XML 的结构完整性约束推理

张剑妹[1),2)] 陶世群[2)] 梁吉业[2)] 曹 峰[2)]

[1)](长治学院计算机系 山西 长治 046011)

[2)](山西大学计算机与信息技术学院 太原 030006)

摘 要 为了有效地优化 XML 路径表达式查询,给出了一个 XML 结构完整性约束体系,这个体系全面描述了 XML 文档中节点或路径之间的结构关系,包括必需性包含、排他性包含、路径蕴涵、路径互斥和路径同现.在此基础上研究了 XML 结构完整性约束的逻辑蕴涵和一致性问题.文章首先采用约束重写技术将各种约束改写为路径蕴涵约束;然后给出了一组路径蕴涵的推理规则;最后以路径蕴涵闭包为工具证明了推理规则的完备性并给出了 XML 结构完整性约束的一致性判断方法.

关键词 XML 结构完整性约束;逻辑蕴涵;一致性;推理规则;路径蕴涵闭包

中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2010.02281

Reasoning about Structural Integrity Constraints for XML

ZHANG Jian-Mei[1),2)] TAO Shi-Qun[2)] LIANG Ji-Ye[2)] CAO Feng[2)]

[1)](*Department of Computer, Changzhi College, Changzhi, Shanxi* 046011)

[2)](*College of Computer and Information Technology, Shanxi University, Taiyuan* 030006)

Abstract In order to efficiently optimize XML path expression queries, this paper gives a system of structural integrity constraints for XML (XSICs). The pattern system provides effective means to specify the structural relationships between different paths or nodes in form of path implication, path concurrence and path mutual-exclusion, element obligatory inclusion, element exclusive inclusion. On the basis of this, the implication and consistency problems of XSICs are studied. Firstly, the technique of constraint rewriting is introduced to rewrite all kinds of constraints into path constraints. Secondly, a set of inference rules for path implication constraints is developed. Finally, the paper proves the completeness of these inference rules, and gives the consistency determinant method of XSICs by using the path implication closure.

Keywords structural integrity constraints for XML; logical implication; consistency; inference rule; path implication closure

1 引 言

随着 Web 上大量 XML 数据的出现,如何高效地管理和查询 XML 数据成为亟待解决的问题.为此,国内外许多学者致力于将传统的完整性约束应用到 XML 文档中的研究,比如键、外键、函数依赖、多值依赖[1-5]等.这类完整性约束规定了相关节点值之间的关系,在 XML 模式规范化设计方面起着重要的作用,但它们不能表示元素或路径之间的结构关系.

收稿日期:2007-01-20;最终修改稿收到日期:2010-10-20.本课题得到国家自然科学基金(70471003)和高等学校博士学科点专项科研基金(20050108004)资助.**张剑妹**,女,1970 年生,博士,副教授,研究方向为数据库技术和 XML 数据管理.E-mail: jmzhmm@yahoo.com.cn; jmzhang@sxu.edu.cn. **陶世群**,男,1946 年生,教授,博士生导师,主要研究领域为数据库理论与技术. **梁吉业**,男,1962 年生,教授,博士生导师,主要研究领域为粗糙集理论、数据挖掘、人工智能. **曹 峰**,男,1981 年生,硕士,主要研究方向为数据库技术.

作为XML文档的主要模式规范，DTD(Document Type Definition)[6]和XML Schema[7]虽然定义了XML文档中允许出现的元素类型、属性及其顺序和结构嵌套关系，但是不能处理元素或路径之间复杂的结构关系，例如：在一个文档中，从一个节点出发的路径要求(或排斥)另一个路径.如果有一种机制能说明XML元素或路径之间的这种结构关系，那么，在XML查询编译时，便可以有效地识别：(1)路径表达式中的冗余条件；(2)路径表达式中不相容的条件；(3)路径表达式中的冗余步；从而删除冗余条件，减少冗余步，以最小化路径表达式，改善XML查询的性能.为此，我们引入了XML结构完整性约束的概念，并研究了它的逻辑蕴涵和一致性问题.

目前，对XML结构完整性约束的研究还比较少，文献[8-9]等研究了半结构化数据中节点的边约束，这些研究可以说是最早的结构完整性约束的研究.文献[10]提出了使用DTD中隐含的结构完整性约束进行XPath路径表达式查询优化的方法.文献[11]研究了XML文档中路径蕴涵、路径互斥和路径同现三种结构约束及其逻辑蕴涵和一致性问题，但其定义是使用仅包含父-子关系的简单路径表达式给出的.文献[12]研究的XML树模式约束是使用包含“/”、“//”、“[]”和“*”的XPath路径表达式子集对文献[11]中的结构约束的扩展，而在实际应用中这种树模式约束又显得过于复杂.文献[13]研究了DTD中隐含的元素之间的排他性包含和必需性包含关系，但其目的是为了PAT代数表达式的等价转换和充分利用结构索引，因此，排他性包含的概念与本文的排他性包含有着本质的区别.

本文研究了包含路径蕴涵、路径互斥、路径同现、元素之间的排他性包含和必需性包含5种结构关系的XSICs的逻辑蕴涵和一致性问题.与文献[11-12]的结构约束不同，为了既能满足路径表达式查询最小化的需要又便于推理证明，XSICs中的路径蕴涵和路径互斥是使用包含“/”和“//”的线性路径表达式定义的.由于“//”的不确定性，线性路径表达式之间的包含关系要远远复杂于简单路径之间的包含关系，但又不同于文献[12]中的树模式之间的包含关系，而这种包含关系的确定恰恰是推理的关键所在.如何有效地描述线性路径表达式之间的包含关系是我们必须解决的一个重要问题.此外，我们引入了元素必需性包含和排他性包含两种结构包含关系.元素之间的结构包含关系与路径之间的结构关系以一种复杂的方式相互作用着，为了对XSICs进行有效的逻辑推理，必须将这两类结构关系有机地结合起来.如何将这两类结构关系结合起来是我们面临的又一个重要问题.毋庸置疑，约束类型和约束定义的不同将导致推理规则及其相关证明的巨大差异，从而直接影响着XSICs一致性问题的解决方法.

文章第2节介绍一些背景知识，并提出了子路径的概念；第3节给出XSICs中5种结构关系的语法和语义定义，并详细阐述对这几种结构关系进行约束重写的技术；第4节给出一组推理规则和路径蕴涵闭包的概念及其计算方法，并证明这组推理规则的正确性和完备性；第5节讨论XSIC的一致性问题；第6节总结全文并指出未来的研究方向.

2 背景知识

2.1 XML文档树

XML文档通常可以表示为节点标签的树，树中每个节点与文档的一个元素、属性或值(文本)相对应，边表示节点之间的嵌套关系.这里假定有3个两两不相交的集合：E是元素类型的有限集；A是属性名的有限集，单元素集$\{S\}$表示文本.XML文档树的形式化定义[2]如下.

定义1. 一个XML文档树T被定义为一个六元组$T=(V,lab,ele,att,val,r)$，其中：

V是节点的有限集；

lab是一个从V到$E\cup A\cup\{S\}$的映射.每个$v\in V$，如果$lab(v)\in E$，则v是一个元素；如果$lab(v)\in A$，则v是一个属性；如果$lab(v)=S$，则v是一个文本节点；

ele和att都是从V到V^*的部分映射.每个$v\in V$，如果$lab(v)\in E$，则$ele(v)$是V中的一个元素和文本节点序列，$att(v)$是V中的一个属性集；每个$v'\in ele(v)$或$v'\in att(v)$，v'叫作v的孩子节点且从v到v'有一个边；

val是一个从V到字符串的部分映射.每个$v\in V$，如果$lab(v)\in A$或$lab(v)=S$，则$val(v)$是v的一个字符串值；

r是文档树中唯一的一个根节点(即文档节点)且$r\in V$.

图1给出了一个关于论文出版数据的XML文档树.为了表示简单，图中忽略了文本节点和根节点.本文对属性和元素采用相同的处理方法，因此不严格区分属性和元素，即下文的节点类型泛指元素类型或属性名.

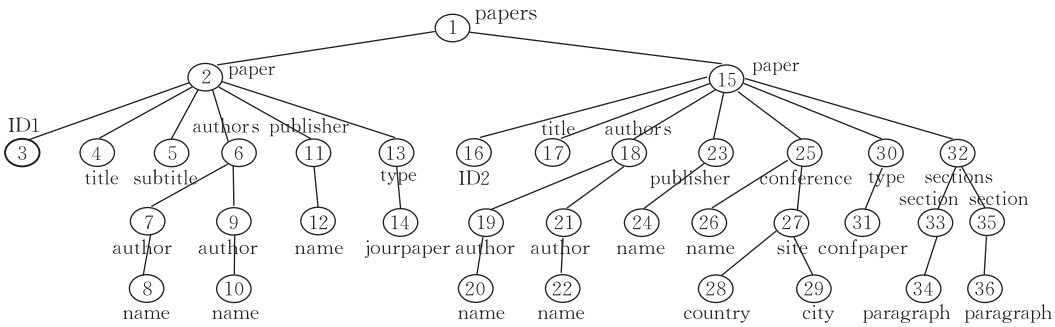

图 1　XML 文档树示例

2.2　路径表达式及相关符号

在 XML 文档树上,由节点序列构成的路径唯一确定一个节点,这样的路径叫作文档路径;路径表达式决定 XML 文档中的路径形式,它定义了一种导航 XML 文档的查询并返回指定路径所能访问到的节点集的方法.一个路径表达式对应于若干条文档路径,也就是说,从一个起始节点开始,路径表达式将返回一个节点集合.除非特别指明,下文中提到的路径均指路径表达式.本文的路径表达式是不包含通配符"*"的 XPath 表达式,其语法定义如下:

$P::=e|.|/P|//P|P/P|P//P|P[P]$, $e\in E\cup A$,

其中,"."表示当前上下文节点;"/"和"//"分别表示节点类型之间的直接包含关系(父-子关系)和间接包含关系(祖先-后代关系);"[]"中的路径表达式是谓词.以"/"开始的路径表达式是绝对路径,它从文档的根节点开始定位路径;其它路径表达式为相对路径.

定义 2.　线性路径表达式的形式为 $P=c_1e_1c_2e_2\cdots c_ne_n$,其中,$e_i\in E(i=1,2,\cdots,n-1)$,$e_n\in E\cup A$,$c_i\in\{/,//\}(i=1,2,\cdots,n)$;特别地,当 $c_i=$"/" $(i=1,2,\cdots,n)$时,P 叫作简单路径表达式.

定义 3.　设线性路径表达式 $P=c_1e_1c_2e_2\cdots c_ne_n$,$Q=c'_1e'_1c'_2e'_2\cdots c'_me'_m$,若 $m\leq n$,$c_i=c'_i$且 $e_i=e'_i$ $(i=1,2,\cdots,m)$,则 Q 是 P 的路径前缀;当 $m=n$ 时,称 P 等于 Q,记为 $P=Q$.

任何路径 P 都是其自身的路径前缀.线性路径表达式 P 中节点类型的个数是 P 的长度,记为 $|P|$.下文用 $P\in LP$ 表示 P 是一个线性路径表达式.

类似于文档树,路径表达式也可以表示为树形结构,这种树形结构被叫作树模式.图 2 给出了路径表达式/papers/paper[.//section][.//section//paragraph]/title 的树模式.路径表达式中的每个节点类型与树模式中的一个节点相对应,其中实心节点为表达式的返回节点.单线边(叫作孩子边)和双线边(叫作后代边)分别与表达式中的"/"和"//"相对应.若(u,v)是孩子边,则节点 v 被叫作节点 u 的孩子节点;若从 u 到 v 有一个边序列,则节点 v 被叫作节点 u 的后代节点.为了区分绝对路径和相对路径,在树模式中引入了一个根节点 #.和文档树一样,$lab(v)$表示树模式中节点 v 的标签.

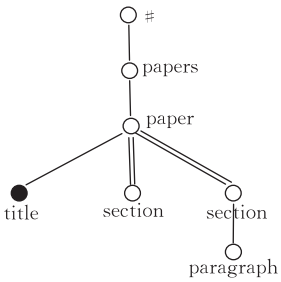

图 2　树模式示例

2.3　子路径

定义 4.　$\forall P,Q\in LP$,N_p 和 N_q 分别表示 P 和 Q 的树模式的节点集,Q 是 P 的子路径,当且仅当存在一个映射 $f:N_q\to N_p$ 满足如下条件:

(1)保持节点类型,即 $\forall v\in N_q$,若 $lab(v)=e$,则 $lab(f(v))=e$;

(2)保持边关系,即 $\forall u,v\in N_q$,若(u,v)是孩子边,则在 P 的树模式中,$f(v)$是 $f(u)$的孩子节点;若(u,v)是后代边,则在 P 的树模式中,$f(v)$是$f(u)$的后代节点.

根据子路径的定义,线性路径表达式"//paper//section"是"//paper/sections/section//paragraph"的子路径,但"/paper//section"不是"//paper/sections/section//paragraph"的子路径.

显然,任何路径表达式 P 和其路径前缀都是该路径表达式的子路径,本文使用 $Subpaths(P)$表示路径表达式 P 的子路径集.

3　XSICs 的定义

描述元素或路径之间结构关系的完整性约束叫作 XML 文档的结构完整性约束(简称 XSICs).

XSICs 描述元素或路径之间的蕴涵、互斥、同现以及元素之间的结构包含关系. 这些结构关系可分为两类:基于路径的 XSICs 和基于元素的 XSICs.

3.1 基于路径的 XSICs

基于路径的 XSICs 描述路径之间的蕴涵、互斥和同现关系(下文简称为路径约束). 任意给定一个 XML 文档树 T,V 表示 T 中的节点集,$u[P]$表示在 T 中从节点 u 开始沿着路径 P 可以访问到的节点集. 若用 ε 表示空路径,则 $u[\varepsilon]=\{u\}$,$r[\varepsilon]=\{r\}$. 路径蕴涵、路径互斥和路径同现定义如下.

定义 5. $\forall C,P,Q\in LP,u,v,v'\in V$,则

(1) 路径 P 蕴涵 Q,记作 $C(P\rightarrow Q)$;如果 $\forall u\forall v(u\in r[C]\wedge v\in u[P]\rightarrow\exists v'(v'\in u[Q]))$,则 T 满足 $C(P\rightarrow Q)$,记作 $T\models C(P\rightarrow Q)$;

(2) 路径 P 与 Q 互斥,记作 $C(P\nrightarrow Q)$;如果 $\forall u\forall v(u\in r[C]\wedge v\in u[P]\rightarrow\neg\exists v'(v'\in u[Q]))$ 且 $\forall u\forall v(u\in r[C]\wedge v\in u[Q]\rightarrow\neg\exists v'(v'\in u[P]))$,则 T 满足 $C(P\nrightarrow Q)$,记作 $T\models C(P\nrightarrow Q)$;

(3) 路径 P 与 Q 同现,记作 $C(P\leftrightarrow Q)$;如果 $C(P\rightarrow Q)\wedge C(Q\rightarrow P)$,则 T 满足 $C(P\leftrightarrow Q)$,记作 $T\models C(P\leftrightarrow Q)$.

其中:

① 路径 C 为约束的上下文路径,它指定路径约束关系在文档树的哪个子树上成立,且规定 C 为简单路径表达式. 当 $C=\varepsilon$ 时,表明特定的路径约束关系在整个文档树中成立,且上下文路径 ε 可以被缺省.

② P 和 Q 分别叫作约束的前件路径和后件路径;C 是 P 和 Q 的公共路径前缀. 当 $|P|=|Q|=|C|+1$ 时,路径蕴涵、互斥和同现表示兄弟元素之间的蕴涵、互斥和同现. 为了书写简单,在约束实例中通常将 P 和 Q 写成相对于 C 的相对路径.

例如:在图 1 所示的 XML 文档树中,如下路径约束成立:

φ_1 /papers/paper(subtitle→authors/author/name)

φ_2 /papers/paper(title↔.//author/name)

φ_3 /papers/paper(type/jourpaper ↛conference)

通俗地讲,路径蕴涵 $C(P\rightarrow Q)$表示在 C 所确定的 XML 文档子树上,如果路径 P 存在,那么路径 Q 也一定出现;如果两个路径相互蕴涵则称这两个路径同现;路径互斥 $C(P\nrightarrow Q)$则表示在 C 所确定的 XML 文档子树上路径 P 与路径 Q 不能同时出现.

3.2 基于元素的 XSICs

基于元素的 XSICs 描述了元素之间的蕴涵、互斥、同现和结构包含关系. 由于元素之间的蕴涵、互斥和同现关系是路径蕴涵、互斥和同现关系的特例,因此,这里只讨论元素的结构包含关系. 根据元素的结构包含关系在查询优化中的不同作用,将其分为排他性包含和必需性包含(下文简称为元素包含约束),定义如下.

定义 6. $\forall P,Q\in LP,e_i\in E,e_j\in E\cup A,u,v,v_1,v_2\in V$,则

(1) 元素 e_i 必需包含 e_j,记作 $e_i\Rightarrow e_j$;如果 $\forall u(lab(u)=e_i\rightarrow\exists v(lab(v)=e_j\wedge v\in u[P]))$,则 T 满足 $e_i\Rightarrow e_j$,记作 $T\models e_i\Rightarrow e_j$;

(2) 元素 e_i 排他包含 e_j,记作 $e_i\mapsto e_j$;如果 $\forall u(lab(u)=e_i\wedge\exists v_1\exists v_2(v_1\in u[P]\wedge v_2\in u[Q]\wedge lab(v_1)=lab(v_2)=e_j)\rightarrow P=Q)$,则 T 满足 $e_i\mapsto e_j$,记作 $T\models e_i\mapsto e_j$.

例如:在图 1 所示的 XML 文档中,下列约束成立:

φ_4 $paper\mapsto author$

φ_5 $author\Rightarrow name$

φ_6 $paper\Rightarrow author$

定理 1. $\forall e_i,e_j\in E,\forall e_k\in E\cup A$,如果 $e_i\Rightarrow e_j$ 且 $e_j\Rightarrow e_k$,则 $e_i\Rightarrow e_k$.

这个定理说明了必需性包含的传递性,其正确性是显然的.

3.3 XSICs 的约束重写

为了便于推理,本文将元素包含约束、路径同现和路径互斥约束表示为等价的路径蕴涵约束. 由于路径同现是双向的路径蕴涵,因此,下面着重讨论路径互斥约束和元素包含约束的约束重写问题.

3.3.1 路径互斥的约束重写

如前所述,路径互斥 $C(P\nrightarrow Q)$表示在 C 所确定的 XML 文档子树上,一个路径 P 存在必然排斥另一个路径 Q 的存在. 为了使用路径蕴涵表示这样的语义,引入了“逻辑非”运算符“¬”,即 $\neg P$ 表示 P 不存在. 这样 $C(P\nrightarrow Q)$就可以表示为 $C((P\rightarrow\neg Q)\wedge(Q\rightarrow\neg P))$.

引入“逻辑非”运算符后,线性路径表达式和线性路径表达式集之间的成员关系定义如下:给定任意线性路径表达式集 B 和一个线性路径表达式 Q,$Q\propto B$ 当且仅当 $P\in B$ 且 $Q\in Subpaths(P)$;$\neg Q\propto B$ 当且仅当 $\neg P\in B$ 且 $P\in Subpaths(Q)$.

3.3.2 必需性包含和排他性包含的约束重写

由于元素包含约束与路径约束之间的相互作用,如何将两类约束有机地结合起来是进行 XSICs

推理的一个必要条件.在关系数据库中,通常采用 chase 技术改写查询使之包含完整性约束的作用[14].本文采用同样的技术改写路径约束集使其集成元素包含约束的作用.下面的定理陈述了元素包含约束与路径蕴涵之间的关系.

定理 2.　令 $P=c_1e_1c_2e_2\cdots c_ke_k$,用 $Nodes(P)$表示 P 中的节点类型集,则有

(1) 若 $e_i\mapsto e_j$ 且 $e_i,e_j\in Nodes(P)(i<j,i=1,2,\cdots,k-1,j=2,\cdots,k)$,则 $c_1e_1\cdots c_ie_i//e_j\cdots c_ke_k\rightarrow P$;

(2) 若 $e_{k-1}\Rightarrow e_k$ 且 $e_{k-1},e_k\in Nodes(P)$,则 $c_1e_1c_2e_2\cdots c_{k-1}e_{k-1}\rightarrow P$;

(3) 若 $e_i\Rightarrow e_n$ 且 $e_i\in Nodes(P)\wedge e_n\notin Nodes(P)(i=1,2,\cdots,k)$,则 $P\rightarrow c_1e_1c_2e_2\cdots c_ie_i//e_n$.

证明.

(1) 从排他性包含的定义可以直接得出这个结论是正确的.

(2) 假设 $e_{k-1}\Rightarrow e_k$ 成立,而 $c_1e_1c_2e_2\cdots c_{k-1}e_{k-1}\rightarrow P$ 不成立,则在 XML 文档中路径 $c_1e_1c_2e_2\cdots c_{k-1}e_{k-1}$ 存在时 P 不一定存在,即在该文档中有无 e_k 作为后代节点的 e_{k-1} 节点存在,这与假设 $e_{k-1}\Rightarrow e_k$ 相矛盾,因此假设不成立.

(3) 从必需性包含的定义可以直接得出这个结论也是正确的.

从这个定理出发,可以直接得到一个使用元素包含约束 chase 路径约束集的方法.给定一个元素包含约束集 Ω 和一个路径约束集 Σ,首先计算约束集 Ω 的闭包 $Closure(\Omega)$,然后使用 $Closure(\Omega)$改写路径约束集 Σ 使之集成元素包含约束的作用.方法如下:

(1) 首先使用约束集 Ω 初始化 $Closure(\Omega)$,然后根据定理 1 检查 $Closure(\Omega)$中的每个约束:如果 $e_i\Rightarrow e_j$ 且 $e_j\Rightarrow e_k$,则在 $Closure(\Omega)$中添加约束 $e_i\Rightarrow e_k$,直到 $Closure(\Omega)$不再发生变化为止.

(2) 依次检查 Σ 中的每个路径表达式 $P=c_1e_1c_2e_2\cdots c_ke_k$,若 $e_i\mapsto e_j\in Closure(\Omega)$ 且 $e_i,e_j\in Nodes(P)(i<j,i=1,2,\cdots,k-1,j=2,3,\cdots,k)$,则将 $c_1e_1\cdots c_ie_i//e_j\cdots c_ke_k\rightarrow P$ 添加到 Σ 中;若 $e_{k-1}\Rightarrow e_k$ 且 $e_{k-1},e_k\in Nodes(P)$,则将 $c_1e_1c_2e_2\cdots c_{k-1}e_{k-1}\rightarrow P$ 添加到 Σ 中;若有 $e_i\Rightarrow e_n\in Closure(\Omega)$ 且 $e_i\in Nodes(P)\wedge e_n\notin Nodes(P)(i=1,2,\cdots,k)$,则将 $P\rightarrow c_1e_1c_2e_2\cdots c_ie_i//e_n$ 添加到 Σ 中.

(3) 重复(2)的操作,直到路径约束集 Σ 不发生变化为止.

例 1.　给定约束集 $\Sigma=\{\varphi_1,\varphi_2,\varphi_3\}$ 和 $\Omega=\{\varphi_4,\varphi_5,\varphi_6\}$,其中,$\varphi_1$、$\varphi_2$ 和 φ_3 为 3.1 节中的路径约束,φ_4、φ_5 和 φ_6 为 3.2 节中的元素包含约束,使用我们的 chase 技术得 $Closure(\Omega)=\{\varphi_4,\varphi_5,\varphi_6,paper\Rightarrow name\}$;chase 后的路径约束集 $\Sigma=\{\varphi_1,\varphi_2,\varphi_3,$ $/papers/paper//author\rightarrow/papers/papaer//author/name$, $/papers/paper\rightarrow/papers/paper//name$, $/papers/paper/authors/author\rightarrow/papers/paper/authors/author/name$, $/papers/paper\rightarrow/papers/paper//author$, $/papers/paper//author/name\rightarrow/papers/paper/authors/author/name\}$.

4　XSICs 的逻辑蕴涵

类似于关系数据库中的函数依赖,XSICs 之间也存在逻辑蕴涵问题,XSICs 逻辑蕴涵的定义及相关概念如下.

定义 7.　给定一个 XSICs 集 Σ 和一个 XSICs φ,如果每一个满足 Σ 的 XML 文档也满足 φ,则称 Σ 逻辑蕴涵 φ,记为 $\Sigma\models\varphi$;一个 XML 文档 T 满足一个 XSIC 集 Σ,记为 $T\models\Sigma$.

定义 8.　给定一个 XSICs 集 Σ,被 Σ 所蕴涵的 XSICs 集合叫作 Σ 的闭包,记为 Σ^+.

使用 3.3 节中的约束重写技术,可以将所有的约束改写为路径蕴涵约束.因此,XSICs 逻辑蕴涵问题被转化为路径蕴涵约束的逻辑蕴涵问题.解决逻辑蕴涵最有效的方法是给出一个有效的、完备的推理规则.不失一般性,下面给出了一个路径蕴涵约束的推理规则集,并证明这个推理规则集的有效性和完备性.

4.1　XSICs 的推理规则

规则 1.　$C(P\rightarrow Q),C'\in Subpaths(C)\Rightarrow C'(P\rightarrow Q)$.

规则 2.　$C(P\rightarrow\neg Q),C\in Subpaths(C')\Rightarrow C'(P\rightarrow\neg Q)$.

规则 3.　$C(P\rightarrow Q),C(Q\rightarrow S)\Rightarrow C(P\rightarrow S)$.

规则 4.　$C(P\rightarrow Q),C(Q\rightarrow\neg S)\Rightarrow C(P\rightarrow\neg S)$.

规则 5.　$C(P\rightarrow\neg Q),C(S\rightarrow Q)\Rightarrow C(P\rightarrow\neg S)$.

规则 6.　$\forall P,Q\in LP,Q\in Subpaths(P)\Rightarrow P\rightarrow Q$.

4.2　推理规则的有效性

推理规则的有效性是指每个能由 XSICs 集 Σ 使用推理规则推导出来的路径蕴涵约束一定被 Σ

所逻辑蕴涵,也就是说这些推理规则是正确的.

定理 3. 推理规则 1～规则 6 是正确的.

(1) 规则 1、2 的正确性

规则 1 的正确性是显然的.根据路径约束的定义,上下文路径为简单路径表达式.这种情况下,路径 C 确定的子树一定在其子路径 C' 所确定的子树范围内,既然在 C 所确定的子树上 P 出现时 Q 一定出现,那么在包含该子树的子树上 P 出现时 Q 也一定出现.

类似于规则 1,在 XML 文档树上,路径 C' 所确定的子树一定在其子路径 C 所确定的子树范围内,在 C 所确定的子树上 P 出现时 Q 不出现,那么在该子树上的任何子树上 P 出现时 Q 也不会出现.因此,规则 2 也是正确的.

(2) 规则 3、4 的正确性

规则 3、4 的正确性可以直接从路径蕴涵的语义得到.这里不再赘述.

(3) 规则 5 的正确性

证明. 假设 $C(P\rightarrow\neg Q)$ 和 $C(S\rightarrow Q)$ 成立,但 $C(P\rightarrow\neg S)$ 不成立,则至少在一个 C 确定的 XML 文档子树上(记作 T_s),P 和 S 同时存在;又因为 $C(S\rightarrow Q)$ 成立,因此,在这个子树 T_s 上,P 和 Q 同时存在.这与 $C(P\rightarrow\neg Q)$ 相矛盾,故假设不成立.

(4) 规则 6 的正确性

规则 6 陈述了这样的事实,在 XML 文档的任何子树上,一个路径存在,其子路径必然存在.其正确性也是显然的.

4.3 推理规则的完备性

4.3.1 路径蕴涵闭包

为了证明推理规则的完备性,引入了路径蕴涵闭包的概念.

定义 9. 给定一个 XSICs 集 Σ 和一个上下文路径 C,路径 P 关于 Σ 和 C 的路径蕴涵闭包是在 Σ 和 C 下路径 P 所蕴涵的全部路径的最小集,记为 $P^{+}_{(\Sigma,C)}$,即 $P^{+}_{(\Sigma,C)}=\{X\mid C(P\rightarrow X)$ 可使用推理规则从 Σ 推出,其中 $X\in\{Q,\neg Q\}\}$.

令 Δ 为路径约束集,Ω 为元素包含约束集,$\Sigma=\Delta\cup\Omega$,$Last(P)$ 表示 P 的最后一个节点类型.$P^{+}_{(\Sigma,C)}$ 的计算算法如下.

算法 1. *Closure-Comp*(Σ,C,P).

输入: non-empty $\Sigma,C,P\in LP$

输出: $P^{+}_{(\Sigma,C)}$

if *Constraint-Rewrite*$(\Delta)==\varnothing$ then return $\varnothing$;

$\Sigma=$*Constraint-Chase*$(\Delta,$*Closure*$(\Omega))$;

$i=0$; $A=\varnothing$; $B_i=\{P\}$; $B_{i+1}=\varnothing$;

if $(P\notin\Delta\&\&e_i\in Nodes(P)\&\&e_i\Rightarrow e_x\in\Sigma)$

if $e_i=Last(P)$ then $B_i=\{P//e_x\}$; else

$B_i=B_i\cup\{c_1e_1c_2e_2\cdots c_ie_i//e_x\}$;

for each $\varphi==C'(P'\rightarrow Q')\in\Sigma$

if $(C\in Subpaths(C')\parallel C'==\varepsilon)$ then $A=A\cup\{\varphi\}$;

/*规则 1*/

for each $\varphi==C'(P'\rightarrow\neg Q')\in\Sigma$

if $(C'\in Subpaths(C))$ then $A=A\cup\{\varphi\}$; /*规则 2*/

do while $B_i\neq B_{i-1}$

$i=i+1$; $B_i=B_{i-1}$;

for each $X\in B_i$

for each $\varphi==C(P'\rightarrow Y)\in A$

if $(X==S\&\&P'\in Subpaths(S))$ then

/*$X==S$ 表示 X 是形如 S 的线性路径表达式*/

if $(Y==Q'\&\&Q'\notin B_i)$ then

$B_i=B_i\cup\{Q'\}$; /*规则 6、3*/

if $(Y==\neg Q'\&\&\neg Q'\notin B_i)$ then

$B_i=B_i\cup\{\neg Q'\}$; /*规则 6、4*/

if $(X==\neg S\&\&Y==Q'\&\&S\in Subpaths(Q')$

$\&\&\neg P'\notin B_i)$ then /*$X==\neg S$ 表示 X 是形如 $\neg S$ 的线性路径表达式*/

$B_i=B_i\cup\{\neg P'\}$; /*规则 5*/;

/*对 B_i 进行最小化处理*/

for each $P\in B_i$

if $(Q\in B_i\&\&Q\in Subpaths(P))$ then $B_i=B_i-\{Q\}$

for each $\neg P\in B_i$

if $(\neg Q\in B_i\&\&P\in Subpaths(Q))$ then $B_i=B_i-\{Q\}$

return B_i.

算法 2. *Constraint-Rewrite*(Δ).

输入: 非空路径约束集 Δ

输出: 改写成路径蕴涵形式的约束集 Δ

for each $\varphi\in\Delta$

if $\varphi==C(P\leftrightarrow Q)$ then

replace φ with $C(P\rightarrow Q),C(Q\rightarrow P)$;

if $\varphi==C(P\not\leftrightarrow Q)$ then

if $|C|=1$ then return $\varnothing$

else replace φ with $C(P\rightarrow\neg Q),C(Q\rightarrow\neg P)$;

return Δ.

算法 *Constraint-Rewrite*(Δ) 将路径同现和路径互斥约束改写为路径蕴涵约束.算法 *Constraint-Chase*$(\Delta,$*Closure*$(\Omega))$ 使用元素包含约束 chase 的路径约束集 Δ,使其集成元素包含约束的作用.由于 chase 过程中不能处理 Δ 中没有路径 P 时元素包含约束对 Δ 的影响,因此,在对 B_i 进行初始化时考虑了这种情况.

路径 P 关于 Σ 和 C 的路径蕴涵闭包是在 Σ 和 C 下 P 所蕴涵的全部路径的最小集.也就是说,任何路径及其子路径(路径的否定及其子路径的否定)在 $P^{+}_{(\Sigma,C)}$ 仅出现一次.此外,一个路径 Q 及其否定形式 $\neg Q$ 也不可能同时出现在路径蕴涵闭包 $P^{+}_{(\Sigma,C)}$ 中,否则该约束集是不一致的.第 5 节将讨论结构完整约束集的一致性问题,这里不妨先假设给定的约束集是一致的.

4.3.2　完备性证明

引理 1.　$\varphi:C(P\rightarrow X)(X\in\{Q,\neg Q\})$ 可以使用 XSICs 推理规则从 Σ 中推出当且仅当 $X\propto P^{+}_{(\Sigma,C)}$.

这个引理的证明可以直接从路径蕴涵闭包的定义得出.

引理 2.　令 $X\in\{Q,\neg Q\}$,如果 P 不出现在文档树 T 中,则对 T 中的任意线性路径 C 和 Q,都有 $T\models C(P\rightarrow X)$ 成立;如果 C 不出现在文档 T 中,则对 T 中的任意线性路径 P 和 Q,都有 $T\models C(P\rightarrow X)$ 成立.

证明.　由于 $C(P\rightarrow Q)$(或 $C(P\rightarrow\neg Q)$)表示对任何满足 $C(P\rightarrow Q)$(或 $C(P\rightarrow\neg Q)$)的 XML 文档,在 C 所确定的文档子树上,如果路径 P 出现,则路径 Q 一定出现(或不出现);但如果路径 P 不出现,那么无论路径 Q 是否出现,也无论出现什么样的路径 Q,$C(P\rightarrow Q)$(或 $C(P\rightarrow\neg Q)$)都成立,这既符合逻辑也符合路径蕴涵的语义要求.同理,如果路径 C 不出现在文档中,则不论出现什么样的路径 P 和 Q,也不论 P 和 Q 以什么样的关系出现,$T\models C(P\rightarrow X)$ 都成立.

定理 4.　推理规则 1～规则 6 是完备的.

推理规则的完备性是指给定一个路径蕴涵约束集 Σ,对任何一个路径蕴涵约束 φ,如果 $\Sigma\models\varphi$,则 φ 一定能使用推理规则 1～规则 6 从 Σ 中推出.根据逆否命题的等价性,如果某个路径蕴涵 $\varphi:C(P\rightarrow X)(X\in\{Q,\neg Q\})$ 不能使用推理规则 1～规则 6 从 Σ 推出,则 φ 一定不为 Σ 所逻辑蕴涵.也就是说至少存在一个 XML 文档树 T,使得 $T\models\Sigma$,但 $T\not\models\varphi$.

证明.　假设 $\varphi:C(P\rightarrow Q)$(或 $\varphi:C(P\rightarrow\neg Q)$)不能使用推理规则 1～规则 6 从 Σ 推出,证明存在一个 XML 文档树实例 T,T 满足 Σ 但不满足 φ.

(1)采用下列方式构造一个 XML 文档树 T,使得 $P^{+}_{(\Sigma,C)}$ 中除 Q(或 $\neg Q$)以外的所有路径出现在 T 上.

①将 $P^{+}_{(\Sigma,C)}$ 分为两个子集 $P^{+}_{(\Sigma,C)1}$ 和 $P^{+}_{(\Sigma,C)2}$,其中 $P^{+}_{(\Sigma,C)1}$ 为所有无逻辑非符号的路径的集合,表示被 P 蕴涵的所有路径;后者是带逻辑非符号的路径组成的集合,表示在路径 C 所确定的子树上被 P 排斥的所有路径;

②从 $P^{+}_{(\Sigma,C)1}$(或 $P^{+}_{(\Sigma,C)2}$)中删除路径 Q(或 $\neg Q$),方法如下:

若 $\varphi:C(P\rightarrow Q)$,在 $P^{+}_{(\Sigma,C)1}$ 中查找路径 Q;如果无 Q,则在 $P^{+}_{(\Sigma,C)1}$ 查找满足 $Q\in Subpaths(Q')$ 的 Q';若在 $P^{+}_{(\Sigma,C)1}$ 中有 Q(或 Q'),则从 $P^{+}_{(\Sigma,C)1}$ 中删除(Q 或 Q');

若 $\varphi:C(P\rightarrow\neg Q)$,在 $P^{+}_{(\Sigma,C)2}$ 中查找路径 $\neg Q$;如果无 $\neg Q$,则在 $P^{+}_{(\Sigma,C)2}$ 查找满足 $Q'\in Subpaths(Q)$ 的 $\neg Q'$;若在 $P^{+}_{(\Sigma,C)2}$ 中有 $\neg Q$(或 $\neg Q'$),则从 $P^{+}_{(\Sigma,C)2}$ 中删除 $\neg Q$(或 $\neg Q'$),且在 $P^{+}_{(\Sigma,C)1}$ 中添加 Q(或 Q').如果在 $P^{+}_{(\Sigma,C)2}$ 中找不到 $\neg Q$ 和 $\neg Q'$,则直接在 $P^{+}_{(\Sigma,C)1}$ 中添加 Q.

③建立一个文档节点 D,将 C 的第一个节点类型作为文档的根元素节点连接到 D 上,依次为 C 中的其它元素类型各建立一个节点,并用单线连接.

④将 $P^{+}_{(\Sigma,C)1}$ 中的路径按其路径表达式规定的结构关系连接在文档的 C 子树上,构成一个子树 T_1;连接方法如下:

i. 对 $P^{+}_{(\Sigma,C)1}$ 中的每个路径 S,将其转换为以 C 为上下文路径的相对路径,为 S 的第一个节点类型建立一个节点并将其连接在 C 的最后一个节点类型所对应的节点上;

ii. 为 S 的其它节点类型各建立一个节点,无论结点类型之间是直接包含关系还是间接包含关系,都用单线连接.

⑤如果 $P^{+}_{(\Sigma,C)2}$ 非空,则产生一个 T_1 的副本 T_2,从 T_2 中删除路径 P(或 P' 且 $P\in Subpaths(P')$).将 $P^{+}_{(\Sigma,C)2}$ 中的路径连接到 T_2 的 C 子树上,连接方法同④;

⑥将 T_1 的文档节点和根元素节点与 T_2 的文档节点和根元素节点合并在一起,构成一个由两个子树 T_1 和 T_2 组成的无文本节点和属性值的文档树实例 T.当然我们可以给 T 添加任意文本节点,而不影响下文的证明.

例 2.　假定 Σ 为例 1 中改写后的约束集,构造一个满足 Σ 但不满足约束 $/papers/paper(type/jourpaper\rightarrow\neg conference)$ 的文档树.

$$P^{+}_{(\Sigma,C)}=\{/papers/paper/type/jourpaper,$$
$$\neg/papers/paper/conference,$$

$/papers/paper/authors/author/name$,
$/papers/paper/title\}$.

经过第②步处理后得 $P^{+}_{(\Sigma,C)1}=P^{+}_{(\Sigma,C)}$,$P^{+}_{(\Sigma,C)2}=\varnothing$. 构造的文档树 T 如图 3 所示.

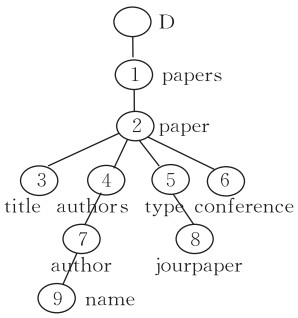

图 3 例 2 的文档树

注意:因为 $P^{+}_{(\Sigma,C)2}=\varnothing$,所以该文档树只有一个以 $r[C]$(即 *paper* 节点)为根的子树.

(2)证明该文档树 T 满足 Σ,也即对任何 $\psi\in\Sigma$,$T\models\psi$.

假设任一路径蕴涵 $\psi:C'(P'\rightarrow X')(X'\in\{Q',\neg Q'\})$,$\psi\in\Sigma$,但文档 $T\not\models\psi$,则

①若 C'不出现在文档 T 中,则根据引理 2 得 $T\models\psi$,与假设 $T\not\models\psi$ 相矛盾;

②若 P'不出现在文档 T 中,则根据引理 2 得 $T\models\psi$,与假设 $T\not\models\psi$ 相矛盾;

③若 C'和 P'都出现在文档 T 中,则有 $P'\propto P^{+}_{(\Sigma,C)}$. 根据引理 1 得 $C(P\rightarrow P')$成立;

若 $X'=Q'$,即 $C'(P'\rightarrow Q')\in\Sigma$. 由于上下文路径 C'出现在文档 T 中,根据 $P^{+}_{(\Sigma,C)}$ 的计算方法可知 $C'=C$ 或 $C\in Subpaths(C')$,由推理规则 1 得 $C(P'\rightarrow Q')$;再根据推理规则 3 得 $C(P\rightarrow Q')$,由引理 1 知 $Q'\propto P^{+}_{(\Sigma,C)}$. 因此 $T\models\psi$,与假设 $T\not\models\psi$ 相矛盾;

若 $X'=\neg Q'$,即 $C'(P'\rightarrow\neg Q')\in\Sigma$. 由于上下文路径 C'出现在文档 T 中,根据 $P^{+}_{(\Sigma,C)}$ 的计算方法可知 $C'=C$ 或 $C'\in Subpaths(C)$,由推理规则 2 得 $C(P'\rightarrow\neg Q')$;再根据推理规则 4 得 $C(P\rightarrow\neg Q')$,由引理 1 知 $\neg Q'\propto P^{+}_{(\Sigma,C)}$. 因此 $T\models\psi$,与假设 $T\not\models\psi$ 相矛盾;

由此可见,$T\models\Sigma$.

(3)证明文档树 T 不满足 $\varphi:C(P\rightarrow Q)$(或 $\varphi:C(P\rightarrow\neg Q)$),即 $T\not\models\varphi$.

设 $T\models\varphi$,若 $\varphi:C(P\rightarrow Q)$,则 Q 和 P 必出现在 T 的同一子树 T_1 上,即 $Q\propto P^{+}_{(\Sigma,C)1}$,也即 $Q\propto P^{+}_{(\Sigma,C)}$;若 $\varphi:C(P\rightarrow\neg Q)$,则 P 出现在 T 的子树 T_1 上且 Q 出现在子树 T_2 上,即 $\neg Q\propto P^{+}_{(\Sigma,C)2}$,也即 $\neg Q\propto P^{+}_{(\Sigma,C)}$;根据引理 1,$\varphi$ 能由 Σ 使用推理规则 1～规则 6 推出,与假设"φ 不能由 Σ 使用推理规则 1～规则 6 推出"矛盾,因此 $T\not\models\varphi$.

综上所述,路径蕴涵 $\varphi:C(P\rightarrow X)(X\in\{Q,\neg Q\})$不能由 Σ 根据推理规则推出,则 φ 一定不为 Σ 所逻辑蕴涵,所以推理规则 1～规则 6 是完备的.

5 XSICs 的一致性

和结构完整性约束相关的另一个判定性问题是它的一致性问题. 类似于文献[11],XSICs 的一致性定义如下.

定义 10. 给定一个线性路径表达式的有限集 B 和 B 上的一个 XSICs 集 Σ,是否存在一个文档树 T,使得 $\forall P\in B$,P 都出现在 T 中且 $T\models\Sigma$. 如果有这样的文档树 T 存在,则 XSICs 集 Σ 是一致的(或可满足的).

造成 XSICs 不可满足的原因是 XSICs 集中的约束本身的不一致性和约束之间的不一致性. 其可能出现的情况有 3 种:首先,一个路径互斥约束的上下文路径的长度不能等于 1,如果上下文路径长度为 1,则表明在 XML 文档树的根元素节点下,一条路径的存在排斥另一条路径,这显然是不可能的[11]. 这个问题在约束重写时(算法 2)已得到了解决. 其次,如果两个路径互斥,则其中的一个不能被另一个直接或间接地蕴涵. 最后,如果两个路径互斥,则这两个路径不能被同一个路径直接或间接地蕴涵. 本文利用路径蕴涵闭包解决 XSICs 的一致性问题.

定理 5. 若 P 为约束集 Σ 中的任意约束的前件路径,如果 $P^{+}_{(\Sigma,C)}=\varnothing$,则 Σ 是不可满足的.

该定理的正确性是显然的,$P^{+}_{(\Sigma,C)}=\varnothing$意味着约束重写后的约束集 Σ 为空;而造成这种结果的直接原因是路径互斥的上下文路径长度等于 1.

引理 3. 若 $C(X\rightarrow Y)$成立,且 $X\propto P^{+}_{(\Sigma,C)}$,则 $Y\propto P^{+}_{(\Sigma,C)}$.

证明. 如果 $X\propto P^{+}_{(\Sigma,C)}$,则有 $C(P\rightarrow X)$;又因为 $C(X\rightarrow Y)$成立,根据规则 3 得 $C(P\rightarrow Y)$,所以 $Y\propto P^{+}_{(\Sigma,C)}$.

推论 1. 若 $C(X\rightarrow\neg Y)$成立,且 $X\propto P^{+}_{(\Sigma,C)}$,则 $\neg Y\propto P^{+}_{(\Sigma,C)}$.

引理 4. 如果$C(P\rightarrow\neg Q)$且$Q\in Subpaths(Q')$,

则 $C(P\rightarrow\neg Q')$一定成立.

证明.　因为 $Q\in Subpaths(Q')$,根据推理规则 6 得 $Q'\rightarrow Q$ 成立;又因为 $C(P\rightarrow\neg Q)$,根据推理规则 5 得 $C(P\rightarrow\neg Q')$.

引理 3 及其推论说明了如果一个路径蕴涵的前件路径在某个路径蕴涵闭包,则其后件路径也一定在该路径蕴涵闭包,也说明了在一个路径蕴涵闭包中若同时存在 Q 和$\neg Q$,则表明这个路径既蕴涵 Q 也排斥 Q,因此 Σ 是不可满足的.根据引理 3 和 4 可以得出:如果在一个路径蕴涵闭包中存在 Q'和$\neg Q$,若 $Q\in Subpaths(Q')$,则该路径既蕴涵 Q'又排斥 Q',因此 Σ 也是不可满足的.定理 6 叙述了 XSICs 集内约束之间不一致性的判断方法.

定理 6.　令 $*^{+}_{(\Sigma,C)}$ 表示在上下文路径 C 下,Σ 中所有约束的前件路径的路径蕴涵闭包的集合,即 $*^{+}_{(\Sigma,C)}=\{P^{+}_{i(\Sigma,C)}\mid C(P_i\rightarrow X)\in\Sigma, X\in\{Q,\neg Q\}\}$,若某个 $P^{+}_{i(\Sigma,C)}(i=1,2,\cdots)$中存在 Q'和$\neg Q$ 使得 $Q\in Subpaths(Q')$,则 Σ 是不可满足的.

这个定理的正确性在引理 3 和 4 的分析中已经得到证明,这里不再赘述.需要强调的是在实际应用中,计算路径蕴含闭包时并不需要穷举 Σ 中的所有约束的上下文路径,而是根据 Σ 中的路径互斥约束决定计算哪些上下文路径下的路径蕴涵闭包.限于篇幅,不再给出 XSICs 一致性的判断算法.

6　结束语

XML 结构完整性约束表示了 XML 文档中元素或路径之间的结构关系,本文把 XML 结构完整性约束看成一个独立的模式体系研究了它的推理问题.文中定义了 5 种结构完整性约束的语法和语义,给出了一组有效的、完备的 XML 路径蕴涵约束的推理规则.借用关系数据库中函数依赖闭包的概念,本文提出了路径蕴涵闭包的概念及其计算方法.以此为工具,证明了推理规则的完备性,解决了 XML 结构完整性约束的一致性判断问题.在以后的工作中,我们将进一步研究 XML 结构完整性约束在路径表达式查询优化中的作用,探讨如何利用 XML 结构完整性约束对路径表达式进行最小化处理.此外,XML 结构完整性约束可以看成是对 DTD 的扩展和补充,今后我们的另一个主要研究课题是如何使用 XSICs 来扩展 DTD,得到所有路径表达式的路径蕴涵闭包,从而对路径表达式查询进行全面的优化.

参　考　文　献

[1] Buneman P, Davidson S, Fan W F, Hara C, Tan W. Keys for XML. Computer Networks, 2002, 39(5): 473-487

[2] Fan W F, Libkin L. On XML integrity constraints in the presence of DTDs. Journal of the ACM, 2002, 49(3): 368-406

[3] Tan Zi-Jing, Pang Yin-Ming, Shi Bo-Le. Reasoning about functional dependency for XML. Journal of Software, 2003, 14(9): 1564-1570(in Chinese)
(谈子敬, 庞引明, 施伯乐. XML 上的函数依赖推理. 软件学报, 2003, 14(9): 1564-1570)

[4] Zhang Zhong-Ping. The research on XML database schema normalization based on constraints [Ph. D. dissertation]. Fudan University, Shanghai, 2004(in Chinese)
(张忠平. 基于约束的 XML 数据库模式规范化研究[博士学位论文]. 复旦大学, 上海, 2004)

[5] Wu Yong-Hui. Hierarchical schemas design for XML schemas and DTDs normalization design. Journal of Software, 2004, 15(7): 1099-1106(in Chinese)
(吴永辉. 用于 XML 模式和 DTD 规范化设计的层次模式设计. 软件学报, 2004, 15(7): 1099-1106)

[6] Bray T, Paoli J, Sperberg-McQueen C M et al. Extensible markup language (XML) 1.0 (Second Edition.). W3C Recommendation, World Wide Web Consortium, Technical Report: RECxml-20001006, 2000

[7] Thompos H S, Beech D, Maloney M et al. XML schema Part 1: Structures (Second Edition). W3C Recommendation, World Wide Web Consortium, Technical Report: REC-xmlschema-1-20041028, 2004

[8] Calvanese D, Giacomo G D, Lenzerini M. What can knowledge representation do for semi-structured data? //Proceedings of the 15th National Conference on Artificial Intelligence (AAAI'98). Madison, Wisconsin, United States, 1998: 205-210

[9] Cali A, Calvanese D, Lenzerini M. Semistructured data schemas with expressive constraints//Proceedings of the 7th International Workshop on Knowledge Representation meets Databases(KRDB). Berlin, Germany, 2000: 3-16

[10] Kwong A, Gertz M. Schema-based optimization of XPath expressions. Department of Computer Science, University of California at Davis, Davis, California, United States, 2001

[11] Kwong A, Gertz M. Structural constraints for XML. Department of Computer Science, University of California at Davis, Davis, California, United States, 2002

[12] Kwong A, Gertz M. On tree pattern constraints for XML documents. Department of Computer Science, University of California at Davis, Davis, California, United States, 2003

[13] Che D, Aberer K, Ozsu M T. Query optimization in XML structured document databases. The International Journal on Very Large Data Bases (The VLDB Journal), 2006, 15(3): 263-289

[14] Maier D, Mendelzon A O, Sagiv Y. Testing implications of data dependencies. ACM Transactions on Database Systems (TODS), 1979, 4(4): 455-469

ZHANG Jian-Mei, born in 1970, Ph. D., associate professor. Her main research interests include database technology and XML data management.

TAO Shi-Qun, born in 1946, professor, Ph. D. supervisor. His main research interests focus on database theory and technology.

LIANG Ji-Ye, born in 1962, professor, Ph. D. supervisor. His main research interests include rough set theory, data mining and artificial intelligence.

CAO Feng, born in 1981, M. S.. His main research interests focus on database technology.

Background

This paper is supported by the National Natural Science Foundation of China under grant No. 70471003 and the Research Foundation for the Doctoral Program of Ministry of Education of China under grant No. 20050108004.

With the emergence of large number of XML data in the internet, more and more people tend to manage XML data and retrieve information from them in the same way they do with the database. In order to enhance semantic specification of XML, many scholars at home and abroad concentrate on extending pattern system of XML documents by the traditional integrity constraints such as keys, foreign keys and functional dependencies. These integrity constraints, which play an important role in XML schema normalization, prescribe the value relationships among related nodes. But they are inadequate to specify structural relationships between different nodes or paths. As main patterns of XML documents, Document Type Definitions (DTDs) and XML Schema specify admissible elements, elements nesting, attributes and their order, but they are also inadequate to deal with complex structural relationships between different nodes or paths. One of the common features shared by all XML languages is the use of path expressions, which define a way of navigating XML Queries in XML documents. Since XML documents are often modeled as node-labeled trees, a path expression can also be modeled as a tree, named a tree pattern. Answering a path expression query requires matching the tree pattern representing this query against an XML document (or document set). The efficiency of the matching operation largely depends on the size of tree patterns, so minimizing path expression is very essential in XML queries. Now, there are many achievements in the research on the path expression minimization. These achievements only consider required children, required descendant and subtypes constraints, without taking complex structural relationships between different nodes or paths into account. However, the structural relationships provide more opportunities for XML queries minimization. Therefore, the paper introduce a pattern system for XML documents—structural integrity constraints for XML (XSICs). As stand-alone formalisms, the paper shows the implication and consistency problems are efficiently decidable, and there exists a sound and complete set of inference rules for XSICs.