S_1/D \mid r/\dots/Parent(A)/B/S_2/D.$

若 XSMVD:  $X \xrightarrow{S} Y \mid Z (Z = DAPaths(S) - (X \cup Y))$  中  $Y$  为路径集合的情况,而  $Z$  为单个路径的情况,需要在转换情形 1 的基础上增加新的节点,此转换情形如图 6 所示。

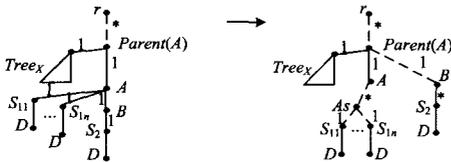


图6 转换情形 2

根据转换情形 2,下面给出其转换规则。

转换规则 2 移动子树并创建一个复杂元素节点。

设转换  $S$  为  $S'=(CE',SE',A',D',CP',SP',R',Type',r')$ ,其中

$CE' := CE \cup A_s; SE' := SE; A' := A; D' := D; r' := r;$

$CP'(parent(A)) := CP(parent(A)) \cup B;$

$CP'(A) := A_s^*; CP'(B) := (CP(B))^*;$

$CP'(A_s) := S_{11}, \dots, S_{1n};$

$Type'$  在  $Type$  的意义上加上  $A_s$  所对应的数据类型和相应的数据约束;

$SP'$  与  $SP$  意义相同;  $R'$  与  $R$  的意义相同;

$\Sigma' := \Sigma - (tree_x \xrightarrow{S} r/\dots/Parent(A)/A/S_{11}/D, \dots, r/\dots/Parent(A)/A/S_{1n}/D \mid r/\dots/Parent(A)/A/B/S_2/D) \cup (tree_x \xrightarrow{S} r/\dots/Parent(A)/A/A_s/S_{11}/D, \dots, r/\dots/Parent(A)/A/A_s/S_{1n}/D \mid r/\dots/Parent(A)/B/S_2/D).$

若 XSMVD:  $X \xrightarrow{S} Y \mid Z (Z = DAPaths(S) - (X \cup Y))$  中的  $Y$  为单个路径而  $Z$  为路径集合,根据 XSMVD 的补规则,转换情形与转换情形 2 相同。

若 XSMVD:  $X \xrightarrow{S} Y \mid Z (Z = DAPaths(S) - (X \cup Y))$  中的  $Y$  与  $Z$  都为路径集合,需要在转换情形 2 的基础上增加新的节点,此转换情形如图 7 所示。

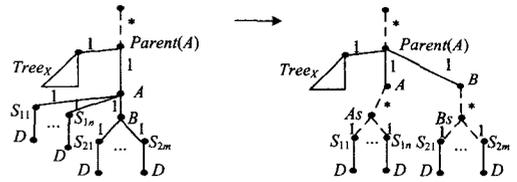


图7 转换情形 3

根据转换情形 3,下面给出其转换规则。

转换规则 3 移动子树并创建两个复杂元素节点。

设转换  $S$  为  $S'=(CE',SE',A',D',CP',SP',R',Type',r')$ ,其中

$CE' := CE \cup A_s \cup B_s; SE' := SE; A' := A; D' := D; r' := r;$

$CP'(parent(A)) := CP(parent(A)) \cup B;$

$CP'(A) := A_s^*; CP'(B) := B_s^*;$

$CP'(A_s) := S_{11}, \dots, S_{1n}; CP'(B_s) := S_{21}, \dots, S_{2m};$

$Type'$  在  $Type$  的意义上加上  $A_s$  和  $B_s$  所对应的数据类型和相应的数据约束;

$SP'$  与  $SP$  意义相同;  $R'$  与  $R$  的意义相同;

$\Sigma' := \Sigma - (tree_x \xrightarrow{S} r/\dots/Parent(A)/A/S_{11}/D, \dots, r/\dots/Parent(A)/A/S_{1n}/D \mid r/\dots/Parent(A)/A/B/S_{21}/D, \dots, r/\dots/Parent(A)/A/B/S_{2m}/D) \cup (tree_x \xrightarrow{S} r/\dots/Parent(A)/A/A_s/S_{11}/D, \dots, r/\dots/Parent(A)/A/A_s/S_{1n}/D \mid r/\dots/Parent(A)/B/B_s/S_{21}/D, \dots, r/\dots/Parent(A)/B/B_s/S_{2m}/D).$

定义 11(异常最大化 XSMVD) 设  $\Sigma$  为  $S$  上的 XSMVD 的集合,  $X \xrightarrow{S} Y \mid Z \in \Sigma$ , 若满足下面的条件,称  $X \xrightarrow{S} Y \mid Z$  为异常最大化 XSMVD。

1)  $X$  不为弱键路径(集);

2) 不存在与条件 1) 相同的 XSMVD 左部路径集相同。

下面给出规范化算法。

算法 1 Normalization-XWNF { 转换 XML Schema 为 XSMVD 弱范式的算法 }

输入: XML Schema  $S=(CE,SE,A,D,CP,SP,R,Type,r)$ ,  $S$  上的 XSMVD 集为  $\Sigma$ ;

输出: 满足 XSMVD 弱范式的 XML Schema  $S'=(CE',SE',A',D',CP',SP',R',Type',r')$ ;

Normalization-XWNF ( $S, \Sigma$ )

begin

(1)  $S' := S$ ; // 初始化

(2) 若  $S'$  是 XSMVD 弱范式,那么转向(4)。

(3) 对  $\Sigma$  中每个异常最大化 XSMVD:  $X \xrightarrow{S} Y \mid Z$ , 根据它的不同形式分别处理:

情况 1: 若  $Y$  和  $Z$  为单个路径且  $X$  不为弱键路径(集),那么利用转换规则 1 转换;

情况 2: 若  $Y$  为路径集合,  $Z$  为单个路径且  $X$  不为弱键路径(集) 或者若  $Z$  为路径集合,  $Y$  为单个路径且  $X$  不为弱键路径(集),那么利用转换规则 2 转换;

情况 3: 若  $Y$  与  $Z$  都为路径集合,且  $X$  不为弱键路径(集),那么利用转换规则 3 转换;

(4) 算法结束,输出  $S'$ 。

定理 1 算法 1 的时间复杂度为  $O(n)$ ,  $n$  为  $\Sigma$  中异常最大化 XSMVD 的个数。

证明:算法的时间复杂度是由  $\Sigma$  中异常最大化 XSMVD 的个数决定的,即为  $n$ ,所以整个算法的时间复杂度为  $O(n)$ 。

**结束语** 关于不完全信息环境下的 XML Schema 规范化理论,作者目前还没有查到国内的相应文献。本文针对此问题提出了存在 XML 强多值依赖的 XML Schema 规范化理论。基于 XML Schema、符合 XML Schema 的不完全 XML 文档树、子树信息等价和子树信息相容等概念提出了 XML 强多值依赖的定义。给出了弱键路径(集)和 XML 强多值依赖弱范式的定义,通过实例分析了 XML Schema 中数据冗余的原因,提出了转换规则,给出了规范化算法。本文的理论研究和实例分析表明,若 XML 文档存在大量不完全信息,一般情况下,此规范化理论只能减少数据冗余,不能完全消除数据冗余;若达到完全消除数据冗余,XML 文档不能出现大量不完全信息,从而表明数据冗余的多少与不完全信息的量存在着相互制约的关系。在未来的工作中,我们将对不完全信息环境下 XML 文档的查询进行研究。

### 参考文献

[1] Vincent M W, Liu Jixue. Multivalued dependencies and a 4NF for XML[C]//International Conference on Advance Information Systems Engineering, Klagenfurt, Austria, 2003

[2] Vincent M W, Liu Jixue, Liu Chengfei. A redundancy Free 4NF for XML[C]//The first International XML Database Symposium. Berlin, Germany, 2003

[3] Vincent M W, Liu Jixue, Liu Chengfei. Strong functional dependencies and their application to normal forms in XML[J]. ACM Transactions on Database System, 2004, 29(3): 445-462

[4] 吕腾,顾宁,施伯乐. XML DTD 的一种范式[J]. 计算机研究与发展, 2004, 41(4): 615-620

[5] 谈子敬,施伯乐. DTD 的规范化[J]. 计算机研究与发展, 2004, 41(4): 594-600

[6] Arenas M, Libkin L. A normal form for XML documents[C]//Proceedings of the 21th ACM SIGA-CT-SIG-MOD-SIGART Symposium on Principles of Database Systems Madison, Wisconsin, USA: ACM Press, 2002: 85-96

[7] 邱威,张立臣. 存在多值依赖的 XML DTD 规范化研究[J]. 计算机科学, 2007, 34(2): 149-152

[8] 郝忠孝. 空值环境下数据库导论[M]. 北京:机械工业出版社, 1996

[9] 殷丽凤,郝忠孝. XML 强函数依赖的推理规则[J]. 计算机科学, 2008, 35(9): 165-167

[10] 殷丽凤,郝忠孝. XML 强闭包依赖的研究[J]. 计算机科学, 2008, 35(11): 591-594

(上接第 191 页)

(见图 6),而在家,他仅可以查询河流的信息(见图 7)。

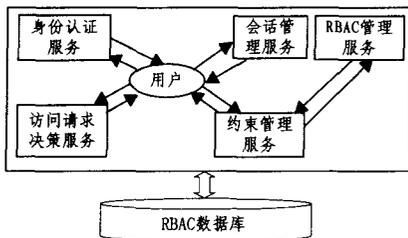


图 5 访问控制子系统的系统结构

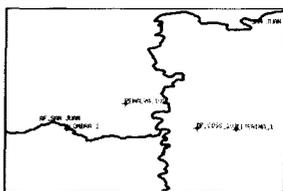


图 6



图 7

**结束语** SRBAC 是在 RBAC96 基础上作空间特性的扩展。引入空间特性后的 SRBAC 模型有着更全面、更具体的安全描述能力。SRBAC 对传统的约束、会话和系统状态进行空间扩充,解决了空间约束、会话状态转变和角色继承问题,但在系统一致性维护方面仍有很多工作需要进一步研究。

### 参考文献

[1] Sandhu R S, Conye E J, Feinstein H L. Role-Based Access Control Models[J]. IEEE Computer, 1996, 29(2): 38-47

[2] Osborn S L, Sandhu R, Munawer Q. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies[J]. ACM Trans. Information and System Security, 2000, 3(2): 85-106

[3] Kuhn D R. Role based access control on MLS systems without kernel changes[C]//Proc. of the third ACM Workshop on Role-Based Access Control. Fairfax, Virginia, United States, October 1998: 25-32

[4] Osborn S. Mandatory access control and role-based access control revisited[C]//Proc of the Second ACM Workshop on Role-Based Access Control. Fairfax, Virginia, United States, November 1997: 31-40

[5] Ferraiolo D F, Sandhu R, Gavrila S. Proposed NIST standard for role-based access control[J]. ACM Transactions on Information and System Security, 2001, 4(3): 224-274

[6] Joshi J, Bertino E, Latif U, et al. A Generalized Temporal Role-based Access Control Model[J]. IEEE Trans. Knowl. Data Eng., 2005, 17(1): 4-23

[7] Bertino E, Bonatti P A, Ferrari E. A Temporal Role-Based Access Control Model[J]. ACM Transactions On Information and System security, 2001, 4(3): 191-223

[8] Hansen F, Oleshchuk V. Spatial Role-Based Access Control Model for Wireless Networks[C]// Vehicular Technology Conference, VTC 2003-Fall, 2003 IEEE 58th: 2093-2097

[9] Sandhu R. Role activation hierarchies[C]//Proceedings of 2rd Acm Workshop on Role-based Access Control. Fairfax, Virginia, October 1998: 65-79