基于转换技术的XML文档规范化及更新

岳 昆[1,2] 吴益忠[1] 王晓玲[1] 周傲英[1]

(复旦大学计算机科学与工程系 上海200433)[1] (云南大学计算机科学与工程系 昆明650091)[2]

摘 要 XML已经成为互联网上数据表示、集成和转换的标准。作为一种半结构化数据,XML更新操作是扩展其查询能力的一个重要方面。为了保持原来的语义完整性,避免更新过程中的冗余和异常现象,作者提出了一种新的方法。该方法基于XML转换技术来规范XML文档,并基于规范化的XML,XNF来直接更新XML文档。该方法完成更新后,保持了XML的结构和原来的完整性约束。

关键词 更新,XNF,XML Key,PTD,XML转换

Normalizing and Updating XML Documents Based on XML Transformation

YUE Kun[1,2] WU Yi-Zhong[1] WANG Xiao-Ling[1] ZHOU Ao-Ying[1]

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

(Department of Computer Science and Engineering, Yunan Yuniversity, Kunming 650091)[2]

Abstract XML has become the standard of data representation, integration and transformation on the Web. As the semi-structured data, XML update operation is an important aspect of extending its query capability. In order to preserve the original semantic integrity, avoid the redundancy and the abnormity in the process of update, in this paper we present a novel method to normalize XML documents based on XML transformation technology, and update XML document directly based on the normalized XML, XNF. In our method, the integrity constraints and structure of XML are preserved after the update is performed.

Keywords Update, XNF (XML Normal Form), XML key, DTD, XML transformation

1 引言

近年来,XML已经成为了互联网上数据表示、集成和转换的标准,同样也成为Web上数据存储的一种方式,从Web数据存储的角度来看,XML作为一种数据库,允许在其上定义更新操作并采用相应的技术进行处理是扩展XML查询能力的一个重要方面。诸如XQuery、XPath等XML查询语言并不支持XML数据的更新,对于XML的变化检测方面,diff算法[5]在XML的新旧版本之间尽可能多地匹配节点。另一方面,很多XML的查询处理是基于关系数据库并采用关系数据库的成熟技术而进行。文[1]将XML作为关系数据的视图和数据的一种展现方式,在XML数据上定义更新操作的最终目标是更新存储XML的关系数据库,更新后的XML文档只有通过完全发布相关关系数据才能得到,该方法在XML文档较大时效率迅速降低,且该方法并未涉及到更新前后XML数据本身完整性约束的保持。文[2]将XML文档保持XML键约束[3]地存储到关系数据库中,同时把XML的关系存储作为XML的视图,利用关系数据库的成熟机制将XML键约束[3]的保持等价地转化为关系数据库中函数依赖的保持,因此XML文档的更新能够保持原来的完整性约束。这种方法的输入是需要更新的XML文档和用扩展的XQuery表达的XML更新操作,XML数据的关系存储和约束检测对用户来说完全透明,然而这种方法涉及到关系数据的操作和发生更新的XML数据的发布,因此开销也较大、效率不高。而对于非规范化XML文档上的更新操作,由于XML数据本身的冗余,上述两种更新方法都没有较好的处理方法。

XML作为Web上数据表示和存储的方式,如果其查询处理不依赖于其它数据存储的中间形式、避免多次的数据存储和发布,直接在XML上进行则可以提高XML数据的处理效率和性能,这对直接在XML文档上进行更新提出了新的要求。文[6]提出了一种XML范式——XNF,为XML文档的规范化提供了一种一般化的方法。XNF以避免XML数据的冗余和更新异常为出发点,提出了用于描述XML数据语义的XML函数依赖,以及将任意DTD D 转化为设计良好的DTD D' 的规范化算法,规范化的XML文档既符合结构约束DTD,也符合预定义的语义约束。文[7]基于XML键约束和DTD结构信息,针对XML文档符合键和DTD约束的各种情况给出了可判定性结论和相应的复杂度证明,并且指出:XML文档在遵循DTD又满足一元键约束的情况下是可判定的(相关概念和定义将在第2节中给出)。由上述结论,本文提出了基于XML范式更新XML文档的新方法,首先,类似文[2,4],该方法仍然考虑XML文档上一类很重要的完整性约束——XML键[3],同时也考虑XML数据在结构上的约束——DTD,由此定义XML函数依赖;其次,文[8]提出了遵循DTD的XML间的转换技术TREX,利用TREX能方便地进行转换中的类型推理和检测。为了避免数据冗余和更新异常,在规范化的XML文档上进行XML的更新操作,我们采用TREX方法进行非规范化XML文档的规范化预处理、XML更新中间结果和更新后文档的生成,从而有效处理了具有冗余数据的XML文档的更新,且保证了XML文档的等价性。

2 背景知识

2.1 XML键

文[3]基于路径表达式和树结构的等值(value equality)定义了XML键完整性约束,如定义1。

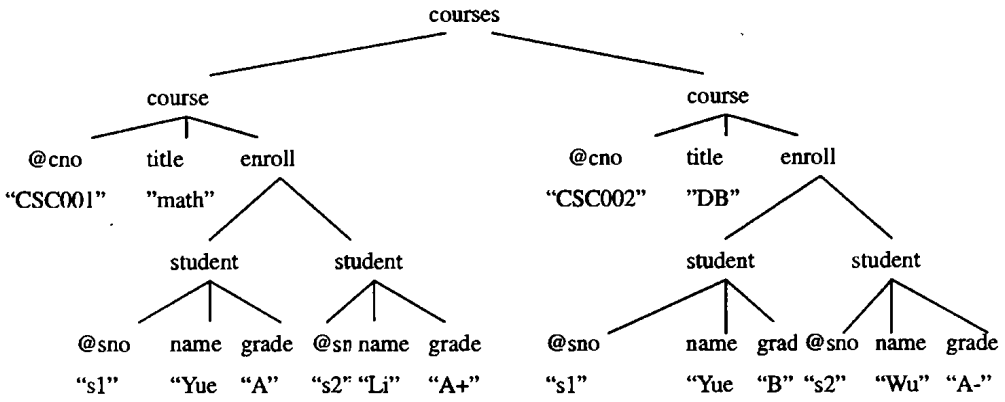

图1 XML 树

定义1(XML 键) XML 树 T 定义为五元组(V,lab,ele,att,$root$),详细定义参见文[3]。T 满足形如(Q,(Q',{P_1,…,P_k}))的约束,其中 Q 为上下文路径、Q' 为目标集路径、P_1,…,P_k 为键路径,在 Q 的范围内 Q' 最终确定了用 P_1,…,P_k 定义键的对象。当且仅当$\forall n\in \square Q \square$,$\forall n1,n2\in n\square Q'\square$,如果对于$\forall i\in[1,\cdots,k]$,$\exists x\in n1\square P_i\square$,$\exists y\in n2\square P_i\square$,$x=_v y$,那么 $n1=n2$,其中=表示"节点相等"(两个节点是同一个),$=_v$ 表示"等值"(两个节点等值,如果这两个节点的标记(*label*)相等,并且以这两个节点为根的子树也同构)。当 $Q=\Phi$ 时称绝对键,否则称相对键,前者是后者的特例。

图1是一个 XML 文档的树(其中@表示属性节点),根据特定的领域知识和语义约束,可以对其定义如下键约束:(*.*course*,{@*cno*})表示@*cno* 唯一决定了 *course*,(*.*course*.*student*,{@*sno*})表示@*sno* 唯一决定了 *student*,(*.*course*,(*title*,{}))表示每个 *course* 只有一个 *title*,键(*.*course*.*enroll*.*student*,(*grade*,{}))表示对每个 *course*,*student* 的 *grade* 值唯一。

2.2 XML DTD

DTD 描述了 XML 数据在结构上的约束,作为 XML 的"模式",文[6]给出了如下形式化的定义。

定义2(DTD) DTD D 为一个五元组:$D=(E,A,P,R,r)$,其中:

·E 是 XML 元素 $t_1.S_2=t_2.S_1$ 类型的有限子集,A 是 XML 属性的有限子集。

·P 是从 E 到元素类型的映射,给定 $\tau\in E$,

则 $P(\tau)$为文本串,或者 $P(\tau)$为如下正则表达式:

$\alpha:\alpha::=\varepsilon|\tau'|\alpha|\alpha,\alpha|\alpha^*$。

·R 是从 E 到 A 的幂集的映射。如果属性$@l\in R(\tau)$,称 $@l$ 为 τ 而定义。

·$r\in E$ 称为根元素类型。

图1中的 XML 文档的 DTD 如图2所示。

```
〈!ELEMENT courses(course * )〉
〈!ELEMENT course(title,enroll)〉
  〈!ATTLIST course cno CDATA #REQUIRED〉
〈!ELEMENT title(#PCDATA)〉
〈!ELEMENT enroll(student * )〉
〈!ELEMENT student(name,grade)〉
  〈!ATTLIST student sno CDATA #REQUIRED〉
〈!ELEMENT name(#PCDATA)〉
〈!ELEMENT grade(#PCDATA)〉
```

图2 SML DTD 示例

2.3 XML 函数依赖及 XML 范式

满足 DTD D 的 XML 文档中可能出现的路径集合记为 $paths(D)$,树元组(Tree Tuple)t 是从 $paths(D)$到 $Vert\cup str\cup\perp$的映射,其中 $Vert$ 为节点集合,Str 为符号串集合,$\perp$表示 *null*。t 的集合记为 $\tau(D)$。XML 树 T 蕴含的最大树元组为 $tuples_D(T)$。函数依赖左部和右部的表达形式由 $paths(D)$定义,而 t 定义了函数依赖中数据值的关系[6]。

定义3(XML 的函数依赖[6]) DTD D 上的函数依赖是形如 $S_1\rightarrow S_2$的表达式,其中 S_1和 S_2为 $paths(D)$的非空有限子集,D 上所有函数依赖的集合记为 $FD(D)$。对于 $S_1\rightarrow S_2\in FD(D)$,并且 T 中的路径集 $paths(T)\subseteq paths(D)$,$S_1\cup S_2\subseteq pats(T)$,如果$\forall t1,t2\in tuplesD(T)$,$t_1.S_1=t_2.S_1$,且 $t_1.S_1\neq\perp$蕴含了 $t_1.S_2=t_2.S_2$,则称 T 满足 $S_1\rightarrow S_2$,记为 $TT\models S_1\rightarrow S_2$。

用 $T\models\Sigma$ 表示:$\Sigma\subseteq FD(D)$,$\forall\varphi\in\Sigma$,均有 $TT\models\varphi$。同理,用 $TT\models(D,\Sigma)$表示:$TT\models D$ 并且 $TT\models\Sigma$。

以图1中的 XML 文档为例,语义约束"具有相同 *sno* 的 *student* 的 *name* 值相同"对应函数依赖 *courses*.*course*.*enroll*.*student*.@*sno*→*courses*.*course*.*enroll*.*student*.*name*.*S*(*S* 为节点文本串)。

定义4(XML 范式[6]) (D,Σ)的闭包记为$(D,\Sigma)^+$,若任一形如 $S\rightarrow P.@l$ 或者 $S\rightarrow p.S$ 的非平凡函数依赖 $\varphi\in(D,\Sigma)^+$,有 $S\rightarrow p\in(D,\Sigma)^+$,则称$(D,\Sigma)$为 XML 范式(XNF)。

上例中,要使(D,Σ)规范化,根据定义4,如下函数依赖必须包含于$(D,\Sigma)^+$中,*course*.*course*.*enroll*.*student*.@*sno*→*courses*.*course*.*enroll*.*strdent*.*name*。

3 基于 DTD 结构的 XML 语义信息

首先,考虑到 XML 在同时遵循 DTD 和键约束的情况下,假设本文中使用的 DTD 无环,XML 键为一元键(即键的定义中只包含一条路径),且每个 XML 节点都有键的定义。本文中 XML 相对键到绝对键的转化、DTD 的无损分解基于上述假设进行。

3.1 从相对键到绝对键的转换

为了从 XML 键约束的定义得到相应形如定义3的 XML 函数依赖,我们首先将相对键转换为等价的绝对键表示形式。

XML 键可具有传递性[3]:相对键$(Q_1(Q'_1,S_1))$中的上下文被另一绝对键$(\varepsilon,(Q'_2,S_2))$作为目标集而定义,即 $Q_1=Q'_2$。为了通过键约束得到 XML 的函数依赖,我们基于上述键的传递性提出了将 XML 相对键转换为绝对键的转换规则,把键约束的目标集转换为从根开始的路径表达式,其中,*的展开通过解析 DTD 得到从 XML 树根到当前节点的路

径表达式，定义如下。

定义5（*的展开） 给定无环 DTD D，从根开始的路径集 $pathsr(D)=\{p|p=r.p'$ 且 $p,p'\in paths(D)\}\subseteq paths(D)$，对于路径表达式 $*.exp$ 或 $Sexp.*.Eexp$，其中 $exp,Sexp,Eexp$ 为不含 * 的路径表达式，* 的展开为映射 $cp:cp(*.exp)=p, p\in pathsr(D), p=p'.exp$；或者 $cp(Sexp*.Eexp)=p, p=sexp.p''.Eexp$，其中 $p'\in pathsr(D)$，$p''\in paths(D)$。

由此上述从相对键到绝对键的转换方法为：

(1) 将键集合中 Σ 的 * 按照定义5的方法展开：对于 $\varphi\in\Sigma$ 中任一如定义5所描述的带有 * 的路径表达式 $pe, pe\leftarrow cp(pe)$。例如 $(*.student,\{@sno\})$ 可转换为 $(courses.course.enroll.student.\{@sno\})$。

(2)将相对键 $(Q,(Q',S))$ 转换为绝对键：$Q\leftarrow\varepsilon, Q'\leftarrow Q.Q'$，即 $(Q.Q'S)$，相对键 $(*.course,(title,\{\}))$ 由(1)可转换为 $(courses.course,(title,\{\}))$，进而转换为绝对键 $(courses.course.title,\{\})$。

将转换后的键(绝对键)集合记为 Σa。

3.2 基于键语义的 XML 函数依赖

通过如下的转换规则，我们由 XML 键语义约束导出 XML 函数依赖。

(1)若 XML 绝对键 $(C,\{P\}\in\Sigma a$，则对应的函数依赖由如下转换规则得到：$C.P\rightarrow C$；

(2)若 XML 绝对键 $(C,\{\})\in\Sigma a$，那么必有另一键 $(C',\{P''\})\in\Sigma a, P\neq null, C'$ 是 C 的子串，设 $|C'|=n, |C|=m$，有 $m>n$ 且 C 中从开始字符长度为 n 的子串与 C' 相等，则对应的函数依赖由如下转换规则得到：$C'.P'\rightarrow C$；

(3)对于 Σa 中的所有绝对键均作如上述(1)，(2)的处理和转化得到 XML 函数依赖集 $FD_m(D)$，并由此得到 $FD_m(D)$ 的闭包 $(D,\Sigma)^+$[6]。

以绝对键 $(courses.course,\{@cno\})$ 和 $(courses.course.title,\{\})$ 为例，由此可得到如下函数依赖：$courses.course.@cno\rightarrow coruses.course, courses.course.@cno\rightarrow courses.course.title$。

4 基于 XML 范式的更新策略

对于非规范化的 XML 文档，数据的冗余性使得某些 XML 节点多次出现在 XML 文档中的不同位置，为了一致地更新这些冗余的 XML 数据，必须进行多次更新操作，直到相关的 XML 节点全部被更新。XML 文档更新前后应符合结构信息 DTD 及键完整性约束，则数据的一致性可得以保证；另一方面，XML 范式(XNF)为此提供了前提，基于 XNF 的更新既能保证语义、结构信息的保持，也避免了上述对冗余数据的多次更新处理，从而避免数据的冗余和更新异常。为此，我们首先基于上述由 XML 键导出的函数依赖对待更新的非规范化 XML 文档作规范化预处理，再执行更新操作，其中 XML 文档的转换采用 TREX 技术[8]。

4.1 基于 TREX 规范化更新 XML 文档

设待更新的 XML 文档为 $T, TT\models(D,\Sigma)$，首先将 D 作无损分解得到 D'，(D',Σ') 分解算法参见文[6]，该方法用“向上移动属性节点”将部分属性节点向上移动，作为其直接函数决定的节点的孩子节点；用“创建新元素节点”将冗余的 XML 数据分离出来作为共享的 XML 数据段。DTD 的无损分解过程中，XML 文档上原先定义的语义得以保持。例如，图2中的 DTD 可无损分解为图3中的 DTD。

```
⟨!ELEMENT⟩courses(coustu,infos)⟩
⟨!ELEMENT coustu(course*)⟩
⟨!ELEMENT course (title,enroll)⟩
  ⟨!ATTLIST course cno CDATA #REQUIRED⟩
⟨!ELEMENT title(#PCDATA)⟩
⟨!ELEMENT enroll(student*)⟩
⟨!ELEMENT student(grade)⟩
  ⟨!ATTLIST student sno CDATA #REQUIRED⟩
⟨!ELEMENT grade(#PCDATA)⟩
⟨!ELEMENT infos(info*)⟩
⟨!ELEMENT number EMPTY⟩
⟨!ATTLIST number sno CDATA #REQUIRED⟩
⟨!ELEMENT name (#PCDATA)⟩
```

图3 无损分解后的 XMLDTD

```
courses->coustu,infos
$coustu=⟨coustu⟩
  let $doc:=document(1"courses.xml")
      for $cou in distinct($doc/course) return
$cou⟨/coustu⟩
$infos=⟨infos⟩
       let $doc:=docunment("courses.xml")
          for $stu in distinct($doc/cours[@cno="cl"]
            /enroll/student)return $stu⟨infos⟩
coustu->course*
$course<-let $coustu:=[[$coustu]]
     for $cou in ($coustu/course)return $cou
course->title,enroll
course=>cno
$title=let $course:=[[$course]]return $course/title
$enroll=let $course:=[[$course]]return $course/enroll
$cno=let $course:=[[$course]]return $course/@cno title->S
$S=let $title:=[[$title]]return $title/text()
... ...
```

图4 用于 XML 文档规范化的 XTG

在规范化的 XML 文档 T 上进行 XML 数据的更新，必须对 T 等价地作规范化处理，即 $TT\models D\Rightarrow T'T\models D'$ 从而得到规范化的 XML 文档 T'。文[8]提出的 TREX 技术是解决 XML 等价转换问题的有效途径，且实验已证明：对于小规模 XML 文档的符合 DTD 的转换，TREX 具有较好的性能，其中 XTG (XML Transformation Grammar)以 Quilt[9]形式为基础稍加改进，定义由符合 D 的 XML 文档 T 到符合 D' 的 XML 文档转换中各元素、属性节点的转换规则产生，而得到目标 XML 文档 T'，XTG 的查询处理结果就是 XML 文档的转换结果[6]。例如，对图1中的 XML 文档 $T, T'T\models\Rightarrow T'T\models D'$ 对应的部分 XTG 如图4，将图1中符合图2DTD 的 XML 文档转换为符合图3DTD 的 XML 文档，完成待更新的非规范化 XML 文档的预处理，规范化结果如图5。

综上，非规范化待更新 XML 文档的规范化过程总结如下：

输入：D,Σ,T；输出：T'

步骤：(1) $(D,\Sigma)\rightarrow(D',\Sigma')$，$(D',\Sigma')$ 是设计良好的 XML 范式[6]；(2) $TT\models D\ \underline{TREX(XTG)}\ T'T\models D'$。

4.2 规范化 XML 文档的更新

更新系统首先由 DTD 及定义在 XML 文档上的键约束得到 XML 函数依赖，由4.1节中的方法对待更新 XML 文档作了规范化，这样更新中的数据不一致就得以消除，然后直接采用文[2,4]中 XML 文档的标注和定位技术，在 XML 文档上进行数据的更新，包括 XML 数据的删除、修改和插入，XML 的更新策略如下(鉴于篇幅的限制，查询重写技术在此不作赘述)：

(1) *更新操作重写和分离* 用户以 XUpdate[4]的表达方式提交更新操作，更新系统对提交的更新操作进行有效性检

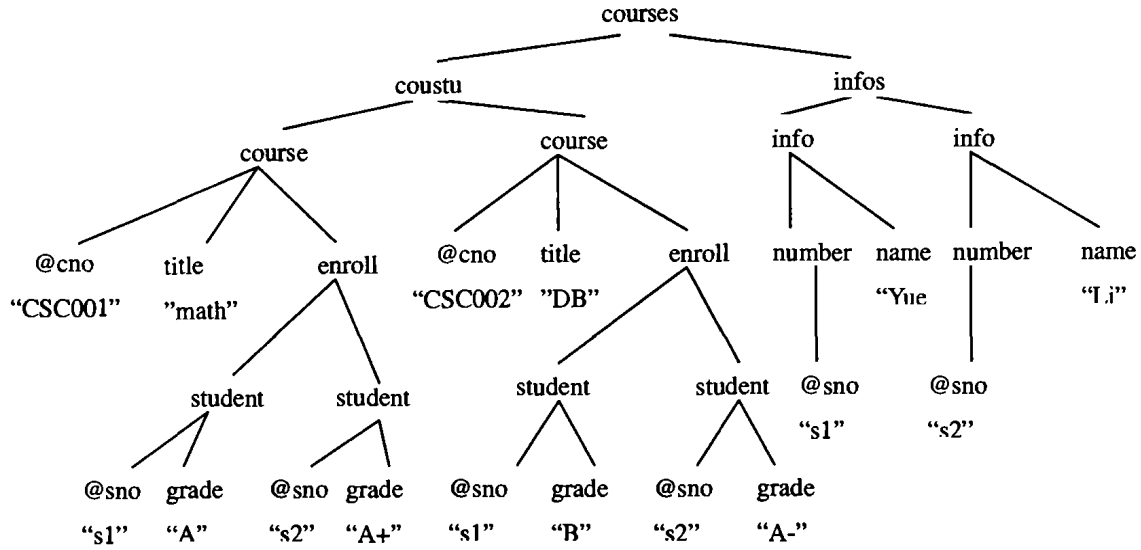

图5 规范化后的 XML 文档

查，特别地，对于 XML 数据的插入，需检测待插入的部分是否符合预定义的键及 DTD 约束。设更新操作表达为 *XUExp*，首先 *XUExp* 经过查询重写将更新操作类型 *XUOP*、更新目标 *XUT*、更新条件 *XUC* 和更新数据 *XUD* 分离出来，要进行的更新操作是：将满足 *XUC* 的 *XUT* 对应的 *XML* 数据经过操作 *XUOP* 进行更新，新增或者修改后的数据为 *XUD*。

(2) *被更新 XML 数据定位* 基于文[2,4]中的 XML 文档标注技术，按照条件 *XUC* 由 *XUT* 在被更新 XML 文档(已作规范化)中定位待更新的 XML 节点或子树。不同之处在于，本文中的 XML 文档并未存储到关系数据库中，标注仅仅在 XML 文档上进行，同时保存标注信息和被标注 XML 数据之间的对应关系。标注的作用类似于索引，查找 XML 节点时由标注信息可减少搜索节点数、减少需要保存的中间结果。例如，对于本文提出的 XML 更新方法，可以对 XML 文档作如下形式的标注：

① 需要标注的节点：

设 V 为文档 T 中节点的集合，Σ 为 XML 键的集合，$\forall v \in V$，若$\exists\ \varphi \in \Sigma, \varphi=(Q,(Q',\{P\}))$，使 v 是键 φ 目标路径 Q' 的终止节点，即 v 被键路径 P 非空的 XML 键 φ 所定义，则 v 被标注。被同一键约束的 XML 节点标注元素名相同。如图1中的 course 节点、student 节点分别被键 $(*.course,\{@cno\})$、$(*.strdent,\{@sno\})$约束，需要被标注。

② 标注及标注对照表：

用文[2]中 XML 文档上的标注策略对满足①的节点进行标注，即添加标注元素〈*Tag_NodeName id*="标注属性值"〉作为被标注节点的父元素节点，对标注元素名相同的节点"标注属性值"递增。例如：对不同 *course* 节点分别作标注：〈*Tag_course id*="1"〉，〈*Tag_course id*="2"〉等。另一方面，对于标注信息，需要保存其与被标注节点的对应关系，对于每个被标注的节点，对照表中存放以其为根的子树中所有叶节点的文本值。对照表的示例如图6。

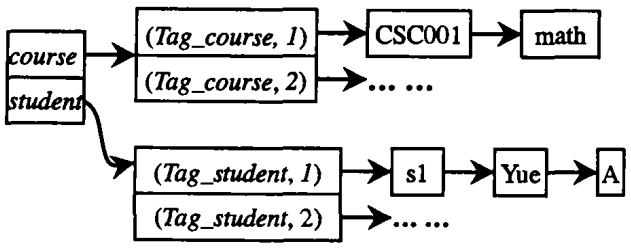

图6 标注对照表

③ 节点定位时，通过查询条件检索得到标注信息〈*Tag_NodeName id*="标注属性值"〉，即可在 XML 文档中进行标注的匹配。但如果 XML 文档较大，图6所示的标注对照表中存储的数据量也较大，但是对层次较深的 XML 节点，利用上述的标注信息进行定位，可以减少大量的 XML 文档搜索和匹配次数、需要保存的路径值信息，作用特别显著。

(3) *文档的一次解析及数据更新* 利用 SAX 解析器扫描被更新的 XML 文档，对于(2)确定的 XML 节点或子树，根据 *XUOP* 和 *XUD* 将新增、修改或删除操作在 XML 文档的解析过程中完成，从而直接在 XML 文档上作 XML 数据的更新。

简言之，上述过程可描述为：

设 T 为待更新 XML 文档，$T\models(D,\Sigma)$，由 D 的无损分解算法及基于 TREX 的 XML 文档规范化策略，得到 $T'\models(D',\Sigma')$，对有效的更新操作，在 T' 上通过解析 T' 及更新策略得到更新后的 XML 文档 Tu (T' *update* Tu)，即 XML 文档在保持 DTD 和键约束下得以更新。更新后的 XML 文档由于其遵循的 DTD D' 是原 DTD D 经过无损分解得到的，D' 可能与 D 形式上并不一致，可将 D' 视为 D 的视图。为了得到符合用户原 DTD 的 XML 更新结果，我们同样基于 TREX 将更新后的 XML 文档进行转换，即 Tu ($Tu\models D'$) *TREX* $Tuser$ ($Tuser\models D$)。上述更新系统的结构如图7。

对于上述的 XML 更新策略，一方面，该方法基于高效的 XML 解析器 SAX 完成，更新的代价主要取决于被更新部分的 XML 数据的删除、写入或替代，而对于更新操作为涉及到的部分，其代价仅仅源于 SAX 扫描 XML 文档的开销，可以忽略不计，因此这种更新策略是增量式的更新，其效率并不依赖于被更新 XML 文档的大小。另一方面，该方法基于 XNF 进行 XML 数据的更新，避免了数据的不一致性，例如，用户提交如下的更新操作：

FOR $\$stu$ IN */courses/course/enroll/student*
WHERE $\$stu/@sno="s1"$
RENAME $\$stu/name="Zhang"$

如果在非规范化的 XML 文档(例如符合图2中的 DTD 的 XML 文档)上更新，由于 *student* 的 *name* 冗余性使得更新中会产生异常，而用本文提出的方法只需要更新 */courses/info/student* 下的 *name*(符合图3中的 *DTD*)，一处更新，全局一致。

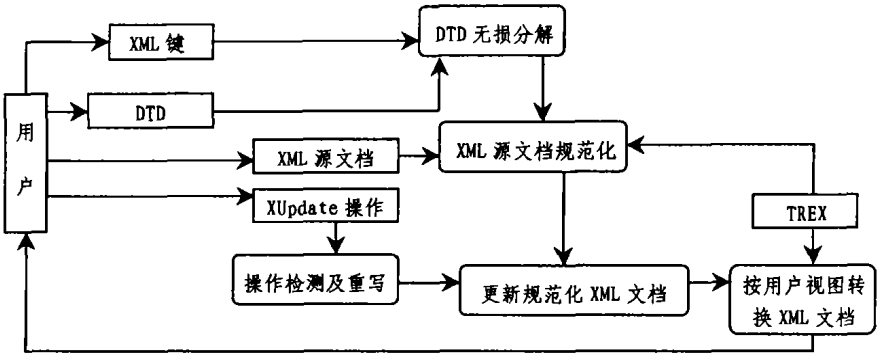

图7 XML 更新系统的结构

总结和展望 本文提出了特别适用于更新非规范化 XML 文档的一种新方法，基于规范化的 XML 文档——"XML 范式，XNF"，保持语义直接更新 XML 数据。该方法中，XML 文档的规范化、XL 数据的更新采用 XML 文档的等价转换技术 TREX 来实现，因此，XML 文档的整个更新过程既保持了 DTD 的结构信息、XML 键的语义约束，也有效地避免了 XML 数据的冗余以及更新而产生的各种异常，同时，DTD 的无损分解、待更新 XML 文档的规范化对用户完全透明，并且 XML 最终更新结果也符合用户视图——原 DTD，该过程中 XML 文档的预处理、中间结果和最终结果的生成均基于有效的 XML 等价转换技术 TREX。本文提出的方法对于规模不大的非规范化 XML 的更新操作具有较好的性能。从原更新操作到符合 XNF 操作的自动转换，更新过程中部分转换的等价性证明、更高效的 XML 文档转换技术、XML 文档规范化过程中的优化策略等方面，是我们正在进行的研究工作。

参 考 文 献

1 Tatarinov I, Ives Z, Halevy A, Weld D. Updating XML. SIGMOD, 2001

2 Yue K, Xu Z, Guo Z, Zhou A. Constraint Preserving XML Updating. APWeb, 2003. 47～58

3 Buneman P, Davidson S, Fan W, Hara C, Tan W C. Keys for XML. WWW10, 2001

4 岳昆，胥正川，周傲英，宫学庆. 用于更新 XML 文档的注释技术. NDBC, 2002. 47～51

5 Cobena G, Abiteboul S, Marian A. Detecting Changes in XML Documents. ICDE, 2002. 41～52

6 Arenas M, Libkin L. A Normal Form for XML Docments. PODS, 2002. 85～96

7 Fan W, Libkin L. On XML Integrity Constraints in the Presence of DTDs. JACM, 2001

8 Zhou A, Wang Q, Guo Z, et al. TREX: DTD-Conforming XML to XML Transformations. SIGMOD, 2003. 670

9 Robie J, Chamberlin D, Florescu D. Quilt: an XML query language. http://www.almaden.ibm.com/cs/people/chamberlin/quilt_euro.html. March 2000

（上接第108页）

8 Moser L E, Amir Y, Melliar-Smith P M, Agarwal D A. Extended virtual synchrony. In: Intl. Conf. Distributed Computing Systems, 1994. 56～65

9 Lamport L. Time, clocks, and the ordering of events in a distributed system. Communications of the ACM, 1978, 21(7): 558～565

10 Tanebaum A S, van Steen M. Distributed System -Principle and Paradigm. Beijing, Tsinghua University Press, 2002

11 Hayden M, Rodeh O. Ensemble Reference annual. http://www.cs.cornell.edu/Info/Projects/Ensemble/doc.html, Aug. 2003

12 Bernstein P, Hadzilacos V, Goodman N. Concurrency Control and Recovery in Database Systems. Addison-Wesley, 1987

13 Jimenez-Peris R, Patino-Martinez M, et al. Improving the Scalability of Fault-Tolerant Database Clusters. In: Proc. of 22nd IEEE Int. Conf. on Distributed Computing Systems (ICDCS'02), 2002. 477～484

14 Wiesmann M, Pedone F, Schiper A, Kemme B, Alonso G. Database Replication Techniques: a Three Parameter Classification. In: Proc. of the 19th IEEE Symposum on Reliable Distributed Systems (SRDS2000), Numberg, Germany, Oct. 2000

15 Information & Communcations Systems Research Group, ETH Z¨urich and Laboratoire de Syst`emes d'Exploitation (LSE), EPF Lausanne. DRAGON: Database Replication Based on Group Communication, May 1998. http://www.inf.ethz.ch/department/IS/iks/research/dragon.html