ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.10, October 2007, pp.2495−2506
DOI: 10.1360/jos182495

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于本体的异构信息集成查询划分及转换*

李 剑+, 宋靖宇, 钟 华

(中国科学院 软件研究所 软件工程技术研究开发中心,北京 100080)

Ontology-Based Query Division and Reformulation for Heterogeneous Information Integration

LI Jian+, SONG Jing-Yu, ZHONG Hua

(Technology Center of Software Engineering, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: +86-13683253892, E-mail: lijian767@hotmail.com, http://www.iscas.ac.cn

Li J, Song JY, Zhong H. Ontology-Based query division and reformulation for heterogeneous information integration. *Journal of Software*, 2007,18(10):2495−2506. http://www.jos.org.cn/1000-9825/18/2495.htm

Abstract: People can obtain information from heterogeneous data resources by semantic querying based on ontology. The difficult problem is how to reformulate a global query into multiple sub-queries over those local data bases. This paper presents a kind of uniform expression for queries of concept instances and a method in dividing the query expression into the local ones. All local query results can be integrated for the users' purposes. Using these methods to query integrated data resources, users can get right results they desire.
Key words: ontology; query division; query equivalence; heterogeneous information integration

摘 要: 使用本体赋予信息语义能够帮助用户准确查询所需要的信息.基于本体的异构信息集成中的关键问题是如何实现全局本体概念实例查询到局域信息数据查询的变换.提出了一种本体概念实例查询的操作表示,并基于这一查询操作表示给出了将全局查询划分为局域查询的方法,局域查询结果经过集成和转换后以统一的形式返回给用户.使用该方法来查询所集成的数据来源,可以获取用户所需要的正确查询结果.
关键词: 本体;查询划分;查询等价;异构信息集成
中图法分类号: TP311 **文献标识码**: A

随着 WWW 的发展,用户需要获取的信息量越来越大,需要获取的信息类型也越来越多,如何准确地在分布和异构的信息来源中获取信息是一个非常重要的研究问题.

1 引 言

本体(ontology)是一种知识表示形式,现实世界实体以及实体之间的联系被抽象表示成本体中的概念以及概念之间的关系.本体中定义的实体具有确定的语义,使用本体中的实体去标记定义信息,便将这些信息看作是

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2004AA112010 (国家高技术研究发展计划(863)); the National Basic Research Program of China under Grant No.2002CB312005 (国家重点基础研究发展计划(973))
Received 2006-04-26; Accepted 2006-10-09

本体实体的实例.用户通过查询本体中实体的实例可以准确地获取所需要的信息.这也是语义 Web 的核心思想.

基于本体语义集成异构信息来源的系统结构一般分为 3 层:底层是各个分布的异构数据源;中间层中存在一个描述特定用户群关心领域的全局本体,它对每个用户具有统一的语义,同时,它与底层的异构数据源有着特定的对应关系;上层是用户查询访问层.用户将自己的查询需求表示为对本体概念实例的查询,根据本体和局域数据源的对应关系,中间层将这些查询分解转换为底层局域数据源的局域查询.获取的局域查询结果之间需要根据特定的关系进行集成,并将其转换成统一的形式作为全局查询结果返回给用户.

在基于本体的信息集成中,如何表示用户对本体概念实例的查询、如何建立本体到异构数据源的映射关系、如何根据这些关系将全局查询转换划分为若干个局域查询、如何集成局域查询结果,这些问题都需要解决.

本文以结构化关系数据和半结构化 XML 文档为例,给出了全局本体到 XML 和关系数据的映射定义方式,根据对本体概念实例查询的操作语义表示,在上述映射定义下将全局查询划分为具体的局域查询.

2 本体和局域数据映射

为了实现基于本体的信息获取,需要形式定义本体以及本体与局域数据的映射方式.

2.1 本体和本体查询语言

本体可以看作为包含下列两种类型定义的集合:一种是对应于现实世界实体的概念(concept);另一种是描述现实中实体间联系的概念与概念之间的关系(role).

定义 1. 一个本体o($o\in O$,O为本体的集合)表示为一个二元组$\langle C,R\rangle$,C为概念的集合,R为概念之间关系的集合.假设$c_1,c_2\in C,r\in R$,且$c_1\ r\ c_2$关系成立,c_1称为r的源概念,c_2称为r的目的概念;同时,对于c_1概念的一个实例vc_1,存在着若干个c_2概念的实例vc_{2i},它们之间存在着r关系,表示为$vc_1\ r\ vc_{2i}$.$\leq(o)$($\leq(o)\subseteq R$)表示本体o中子类关系集合,如果$r\in\leq(o)$,$c_1\ r\ c_2$表示c_1是c_2的子类概念.$K(o)$($K(o)\subseteq R$)表示本体o中主键值关系的集合,当$r\in K(o)$,$vc_{1i}\ r\ vc_{2i}$,$vc_{1j}\ r\ vc_{2j}$时,如果$vc_{2i}=vc_{2j}$,则$vc_{1i}=vc_{1j}$,反之亦然.

图 1 给出了一个描述艺术家(artist)和他创作的艺术作品(artifact)以及所属的艺术流派(genre)等信息的本体o,其中,Artist 是 Person 的子类概念;虚线椭圆表示的是 String,Int 等基本类型概念;虚线箭头表示从其父类概念继承来的关系,例如:概念 Person 和概念 Country 之间有 nationality 关系,那么,Person 的子类概念 Artist 和 Country 之间也继承这一 nationality 关系;本体 o 的主键值关系用粗黑线箭头表示.例如:name 是主键值关系,它表示名字相同的 Artist 默认为同一个人.

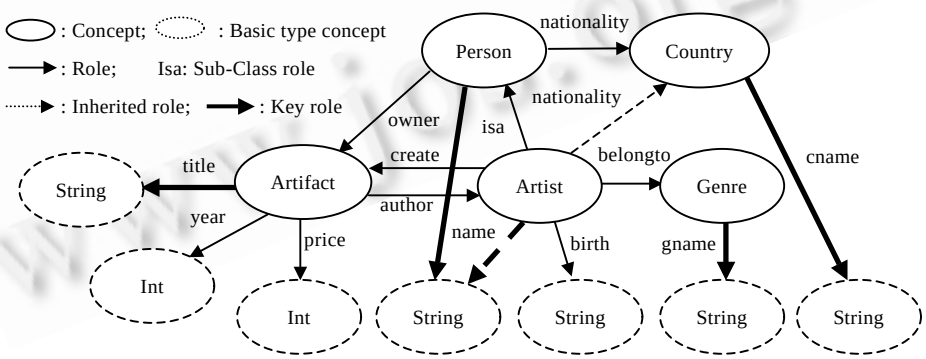


Fig.1 An ontology expresses information about artists and artifacts, etc.

图 1 一个表示艺术品、艺术家等信息的本体

定义 2. 对本体概念实例的查询可以表示为下述形式:

Select $L_1,L_2,\ldots,L_n$

From $\ldots,c_i\ L_i,\ldots,L_{j1}\ .r_j\ L_{j2},\ldots$

Where $pred(L_1,L_2,\ldots,L_n)$

其中,Select中的$L_1,\ldots,L_n$是查询结果变量表示;c_i L_i表示L_i是c_i概念的实例,L_{j1}. r_j L_{j2}表示L_{j1},L_{j2}满足r_j关系,$pred(L_1,L_2,\ldots,L_n)$表示将$L_1,L_2,\ldots,L_n$作为变量的条件表达式,查询结果是满足上述条件的概念实例集.表 1 给出了对于图 1 本体的两个概念实例查询语句例子:$\boldsymbol{Q}_1$查询作者是意大利人并且价值大于 500 的作品,获取作品名字以及创作者的名字和所属派别名;$\boldsymbol{Q}_2$查询所有人并获取他们的名字.

Table 1 Two example queries for concepts' instances

表 1 本体的两个概念实例查询例子

$\boldsymbol{Q}_1$: Select *t, n, gn* From *Artifact a, Artist p, p .name n, p .create pa, a .title t, a .price pr,* *p .belongto g, g .gname gn, p .nationality mc* Where *pr*>500 and *a*=*pa* and *mc*=Select *c* From *Country c, c .cname cn* Where *cn*="italy"
$\boldsymbol{Q}_2$: Select *n* From *Person p, p .name n*

2.2 本体与局域数据映射

定义 3. 本体$o=\langle C,R\rangle$,本体o到实例集合的映射为$I,I:o\rightarrow\langle\Delta,\Delta\times\Delta\rangle$,$\Delta$为实例集合.对于$c_1,c_2\in C,r\in R,c_1\ r\ c_2$,如果$c_1^I=\Delta_1\subseteq\Delta$, $c_2^I=\Delta_2\subseteq\Delta$,则$r^I\subseteq\Delta_1\times\Delta_2$,$r^I=\{\langle d_1,d_2\rangle|d_1\in\Delta_1,d_2\in\Delta_2,d_1\ r\ d_2(d_1,d_2$之间满足$r$关系)$\}$. c_1^I, c_2^I 分别表示I映射下c_1,c_2对应的概念实例集合,r^I表示I映射下关系r的解释实例集合.对于$c\in C,Concept(c,L)$表示概念c的实例集合,并将概念实例附上L标签.在I映射下获取的概念实例结果集$Concept(c,L)^I=\{[v:L]|v\in c^I\}$.

关系映射函数$f_{Rv}:\Delta,R\rightarrow\Delta,f_{Rv}$是一个多值函数,$f_{Rv}(d_1,r)=d_{2i},\ldots,d_{2j}$,当且仅当$d_1$是$r$源概念的实例,$d_{2i},\ldots,d_{2j}$都是$r$目的概念的实例,并且$d_1\ r\ d_{2i},\ldots,d_1\ r\ d_{2j}$都成立.在$I$映射下,$f_{Rv}(d_1,r)^I=d_{2i},\ldots,d_{2j}$,当且仅当$\langle d_1,d_{2i}\rangle\in r^I,\ldots,\langle d_1,d_{2j}\rangle\in r^I$.

在上述定义中,关系r被映射解释为一个二元向量集合,向量的两个分量分别为之间存在r关系的两个概念实例.而关系映射函数则是在这一映射定义下,寻找和源概念实例值d_1满足r关系的目的概念实例值,所获得的值可能为多个.

本体到局域数据源的映射可以看作是局域映射,而本体到多个局域数据源的全局映射是这些局域映射复合的结果.

定义 4. 对于本体$o=\langle C,R\rangle$,本体o到两个局域数据源的局域映射分别为$I_1:o\rightarrow\langle\Delta_1,\Delta_1\times\Delta_1\rangle,I_2:o\rightarrow\langle\Delta_2,\Delta_2\times\Delta_2\rangle$,那么,本体对于这两个数据源的全局映射$I(I=I_1\cup I_2):o\rightarrow\langle\Delta_1\cup\Delta_2,(\Delta_1\cup\Delta_2)\times(\Delta_1\cup\Delta_2)\rangle,\forall c\in C,r\in R(c'=c^{I_1}\cup c^{I_2},\ r^I=r^{I_1}\cup r^{I_2})$.

定义 5. 本体$o=\langle C,R\rangle$到数据源的映射为$I,\forall c_1,c_2\in C,r\in R,c_1\ r\ c_2$,如果$c_1,r$都有映射定义,那么$c_2$也存在对应的映射定义;如果$c_2,r$都有映射定义,那么$c_1$也存在对应的映射定义;如果$c_1,c_2$都有映射定义,那么$r$也存在对应的映射定义;如果$r$有映射定义,那么$c_1,c_2$也有对应映射定义,则$I$对于$o$和数据源是一个完备的映射.下文涉及的映射都是完备的映射(我们假设本体o中的String,Int等基本类型概念在任何到XML和关系数据的映射中都有映射定义,并且此定义中的r不包括子类关系isa).

2.3 本体到XML文档映射

定义 6. 对于一个XML文档D_{xml},$Path$为XML文档中的绝对路径,$N(Path)$表示通过这条路径找到的所有结点实例;$Get(N(Path))$表示这些结点实例值的集合;$Get(N(Path),L)$表示获取这些结点实例的值,并将这些值都标记上L标签;$vPath$为XML文档中的相对路径,相对路径结点实例获取函数$f_{getPath}(d_1,vPath)=v_1,\ldots,v_m$($f_{getPath}$是多值函数),它表示获取所有$d_1$所在结点$n$经过$vPath$相对路径所到达的结点的值,其值可能有多个.

定义 7. 设本体$o=\langle C,R\rangle$和XML文档数据之间,存在着概念到XML文档绝对路径以及关系到XML文档相对路径的对应集合,此对应集合定义的本体o到XML文档的映射为$I_{xml}:c,c_1\in C,r\in R,c\ r\ c_1$.如果$p$为概念$c$对应的一个绝对路径,则$c^{I_{xml}}=Get(N(p))$,这样,$Concept(c,L)^{I_{xml}}=Get(N(p),L)$;如果 $vPath$ 是 r 对应的相对路径,则 $r^{I_{xml}}=$

$\{\langle d_1,d_2\rangle|d_1\in c^{I_{xml}},d_2=f_{getPath}(d_1,vPath)\}$,这样, $f_{Rv}(d_1,r)^{I_{xml}}=f_{getPath}(d_1,vPath)$.

表 2(b)中的对应关系就描述了图 1 本体o到表 2(a)XML文档的一个映射I_{xml}.其中,“//creator”绝对路径对应的一个结点实例“⟨creator name=“Stefano Vitale”/⟩”(设其为dc)表示意大利画家Vitale,他是Artist概念的一个实例.create本体关系对应的相对路径为“/..”(表示此结点的父结点),$f_{getPath}(dc,\text{“/..”})$表示寻找值为$dc$结点的父结点(“⟨artifact⟩”结点),获取它的值,也就是寻找Vitale创作的所有艺术作品,结果可能为多个,其中一个XML结点片断便描述了名为“When The Wind Stops”的名画.

Table 2 Mappings between the ontology and XML document

表 2 本体与 XML 文档的映射关系

(a) A XML fragment about artifact:

```
⟨artifact title=“When The Wind Stops”⟩
    ⟨creator name=“Stefano Vitale”/⟩
    ⟨price⟩1000⟨/price⟩
    ⟨time⟩⟨year⟩1860⟨/year⟩
        ⟨month⟩4⟨/month⟩⟨/time⟩
    ⟨ownerlist⟩
        ⟨owner name=“owner1”/⟩
        ⟨owner name=“owner2”/⟩
    ⟨/ownerlist⟩
⟨/artifact⟩
```

(b) Mapping I_{xml} between ontology o and XML

Index	Concepts and roles	Paths in XML	Mapping of the *SC*
*M*1	Artifact	//artifact	
*M*2	create	/..	*M*3
*M*3	Artist	//creator	
*M*4	Person	//owner	
*M*5	name	/@name	*M*3
*M*6	name	/@name	*M*4
*M*7	price	/price	*M*1
*M*8	year	/time/year	*M*1
*M*9	title	/@title	*M*1
*M*10	author	/creator	*M*1
*M*11	owner	/../..	*M*4

“//”: Absolute path; “/”: Relative path; “..”: Parent node; “@”: Attribute node; “*SC*”: Source concept

2.4 本体到关系数据源映射

定义 8. 在关系数据库中,$R^{db}(a)$表示一个名为a的数据库关系.$GetR(R^{db}(a))$表示获取数据库关系a所有的元组实例;$GetR(R^{db}(a),L)$表示获取所有的数据库关系元组实例,并将这些元组实例加上L标签.$attrList\rightarrow attrList'$表示数据库关系中的函数依赖,它可以是两个有联系的关系表之间的主键和外键依赖关系,也可以是一个关系表内属性项之间的函数依赖关系.函数依赖元组获取函数$f_{getF}(d_1,attrList\rightarrow attrList')$表示按照这一函数依赖关系,根据元组$d_1$中$attrList$属性列的值,寻找那些$attrList'$属性列值与之相等的元组实例.例如,根据表 3(a),$f_{getF}(d_1,\text{“FK(sid }R^{db}\text{(Artist))}\rightarrow\text{PK(SID }R^{db}\text{(Genre))”})$表示:$d_1$是$R^{db}$(Artist)的一个关系元组实例,根据$d_1$中外键属性“sid”对应的值$v$,去寻找$R^{db}$(Genre)关系表中主键属性“SID”的值和$v$相等的元组实例.

定义 9. 设本体$o=\langle C,R\rangle$到关系数据库存在着本体概念到关系数据库关系,本体关系到$attrList\rightarrow attrList'$函数依赖的对应集合,此对应集合定义的$o$到关系数据库的映射为$I_{db}$:$c,c_1\in C,r\in R,c\ r\ c_1$.如果$a$为$c$对应的数据库关系,则$c^{I_{db}}=GetR(R^{db}(a)).a$可以是关系数据库中的实际数据表,也可以是关系视图.这样,$Concept(c,L)^{I_{db}}=GetR(R^{db}(a),L)$;如果$attrList\rightarrow attrList'$为$r$对应的函数依赖,则$r^{I_{db}}=\{\langle d_1,d_2\rangle|d_1\in c^{I_{db}},\ d_2=f_{getF}(d_1,attrList\rightarrow attrList')\}$,这样, $f_{Rv}(d_1,r)^{I_{db}}=f_{getF}(d_1,attrList\rightarrow attrList')$.

表 3(b)描述了图 1 本体到表 3(a)关系数据的一个映射I_{db},其中,本体概念Artist对应关系表R^{db}(Artist),关系表R^{db}(Artist)中的每一个关系元组都表示概念Artist的实例.同时,$M2$ 中本体name关系对应关系函数依赖“PK(Name R^{db}(Artist))→(Name R^{db}(Artist))”,它表示概念Artist实例元组中的Name属性项就是它的名字.

Table 3 Mappings between the ontology and DB

表 3 本体与关系数据库的映射关系

(a) Schemas of a relational database about artists' information:

Artist ⟨#Name, BirthYear, sid, motherland⟩

Genre ⟨#SID, Sname, Sintro⟩ "#xxx": Primary key

(b) Mapping I_{db} from ontology o to the relational database

Index	Concepts and roles	Relations and dependencies in the relational database	Mapping of the *SC*
*M*1	Artist	R^{db}(Artist)	
*M*2	name	PK(Name R^{db}(Artist))→(Name R^{db}(Artist))	*M*1
*M*3	birth	PK(Name R^{db}(Artist))→(BirthYear R^{db}(Artist))	*M*1
*M*4	Genre	R^{db}(Genre)	
*M*5	gname	PK(SID R^{db}(Genre))→(Sname R^{db}(Genre))	*M*4
*M*6	belongto	FK(sid R^{db}(Artist))→PK(SID R^{db}(Genre))	*M*1
*M*7	Country	$\pi^{db}_{motherland}$ R^{db}(Artist)	
*M*8	nationality	PK(Name R^{db}(Artist))→(motherland R^{db}(Artist))	*M*1
*M*9	cname	PK(motherland ($\pi^{db}_{motherland}$ R^{db}(Artist)))→(motherland R^{db}(Artist))	*M*7

R^{db}: Relation; PK: Primary key; FK: Foreign key; π^{db}: DB's division; "*SC*": Source concept

3 本体概念实例查询表示

为了实现本体概念实例的全局查询到局域查询转换,需要有一种方式定义查询操作以及相关的等价转换方式.借鉴文献[1,2]中对关系数据查询和对 XML 查询的操作定义,我们定义对本体概念实例查询的操作表示.

3.1 查询操作表达式

下面首先定义元组以及一系列对元组集合操作的算子,通过它们组合起来的元组操作表达式作为本体概念实例查询的操作表示.

令τ=[v_1:L_1,v_2:L_2,…,v_n:L_n]是一个元组,其中,v_i表示本体概念实例值,L_i为对应的标签.我们用"{τ}"表示包含若干个元组实例的集合.除对元组的基本操作如*append*(元组添加),*combine*(元组合并)以及对元组集合的基本操作∪(并集)、∩(交集)、/(差集)等通常定义之外,还可以定义如下元组集合操作:

变量绑定操作$\chi_{vBind}[L,f_{Rv}(L',r)]$:{$\tau$}→{$\tau$},$\chi_{vBind}[L,f_{Rv}(L',r)]$({$\tau$})={*append*($\tau$,[$v_i$:$L$])|∀ τ∈{τ},$f_{Rv}(d_1,r)$=v_1,…,v_n,d_1为τ中L'标签对应值,∀v_i,v_i∈{v_1,…,v_n}},f_{Rv}是关系映射函数,它根据关系r的源概念实例值(τ中L'标签对应值)获取目的概念实例值,χ_{vBind}操作将这些值标记以L标签,并添加到对应的元组中;

选择操作σ[*Pred*(L)]:{τ}→{τ},σ[*Pred*(L)]({τ})={τ|*Pred*(d),d 为τ中 L 标签对应值},*Pred* 为条件表达式,σ表示获取那些满足 *Pred* 条件的元组;

投影操作$\pi[L_k,…,L_l]$:{τ}→{τ},$\pi[L_k,…,L_l]$({τ})={τ_1|τ中标签属于{L_k,…,L_l}的分量被保留,将它们作为τ_1的各个分量},π除去τ中不需要的元组分量;

连接操作∞:{τ},{τ}→{τ},

{τ_1}∞[L_i=L_j]{τ_2}={*combine*(τ_1,τ_2)|τ_1分量值与τ_2中标签相同的分量值对应相等,并且τ_1中L_i标签对应值和τ_2中L_j标签对应值相等};{τ_1}∞{τ_2}={*combine*(τ_1,τ_2)|τ_1分量值与τ_2中标签相同的分量值对应相等}.∞类似关系数据库中的连接操作,根据元组分量的等价关系对两个元组进行连接合并.

令⊙,⊕分别表示对元组集合操作的一元、二元操作算子,设{L_i,…,L_j}是一个元组标签集.我们用 $\oplus_{L_i,…,L_j}$ 表示相应元组集合操作仅限于对指定标签集合{L_i,…,L_j}对应的元组分量进行.

定义 10. 查询本体概念实例的查询操作表达式e可以是e=*Concept*(c,L)或e=⊙e_1|e_1⊕e_2.这里,e_1,e_2均为查询操作表达式.

若 e 是查询操作表达式,则 e{L/L'}也是,它表示将 e 中所有 L 标签都换成 L'.

定义 11. 如果查询操作表达式 e 中不含有二元操作算子,那么称 e 为单树表达式.

定义 2 形式的本体概念实例查询语句可以转换为上述定义形式的查询操作表示.例如表 1 中Q_1,Q_2可以转

换为表 4 中的查询操作表示.

Table 4 Query operation expressions of Q_1 and Q_2

表 4 Q_1和Q_2的查询操作表示

Q_1's query operation expression:
$\pi[t,n,gn](\sigma[pr>\text{“500”}]\chi_{vBind}[pr,f_{Rv}(a,price)]\chi_{vBind}[t,f_{Rv}(a,title)]Concept(Artifact,a)\infty[a=pa]$ $\chi_{vBind}[mc,f_{Rv}(p,nationality)]\chi_{vBind}[gn,f_{Rv}(g,gname)]\chi_{vBind}[g,f_{Rv}(p,belongto)]\chi_{vBind}[pa,f_{Rv}(p,create)]\chi_{vBind}[n,f_{Rv}(p,name)]$ $Concept(Artist,p)\infty[mc=c]\sigma[cn=\text{“italy”}]\chi_{vBind}[cn,f_{Rv}(c,cname)]Concept(Country,c))$
Q_2's query operation expression: $\pi[n]\chi_{vBind}[n,f_{Rv}(p,name)](Concept(Person,p)\cup Concept(Artist,p))$

如果对任意的查询映射,两个查询操作表达式的查询结果集是相同的,则这两个查询操作表达式是查询等价的,在元组集合代数中的两个等价表达式必然也是查询等价的.基于χ元组集合代数的表达式等价变换理论及其扩展,由于篇幅限制,这里不再赘述.

3.2 查询操作图

为了便于表示和计算,我们可以将一个查询操作表达式和一个查询操作图对应起来.

定义 12. 对于定义 10 方式定义的查询操作表达式 e,可以构建其对应的查询操作图 $D(e)$,$D(e)$可以由单个或多个树组成.每个树 t 对应的查询操作表达式为 $E(t)$.单树表达式 e 只对应一棵树 t,表示为 $T(e)$.如果 $T(e)$中只有一条从根结点 r 到叶结点 l 的向下路径,那么它可以表示为 $T[\text{“}r\rightarrow\ldots\rightarrow l\text{”}]$.$D(e)$中所有结点的集合为 N,每个结点表示为$\langle L,c\rangle$,L 为结点的标签,可以称此结点为 L 结点,c 是这一结点对应的本体概念.

可采用如下方法构造查询表达式 e 对应的查询操作图:

(1) 如果 $e=Concept(c,L)$,则构建一个根结点$\langle L,c\rangle$;

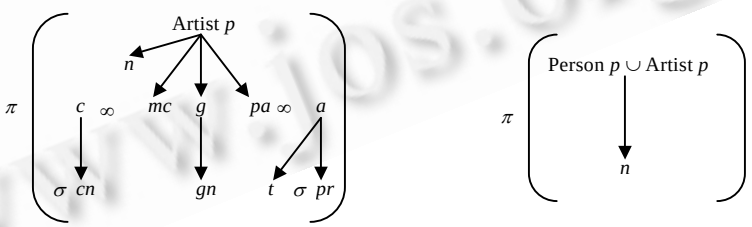
(2) 如果$e=\chi_{vBind}[L_2,f_{Rv}(L_1,r)]e_1$,则在$e_1$对应的查询操作图$D(e_1)$中找到标签为$L_1$的结点$L_1$,从$L_1$结点出发构造一条有向边,生成边指向的$L_2$结点为$\langle L_2,c\rangle$,这条有向边对应的关系为$r$,$L_2$结点对应$r$关系的目的概念$c$;

(3) 如果$e=e_1\infty e_2$(或者$e_1\infty[L_i=L_j]e_2$),则在查询操作图$D(e_1)$,$D(e_2)$之间构建一条“∞”(或者“$\infty[L_i=L_j]$”)边,其连接的结点为在两个图中标签相同的结点(或者是$D(e_1)$的L_i结点与$D(e_2)$的L_j结点);

(4) 如果$e=\odot e_1$,$\odot\in\{\sigma,\pi\}$,则在$D(e_1)$查询操作图的外围加上“⊙[]”,对$\sigma[Pred(L)]$可在L结点处标记“σ”;

(5) 如果$e=e_1\oplus e_2$,$\oplus\in\{\cup,\cup_{L_k,\ldots,L_l},\cap,\cap_{L_k,\ldots,L_l},/,/_{L_k,\ldots,L_l}\}$,则在查询操作图$D(e_1)$,$D(e_2)$之间构建“⊕”连接.

图 2(a)、图 2(b)分别表示了Q_1,Q_2所对应的查询操作图,其中Artist,Person表示结点对应的概念,为省略起见,其他结点只标记了结点的标签,它们与Q_1,Q_2查询表达式中的对应标签相同.



(a) Q_1's query diagram
(a) Q_1的查询操作图

(b) Q_2's query diagram
(b) Q_2的查询操作图

Fig.2 Query diagrams of Q_1 and Q_2

图 2 Q_1和Q_2的查询操作图

查询操作图之间的等价变换是基于查询操作表达式的等价关系,例如,如果单树表达式e对应的树$T(e)$在结点L_p处分裂成两棵树为t_1,t_2,设t_1,t_2对应的单树表达式为e_1,e_2,则$e=e_1\infty e_2$(证明略).因此,树$T(e)$与在L_p结点采用“∞”连接t_1,t_2的查询图等价.通过此等价关系可以将一棵查询树进行等价分解,也可以将两棵“∞”连接的查询树等价合并成一棵树.

4 全局查询到异构局域查询

在将本体映射到分布的局域数据时,全局映射I表示为若干个本体到局域数据映射的复合形式($I=I_1\cup\ldots\cup I_n$).我们首先考虑两个局域数据映射复合的情况,多个映射的复合被看作是重复这一复合操作的结果.

4.1 基于映射的查询等价

定义 13. 本体o到数据源的映射为I,e为一个查询操作表示,如果e中所涉及的概念和关系在映射I中都有映射定义,则在I映射下,『e』^{I}是一个合法表达式,我们将『e』^{I}看作在I映射下e查询所获取的查询结果集,表达式的等价变换不会影响其查询结果集,但在全局映射下的合法表达式不一定在局域映射下也是合法的.

定义 14. 如果I是本体$o=\langle C,R\rangle$到两个局域数据源的全局映射,并且它是两个局域映射的复合($I=I_1\cup I_2$),设$C^{I_1}, C^{I_2}\subseteq C, C^{I_1}, C^{I_2}$分别是在$I_1$和$I_2$中有映射定义的概念集合.设『$e$』$^{I_1\cup I_2}$对映射$I(I=I_1\cup I_2)$是一个合法表达式,$N$为$e$所对应的查询图$D(e)$中所有结点的集合,结点$n\in N$,我们定义:$N^{I_1\cap I_2}(e)=\{n|n$对应的概念$c$在$I_1$和$I_2$中都有映射定义,$c\in C^{I_1}\cap C^{I_2}\}$;$N^{I_1-I_2}(e)=\{n|n$对应的概念$c$仅在$I_1$中有映射定义,$c\in C^{I_1}-C^{I_2}$(集合差集)$\}$;$N^{I_2-I_1}(e)=\{n|n$对应的概念$c$仅在$I_2$中有映射定义,$c\in C^{I_2}-C^{I_1}\}$.

显然,如果$I=I_1\cup I_2$,并且e对应的查询图$D(e)$(或者查询树$T(e)$)中所有的结点都属于$N^{I_1\cap I_2}(e)$,则:

『e』$^{I_1\cup I_2}$=『e』$^{I_1}\cup$『e』^{I_2}(并不是所有的情况下上述等价都成立,『e』^{I_1}和『e』^{I_2}必须都是合法表达式).

定理 1. 对于一个映射I和查询表达式e_1,e_2,『e_1』^{I}和『e_2』^{I}都是合法表达式(表示e_1和e_2涉及到的所有概念和关系在映射I中都有对应的映射定义),则:『$e_1\oplus e_2$』^{I}=『e_1』$^I\oplus$『e_2』^{I},$\odot$『e_1』^{I}=『$\odot e_1$』^{I}.

定理 2. 对于复合映射$I(I=I_1\cup I_2)$,单树表达式e对应的查询树为$T(e)$,『e』^{I_1}是合法表达式,$N^{I_1-I_2}(e)$非空,并且在$T(e)$中不存在p,a结点都属于$N^{I_1\cap I_2}(e)$的"$p\to a$"连接边(表示所有边连接的两个结点中至少有一个结点对应的概念仅在I_1中有映射定义,而在I_2中没有映射定义),则:『e』$^{I_1\cup I_2}$=『e』^{I_1}.

证明:(1) 当$T(e_0)$只有一个结点时,因为$N^{I_1-I_2}(e_0)$非空,此结点对应概念仅在I_1中有映射定义,显然成立:『e_0』$^{I_1\cup I_2}$=『e_0』^{I_1}.假设$T(e_1)$只有两个结点,设$e_1=E(T["p\to a"])$,p,a对应概念在I_1中都有映射定义,至少有一个结点对应概念仅在I_1中有映射定义,根据完备映射定义,"$p\to a$"边对应的关系r也只能在I_1中有映射定义,这样,$r^{I_1\cup I_2}=r^{I_1}$.在I映射下的查询中,若p对应概念实例值v_p属于I_1映射数据源,则$f_{Rv}(v_p,r)$获得的值v_a属于I_1映射数据源;如果v_p不属于I_1映射数据源,则f_{Rv}没有返回值.因此,r对应的绑定操作$\chi_{vBind}[a,f_{Rv}(p,r)]$将只保留那些$v_p$和$v_a$都属于$I_1$映射数据源的结果元组$[v_p:p,v_a:a]$,$\langle v_p,v_a\rangle\in r^{I_1}$.同时,在$I_1$映射下,$e_1$查询结果集也为满足$\langle v_p,v_a\rangle\in r^{I_1}$条件的$[v_p:p,v_a:a]$结果元组集合,因此证明:『$e_1$』$^{I_1\cup I_2}$=『$e_1$』^{I_1}.

(2) 设查询操作树$t=T(e)$,『e』^{I_1}是合法表达式,$N^{I_1-I_2}(e)$非空,在t中不存在p,a结点都属于$N^{I_1\cap I_2}(e)$的"$p\to a$"连接边,假设『$E(t)$』$^{I_1\cup I_2}$=『$E(t)$』^{I_1}.

设查询操作树t'为在树t的某一个结点p上增加一条连接到a的路径的结果,a结点对应的概念c在I_1映射中有映射定义,并且p,a对应的概念c_p,c_a中至少有一个仅在I_1中有映射定义,则t'可以在p结点处分解为t和$T["p\to a"]$(设为t'')两个子树,子树之间用"∞"连接,则$E(t')=E(t)\infty E(t'')$,因此,

$$『E(t')』^{I_1\cup I_2}=『E(t)』^{I_1\cup I_2}\infty『E(t'')』^{I_1\cup I_2}=『E(t)』^{I_1}\infty『E(t'')』^{I_1\cup I_2}.$$

根据(1):『$E(T["p\to a"])$』$^{I_1\cup I_2}$=『$E(T["p\to a"])$』^{I_1},则:『$E(t')$』$^{I_1\cup I_2}$=『$E(t)\infty E(t'')$』^{I_1}=『$E(t')$』^{I_1}.

综合(1)、(2),归纳得证. □

定理 3. 如果$I=I_1\cup I_2$,单树表达式e对应的查询树为$T(e)$,在$T(e)$中存在"$p\to a$"这样的连接,其中,$p\in N^{I_1-I_2}(e)$并且$a\in N^{I_2-I_1}(e)$,则:『e』$^{I_1\cup I_2}=\varnothing$,$\varnothing$表示空集.

定理 4. 对于复合映射$I(I=I_1\cup I_2)$,单树表达式e_1,e分别对应查询树$T(e_1),T(e)$,如果:『e_1』^{I_1}是合法表达式;『e』^{I_1},『e』^{I_2}都是合法表达式,并且$T(e)$中所有边对应的都是主键值关系,则

$$『e_1』^{I_1}\infty『e』^{I_1\cup I_2}=『e_1\infty e』^{I_1}.$$

定理 5. 对于复合映射$I(I=I_1\cup I_2)$,单树表达式e_1,e,e_2分别对应查询树$T(e_1),T(e),T(e_2)$,如果:『e_1』^{I_1}是合法表达式;『e』^{I_1},『e』^{I_2}都是合法表达式,并且$T(e)$中所有边对应的都是主键值关系;『e_2』^{I_2}是合法表达式,则

$$『e_1』^{I_1} \infty 『e』^{I_1 \cup I_2} \infty 『e_2』^{I_2} = 『e_1 \infty e』^{I_1} \infty 『e \infty e_2』^{I_2}.$$

4.2 基于映射的查询划分

根据映射下的查询等价关系,我们提出一种方法,它将全局映射$I(I=I_1 \cup I_2)$下的全局查询e划分为多个局域子查询部分:$e_1,\ldots,e_n$,每个e_i是局域映射下的对局域数据源的子查询,它们之间通过定义 10 中连接算子连接.局域查询所产生的查询结果($『e_i』^{I_j}$)根据子查询之间的连接算子来进行集成,从而生成全局查询结果集.

查询划分函数.$QF(T(e),I_1,I_2)$(e为单树表达式,映射$I=I_1 \cup I_2$,返回Rq为$『e』^I$等价的局域查询表达式):

(1) (判断 $T(e)$查询结果值)

如果e中有I_1,I_2中都没有映射定义的概念和关系,则返回结果∅;如果$T(e)$中存在"$p \rightarrow a$"这样的连接,其中$p \in N^{I_1-I_2}(e) \wedge a \in N^{I_2-I_1}(e) \vee p \in N^{I_2-I_1}(e) \wedge a \in N^{I_1-I_2}(e)$,则返回结果∅;如果$T(e)$中的结点都属于$N^{I_1 \cap I_2}(e)$,那么返回结果$『e』^{I_1} \cup 『e』^{I_2}$;如果在$T(e)$中不存在$p,a$结点都属于$N^{I_1 \cap I_2}(e)$的"$p \rightarrow a$"连接边,则:如果$『e』^{I_1}$是合法表达式,$N^{I_1-I_2}(e)$非空,那么返回结果$『e』^{I_1}$;否则,如果$『e』^{I_2}$是合法表达式,$N^{I_2-I_1}(e)$非空,那么返回结果$『e』^{I_2}$;

(2) (递归调用)不满足上述几种情况时,则:

如果$T(e)$的根结点$n \notin N^{I_1 \cap I_2}(e)$,则从根结点出发,沿向下的路径寻找第一个属于$N^{I_1 \cap I_2}(e)$的结点$p$,在$p$处将$T(e)$分裂为若干个子树:$t_1,t_2,\ldots,t_m$,其中,$t_1$是将$T(e)$剪除掉$p$所有子结点的结果,$t_2,\ldots,t_m$是将根结点为$p$的$T(e)$子树分裂的结果,它们的根结点都是$p$,分别包含若干条$p$结点到$T(e)$叶结点的路径.分别对它们递归调用查询划分函数:

$$Rq=QF(t_1,I_1,I_2) \infty QF(t_2,I_1,I_2) \infty \ldots \infty QF(t_m,I_1,I_2);$$

如果 $T(e)$是一条根结点 $n \in N^{I_1 \cap I_2}(e)$的单路径树,则从根结点 r 出发到达叶结点 l 的路径["$r \rightarrow \ldots \rightarrow l$"]上寻找第一个这样的非根结点 p:p 结点属于 $N^{I_1 \cap I_2}(e)$,p 的父结点和子结点中至少有一个不属于 $N^{I_1 \cap I_2}(e)$,将此单路径$T(e)$分裂为两个子树:T["$r \rightarrow \ldots \rightarrow p$"]和 T["$p \rightarrow \ldots \rightarrow l$"],分别对其递归调用查询划分函数:

$$Rq=QF(T[\text{“}r \rightarrow \ldots \rightarrow p\text{”}],I_1,I_2) \infty QF(T[\text{“}p \rightarrow \ldots \rightarrow l\text{”}],I_1,I_2);$$

如果$T(e)$为根结点$n \in N^{I_1 \cap I_2}(e)$的多路径树,将$T(e)$从根结点$n$处分裂成若干个子树$t_1,t_2,\ldots,t_m$(它们的根结点都为$n$),每个子树包含若干条原$T(e)$中从根结点到叶结点的路径,分别对它们递归调用查询划分函数:

$$Rq=QF(t_1,I_1,I_2) \infty QF(t_2,I_1,I_2) \infty \ldots \infty QF(t_m,I_1,I_2);$$

(3) (结果合并返回)在 Rq 中如果存在下列形式的表达式片断,可以转换为其等价的形式:

$$『e』^{I_i} \infty \varnothing = \varnothing;\ 『e』^{I_i} \cup \varnothing = 『e』^{I_i};\ 『e』^{I_i} \cap \varnothing = \varnothing;\ 『e』^{I_i} / \varnothing = 『e』^{I_i};$$

$$\varnothing / 『e』^{I_i} = \varnothing;\ 『e_1』^{I_i} \infty 『e_2』^{I_i} = 『e_1 \infty e_2』^{I_i};$$

然后返回转换后的结果 Rq'.

查询划分算法.基于查询划分函数QF,设e为查询操作表达式,全局映射$I=I_1 \cup I_2$,其查询划分算法如下:

(1) 对于一个查询表达式e,将所有的σ,π操作外移,构造其查询操作图$D(e)$,$D(e)$由多个树$t_1,\ldots,t_n$构成,子树之间用二元操作算子⊕连接.

(2) 分别对这些树t_i调用查询划分函数$QF(t_i,I_1,I_2)$,获得结果Rq_i,则整体查询结果Rq为这些Rq_i根据子树的连接算子进行连接,然后将σ,π作用到最外层的结果.

(3) 将获得的结果 Rq 根据如下等式进行查询优化:

① $『e_1』^{I_i} \oplus 『e_2』^{I_i} = 『e_1 \oplus e_2』^{I_i}$ (⊕为对元组集合的二元操作);

② 如果$『e_1』^{I_1}$是合法表达式;$T(e)$中的所有结点都属于$N^{I_1 \cap I_2}(e)$,$T(e)$中所有边对应的关系都是主键值关系,同时表达式不再出现其他I_2映射下的局域查询,则

$$『e_1』^{I_1} \infty (『e』^{I_1} \cup 『e』^{I_2}) = 『e_1 \infty e』^{I_1}.$$

(反之,将条件和结论中的所有I_1,I_2互换亦然)

③ 如果$『e_1』^{I_1}$是合法表达式;$T(e)$中所有结点都属于$N^{I_1 \cap I_2}(e)$,$T(e)$中所有边对应的都是主键值关系;$『e_2』^{I_2}$是合法表达式,则

$$『e_1』^{I_1} \infty (『e』^{I_1} \cup 『e』^{I_2}) \infty 『e_2』^{I_2} = 『e_1 \infty e』^{I_1} \infty 『e \infty e_2』^{I_2}.$$

④ 如果n是$T(e_a)$和$T(e_b)$中都出现的结点,其对应的概念为rc,并且rc为主键值关系$kr(kr\in K(o),o$是查询基于的本体)的源概念,kr在I_1和I_2中都有映射定义.设$e=\chi_{vBind}[rv,f_{Rv}(n,kr)]Concept(rc,n)$,并且$\pi$中不包含$n$对应项,那么

$$\pi(\llbracket e_a \rrbracket^{I_1} \oplus \llbracket e_b \rrbracket^{I_2})=\pi(\pi_1 \llbracket e_a \infty e \rrbracket^{I_1} \oplus rv\ \pi_2 \llbracket e_b \infty e \rrbracket^{I_2}).$$

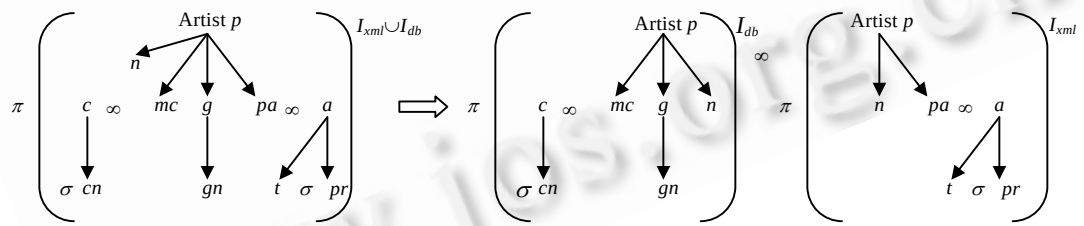
其中,$\oplus\in\{\cup,\cap,/,\infty\}$,$\infty$根据$rv$进行连接,$\pi_1,\pi_2$除去对应表达式中所有$n$对应项.

⑤ 最后,将σ,π操作根据每个局域查询表达式中出现的标签内移,便构成最后的子查询划分的结果.

在以上划分算法中,变换①~变换③的意义在于:尽量在局域数据源层次进行查询并使局域结果集规模尽可能地小,减少合并层次的操作规模;在进行合并集成时,可能某些局域查询结果之间的连接概念对应不到实际可以进行比较的值(比如 Artist 概念),使用变换④可以根据主键值关系定义将其转移到那些可以从局域数据获取并比较的值上.

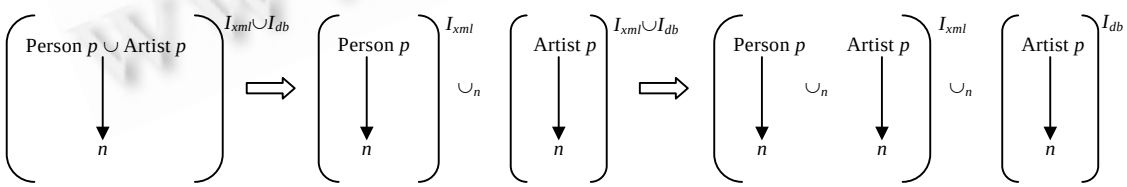
查询划分函数分为 3 个阶段:在判断结果值阶段,根据第 4.1 节中的定理 2 和定理 3 以及其他显然的等价关系来获取局域查询表示结果;在递归调用阶段,根据第 3.2 节中描述的查询操作图等价变换来分裂查询操作树,并根据第 4.1 节中的定理 1 将对 e 全局映射查询分解到 e 的各个分裂子树上;在结果合并阶段进行等价合并.在查询划分算法中,σ,π外移内移操作满足对应的等价条件,而变换①~变换③则是根据第 4.1 节中的定理 1、定理 4、定理 5 进行的等价变换.根据主键值关系的定义,可以证明变换④中的变换等价(证明略).因此可以证明,根据查询划分算法的划分结果进行局域查询和局域查询结果集成,其获得的集成结果就是满足条件的全局查询结果.同时,我们采用一定数量的查询试例,应用本方法进行查询划分,其获得的结果都是符合要求的正确划分结果.

对于图 1 中描述的本体o以及表 2(a)、表 3(a)对应的XML数据源和关系数据库数据源,假设存在表 2(b)和表 3(b)中描述的本体o到XML和关系数据的映射I_{xml}和I_{db},全局映射$I=I_{xml}\cup I_{db}$.图 3 表示Q_1,Q_2的查询操作图在上述查询划分算法操作下的划分过程,表 5 给出了查询划分后Q_1,Q_2的局域查询表达式.其中,Q_1查询表示为对XML和关系数据局域查询结果的连接操作,而Q_2则需要对两个局域查询结果进行并集操作.这是因为在Q_1中涉及到了在另一数据源中没有定义映射的概念和关系(例如作品的价格和作者的流派条件等),因此需要通过连接操作将那些不同时满足上述条件的数据删除;而Q_2查询涉及到的概念和关系在两个局域映射中都有映射定义,因此只需要将两个局域查询结果集进行并集操作即可.



(a) Q_1's query division

(a) Q_1查询的局域划分



(b) Q_2's query division (π is omitted in this figure)

(b) Q_2查询的局域划分(图中省略π操作)

Fig.3 Query divisions of Q_1 and Q_2

图 3 Q_1和Q_2查询的局域划分

Table 5 Local query expressions of Q_1 and Q_2 got by query division

表 5 经过查询划分后的Q_1和Q_2的局域查询

Q_1:	$『\pi[t,n](\sigma[pr>\text{“500”}]\chi_{vBind}[pr,f_{Rv}(a,price)]\chi_{vBind}[t,f_{Rv}(a,title)]Concept(Artifact,a)$ $\infty[a=pa]\chi_{vBind}[pa,f_{Rv}(p,create)]\chi_{vBind}[n,f_{Rv}(p,name)]Concept(Artist,p))』^{I_{xml}}\ \infty$ $『\pi[n,gn](\chi_{vBind}[mc,f_{Rv}(p,nationality)]\chi_{vBind}[gn,f_{Rv}(g,gname)]\chi_{vBind}[g,f_{Rv}(p,belongto)]\chi_{vBind}[n,f_{Rv}(p,name)]$ $Concept(Artist,p)\infty[mc=c]\sigma[cn=\text{“italy”}]\chi_{vBind}[cn,f_{Rv}(c,cname)]Concept(Country,c))』^{I_{db}}$
Q_2:	$『\pi[n](\chi_{vBind}[n,f_{Rv}(p,name)]Concept(Person,p))\cup_n\pi[n](\chi_{vBind}[n,f_{Rv}(p,name)]Concept(Artist,p))』^{I_{xml}}\cup_n$ $『\pi[n](\chi_{vBind}[n,f_{Rv}(p,name)]Concept(Artist,p))』^{I_{db}}$

5 局域查询和结果集成

通过上一节中所描述的全局查询到局域查询的转换,可以获得对每个局域数据源的局域查询表达式.这些局域查询表达式需要转换为局域数据源的本地查询形式.

5.1 XML查询

首先根据I_{xml}中定义的概念、关系与XML数据源的映射关系,将XML局域查询表达式中的概念实例获取函数$Concept(c,L)$和绑定操作χ_{vBind}中的f_{Rv}函数转换为对应的XML文档结点实例获取函数$Get(N(Path),L)$和相对路径结点实例获取函数$f_{getPath}(d_1,vPath)$.Q_1在I_{xml}映射下的XML局域数据查询为

$$\pi[t,n](\sigma[pr>\text{“500”}]\chi_{vBind}[pr,f_{Rv}(a,price)^{I_{xml}}]\chi_{vBind}[t,f_{Rv}(a,title)^{I_{xml}}]Concept(Artifact,a)^{I_{xml}}\infty[a=pa]$$
$$\chi_{vBind}[pa,f_{Rv}(p,create)^{I_{xml}}]\chi_{vBind}[n,f_{Rv}(p,name)^{I_{xml}}]Concept(Artist,p)^{I_{xml}}).$$

根据表 2(b)中的对应关系以及此对应关系定义映射下的函数等价关系,将其转换为Q_{1xml}:

$$\pi[t,n](\sigma[pr>\text{“500”}]\chi_{vBind}[pr,f_{getPath}(a,\text{“/price”})]\chi_{vBind}[t,f_{getPath}(a,\text{“/@title”})]Get(N(\text{“//artifact”}),a)$$
$$\infty[a=pa]\chi_{vBind}[pa,f_{getPath}(p,\text{“/..”})]\chi_{vBind}[n,f_{getPath}(p,\text{“/@name”})]Get(N(\text{“//creator”}),p)).$$

根据这一查询表示的查询操作图,可以将其转换为如下 XQuery 查询(受篇幅所限,这里不再详细介绍转换算法):

```
⟨results⟩   for $a in doc("Source1.xml")//artifact
            where integer($a/price )>500
            return ⟨result⟩ ⟨t⟩$a/@title ⟨/t⟩ ⟨n⟩$a/creator/@name⟨/n⟩ ⟨/ result⟩ ⟨results⟩
```

使用它查询的结果是包含多个“⟨result⟩…⟨/result⟩”片断的 XML 文档,每个“⟨result⟩”XML 文档片断对应一个元组实例,其中,每个子结点对应元组分量,XML 结点标签就是元组分量的标签.

5.2 关系数据查询

与上节中所述的XML查询转换一样,首先要将关系数据局域查询表示中的$Concept(c,L)$替换成对应的数据库元组实例获取函数$GetR(R^{db}(a),L)$,绑定操作χ_{vBind}中的f_{Rv}关系映射函数替换成对应的函数依赖元组获取函数$f_{getF}(L,attrList\rightarrow attrList')$.在替换后,$Q_1$在$I_{db}$映射下关系局域数据查询$Q_{1db}$为

$$\pi[n,gn](\chi_{vBind}[mc,f_{getF}(p,M8)]\chi_{vBind}[gn,f_{getF}(g,M5)]\chi_{vBind}[g,f_{getF}(p,M6)]\chi_{vBind}[n,f_{getF}(p,M2)]GetR(R^{db}(Artist),p)$$
$$\infty[mc=c]\sigma[cn=\text{“italy”}]\chi_{vBind}[cn,f_{getF}(c,M9)]GetR(R^{db}(\pi^{db}_{motherland}R^{db}(Artist)),c)),$$

其中,$M1,\ldots,M9$ 为表 3(b)中所对应的 $attrList\rightarrow attrList'$函数依赖关系.

根据它的查询操作图以及表 3(b)中的映射对应关系,可以将其转换为关系数据查询表示(由于篇幅限制,在此不详细介绍转换算法).Q_{1db}对应的关系数据查询表示为

$$\pi^{db}_{Name,Sname}(\pi^{db}_{motherland}R^{db}(Artist)\infty^{db}R^{db}(Genre)\infty^{db}R^{db}(Artist)\infty^{db}_{motherland}\sigma^{db}_{motherland=\text{“italy”}}\pi^{db}_{motherland}R^{db}(Artist)).$$

其经过优化后对应的 SQL 查询语句为

```
Select Artist.Name as n, Genre.Sname as gn
From Artist, Genre
Where Genre.SID=Artist.sid and Artist.motherland="italy"
```

然后,根据此SQL语句对数据库进行查询,其查询结果为关系元组集合,每个关系元组可以看作是本体概念实例查询结果元组τ,其属性名就是它的标签.

5.3 查询结果集成

在获得各个局域查询结果后,需要根据查询划分给出的局域子查询之间的关系进行合并集成.例如:对于Q_2,只需要将XML查询的姓名结果集合和关系数据查询结果集合进行并集操作"$\cup_n$";对于Q_1,需要根据两个结果集中n标签对应值进行一次连接操作,获取那些同时满足涉及两个数据源条件的Artist实例及相关信息.

XML文档的XQuery局域查询产生的结果为XML文档片断,关系数据库局域查询的结果为关系元组集,它们需要根据局域查询之间的关系进行结果集成.这些集成操作$\oplus(\oplus\in\{\cup,\cap,/,\infty\})$的定义类似于关系数据库中的元组操作定义.比较容易实现的方法是:将对XML局域查询的XML文档结果转换为关系数据形式,然后再与关系数据库的查询结果集进行合并集成.

6 相关工作

OBSERVER[3],SIMS[4]等基于本体集成关系数据的信息集成系统主要注重于领域模型和局域模型对应定义下的查询转换.例如SIMS研究的是如何对全局查询所涉及的全局模型实体进行特殊化、一般化以及概念划分等变换,从而将全局查询变换为只涉及局域模型实体的局域查询.这种方式只能在查询涉及的全局模型实体与局域模型实体都存在对应关系的前提下,才能实现其查询变换.而在全局查询涉及某些局域数据源中没有映射对应关系的概念时,本文设计的方法可以对查询进行划分,从而生成对应的局域查询.

WEESA[5,6]是一个基于本体的XML文档集成系统,它通过构建OWL(Web ontology language)本体到XML模式的对应来为XML文档赋予语义.文献[7−10]中提出了一系列使用本体来语义集成XML文档的方法,其中大部分研究也采用本体概念到XML结点以及本体关系到XML相对路径这种映射集成方式.这些系统和方法主要关注本体中的概念与XML文档的映射方式,而没有详细考察全局查询到局域XML查询的划分转换以及查询结果集成等问题.

Amann[11]提出了一种用于语义集成XML文档的从全局查询获取局域查询的方法,其从局域数据源的角度出发来构建对应于全局查询的局域查询.本文采用的方法则从全局角度出发来对全局查询进行划分,同时在划分过程中采用了优化步骤,相对于Amann提出的方法,它可以获得查询和集成效率更高的局域查询.

另外,上述这些映射集成方法都是针对同一类型的数据源(或者是关系数据、或者是XML文档),而本文采用统一的形式实现了使用本体对局域XML文档和关系数据的集成.

7 结束语

在基于本体的异构信息集成中,用户对本体概念实例的全局查询被分解转换为局域查询,局域查询的结果需要根据分解时所定义的联系进行集成并返回给用户.本文的主要贡献在于:定义了本体到多种异构数据源的映射方式;定义了本体概念实例查询的操作语义表示,根据这一查询语义表示可以进行等价的查询变换;根据查询表达式的操作语义表示以及本体到局域数据源的映射关系,给出了全局查询到局域查询的划分方法,根据此划分所进行分布局域查询和集成的结果就是全局查询所需要的结果.

今后的工作将着重于多种异构信息的集成方法研究.本文只讨论了简单的映射复合情况,如何根据子映射之间的关系来定义它们的复合,以及在这种复杂复合下实现全局到局域的查询划分也是值得研究的内容.

References:

[1] Cluet S, Moerkotte G. Nested queries in object bases. In: Proc. of the Int'l Workshop on Database Programming Languages. New York, 1993. 226−242. http://citeseer.ist.psu.edu/cluet95nested.html

[2] Beeri C, Tzaban Y. SAL: An algebra for semistructured data and XML. In: Proc. of the 4th Int'l Workshop on the Web and Databases. 1999. 37−42. http://citeseer.ist.psu.edu/beeri99sal.html

[3] Mena E, Kashyap V, Sheth A, Illarramendi A. OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. Int'l Journal of Parallel and Distributed Databases, 2000,8(2):232−271.

[4] Arens Y, Knoblock CA, Shen WM. Query reformulation for dynamic information integration. Journal of Intelligent Information Systems (Special Issue on Intelligent Information Integration), 1996,6(2/3):99−130.

[5] Reif G, Jazayeri M, Gall H. Towards semantic Web engineering: WEESA—Mapping XML schema to ontologies. In: Proc. of the World Wide Web Workshop on Application Design, Development and Implementation Issues in the Semantic Web 2004. 2004. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-105/

[6] Reif G, Gall H, Jazayeri M. WEESA—Web engineering for semantic Web applications. In: Proc. of the 14th Int'l World Wide Web Conf. Japan, 2005. 722−729. http://www2005.org/cdrom/contents.html

[7] Cruz I, Xiao H, Hsu F. An ontology-based framework for XML semantic integration. In: Proc. of the 8th Int'l Database Engineering and Applications Symp. Portugal, 2004. 217−226. http://citeseer.ist.psu.edu/cruz04ontologybased.html

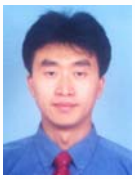
[8] Camillo SD, Heuser CA, Mello RS. Querying heterogeneous XML sources through a conceptual schema. In: Proc. of the 22nd Int'l Conf. on Conceptual Modeling. 2003. 186−199. http://springerlink.metapress.com/content/rm1nv2yg65p9wnr5/?p=60823bbba03a4210bc6045cd83f76876&pi=38

[9] Erdmann M, Studer R. How to structure and access XML documents with ontologies. Data and Knowledge Engineering, 2001,36(3): 317−335.

[10] Amann B, Fundulaki I, Scholl M, Beeri C, Vercoustre AM. Mapping XML fragments to community Web ontologies. In: Proc. of the 4th Int'l Workshop on the Web and Databases (WebDB 2001). 2001. http://citeseer.ist.psu.edu/462796.html

[11] Amann B, Beeri C, Fundulaki I, Scholl M. Querying XML sources using an ontology-based mediator. In: Proc. of the 10th Int'l Conf. on Cooperative Information Systems. 2002. 429−448. http://www.springerlink.com/content/k5j6xt4a0q2nn3rg/

李剑(1976－),男,博士生,主要研究领域为语义 Web,网络信息处理.



宋靖宇(1976－),男,博士生,助理研究员,主要研究领域为网络分布计算,表示层中间件.

钟华(1971－),男,博士,研究员,CCF 高级会员,主要研究领域为网络分布计算,软件工程.