本体模型的逆向获取研究*)

王洪伟[1] 伊 磊[2] 张元元[3]
(同济大学经济与管理学院 上海 200092)[1] (复旦大学数学科学学院 上海 200433)[2]
(复旦大学新闻学院 上海 200433)[3]

摘 要 从遗留信息系统中获取领域信息是创建本体的重要环节。本文以最为常用的关系数据库为对象,分析了如何从遗留系统中识别关系模式的结构信息,然后提出了12条术语转换规则,并根据转换规则从关系模式的结构信息中逆向提取出领域术语及相互关系,最后利用扩展的关系实体图进一步获取关系模式的语义信息,并以此来精炼领域本体。

关键词 本体模型,逆向,关系模式

Study on Ontology Development by Reverse Engineering

WANG Hong-Wei[1] YI Lei[2] ZHANG Yuan-Yuan[2]
(Tongji University, Shanghai 200092)[1] (Fudan University, Shanghai 200433)[2]

Abstract It is a key issue for ontology development to acquire domain information hidden in legacy systems. This paper, taking relational database as example, analyzes how to identify the information about the structure of relational schemes in legacy systems. Then, presents 12 transformation rules, according to which terms and relations between them can be obtained from the relational schemes in reverse way. Finally, uses the Extension Entity-Relationship (EER)diagram to further get semantic information from relational schemes for refining ontology model.

Keywords Ontology model, Reverse engineering, Relational schema, Descriptionlogic

1 前言

本体是指对领域概念化(conceptualization)的一个显式的规格说明[1,2],是对特定领域中的概念及相互间的关系的抽象描述,已被应用在企业建模、异构信息集成、语义Web、信息检索、知识系统等领域。随着本体应用的深入,各种复杂的领域本体有待创建。而现有的领域本体开发方法多数是从正向角度来探索领域本体的创建规律,如Uschold & King模式、Grüninger & Fox模式以及Methontology模式等[3~5]。现实中,信息系统都有后台数据库支持,而数据库结构是系统工程师在需求分析的基础上创建的,因此数据库结构实际上隐含着相应领域的概念模型[6]。因此,从遗留系统的数据库结构中逆向获取本体,显得尤为必要。

2 本体的基本模型

本文利用描述逻辑建立了本体模型[6~8],相关定义如下。

定义1 本体模型是一个7元组,记为$O=\langle T, X, T_D, X_D, A_A, T_C, T_R\rangle$。其中,$T$是术语集;$T_D$是术语定义集;$X$为实例集;$X_D$为实例声明集;$A_A$为属性分配集;$T_C$为术语注释集;$T_R$为术语约束集。

定义2 给定$O=\langle T, X, T_D, X_D, A_A, T_C, T_R\rangle$,语义解释为一个二元组,记为$I=\langle \Delta^I, (\cdot)^I\rangle$。其中,$\Delta^I\neq\varnothing$为$O$的论域,$(\cdot)^I$是解释函数,它将$T$中的每个原子类$C$都映射为$\Delta^I$的一个子集$C^I\subseteq\Delta^I$,将$T$中的每个原子属性$P$都映射为一个二元关系$P^I\subseteq\Delta^I\times\Delta^I$,将$X$中的每一个体$a$映射为$\Delta^I$中的一个元素$a^I\in\Delta^I$。下面给出了一个本体实例。

$O=\langle$ {Person, Woman, Man, Mother, Father, Parent, Child, Son, Daughter, Wife, Husband, Son-in-law, Daughter-in-law, Sex, Grandmother, MotherWithoutSon, hasSex, hasChild, hasMother, hasFather, hasParent, hasSon, hasDaughter, hasHusband, hasWife, hasOffspring, age, name}, {female, male}, {Woman≡Person ∧ ∃hasSex. {female}, Man≡Person ∧ ∃hasSex. {male}, Mother≡Woman ∧ ∃hasChild. Person, Father≡Man ∧ ∃hasChild. Person, Parent≡Father ∨ Mother, Grandmother≡Mother ∧ ∃hasChild. Mother, Son≡Man ∧ Child, Daughter≡Woman ∧ Child, Wife≡Woman ∧ ∃hasHusband. Man, Husband(Man ∧ ∃hasWife. Woman, Son-in-law≡Man ∧ ∃(hasWife × hasParent). Person, Daughter-in-law(Woman ∧ ∃(hasHusband × hasParent). Person, MotherWithoutSon(Mother ∧ ∃hasSon. ⊥, hasChild(hasSon ∨ hasDaughter, hasParent (hasFather ∨ hasMother, hasHusband(hasWife-, hasParent≡hasChild-, hasOffspring(hasChild$^+$}, {Sex(female), Sex(male)}, {Person⊆∃age. xsd:Integer ∧ =1, age ∧ ∃name. xsd:String}, ∅, {∃hasSex. {male}⊆∃hasSex. {female}, ChildPerson }$\rangle$

术语集T由原子术语构成。原子术语分为原子类术语与原子属性术语。如上述代码中,Person、Sex为原子类术语,hasChild、hasWife为原子属性术语。原子类术语有2个特殊的类型:根类"⊤"与空类"⊥",相应的语义解释为$\top^I=\Delta^I$,$\bot^I=\varnothing$。

术语公式是由原子术语与术语构造符相互作用形成的表

*)国家自然科学基金资助(70501024)、教育部人文社会科学项目资助。王洪伟 讲师,博士,研究方向:信息系统集成化管理与智能决策;伊 磊 博士生,研究方向:计算数学。

达式，可分为类术语公式和属性术语公式。如代码中 Man∧∃hasChild. Person 是类术语公式，hasWife×hasParent 是属性术语公式。表 1 构建出 11 种术语构造符，其中 C 与 D 是类术语，P 与 R 是属性术语。

表 1　本体模型的术语构造符及其解释

构造符的名称及语法	构造符的解释
类术语否：$\neg C$	$(\neg C)^I=\Delta^I-C^I$
属性术语否：$\neg P$	$(\neg P)^I=\Delta^I\times\Delta^I\backslash P^I$
术语合取：$C\wedge D, P\wedge R$	$(C\wedge D)^I=C^I\cap D^I, (P\wedge R)^I=P^I\cap R^I$
术语析取：$C\vee D, P\vee R$	$(C\vee D)^I=C^I\cup D^I, (P\vee R)^I=P^I\cup R^I$
属性的积：$P\times R$	$(P\times R)^I=\{(a, c)\in\Delta^I\times\Delta^I \mid \exists b.$ $(a, b)\in P^I\wedge(b, c)\in R^I\}$
属性的逆：P^-	$(P^-)^I=\{(b, a)\in\Delta^I\times\Delta^I(a, b)\in P^I\}$
属性的传递闭包：P^+	$(P^+)^I=\bigcup_{i\geqslant 1}(P^i)^I$
全称量词：$\forall P. C$	$(\forall P. C)^I=\{a\in\Delta^I \mid \forall b. (a, b)\in P^I\rightarrow b\in C^I\}$
全称量词：$\forall P. \{z\}$	$(\forall P. C)^I=\{a\in\Delta^I \mid \forall b. (a, b)\in P^I\rightarrow b=z^I\}$
存在量词：$\exists P. C$	$(\exists P. C)^I=\{a\in\Delta^I \mid \exists b. (a, b)\in P^I\wedge b\in C^I\}$
存在量词：$\exists P. \{z\}$	$(\exists P. C)^I=\{a\in\Delta^I \mid \exists b. (a, b)\in P^I\wedge b=z^I\}$
属性数目下限约束：$\geqslant n, P$	$(\geqslant n, P)^I=\{a\in\Delta^I \mid \{b \mid (a, b)\in P^I\} \mid \geqslant n\}$
属性数目上限约束：$\leqslant n, P$	$(\leqslant n, P)^I=\{a\in\Delta^I \mid \{b \mid (a, b)\in P^I\} \mid \leqslant n\}$

定义 3(术语关系)　给定 $O=\langle T, X, T_D, X_D, A_A, T_C, T_R\rangle$，$D$、$E$ 为 2 个术语，I 为任意一个解释，

(1)如果 $D^I\subseteq E^I$，则称 E 包含 D，记为 $D\hat{o}E$。

(2)如果 $D\hat{o}E$ 和 $E\hat{o}D$，则称 D 与 E 等价，记为 $D\equiv E$。

(3)如果 $D\hat{o}\neg E$ 和 $E\hat{o}\neg D$，则称 D 与 E 非交，记为 $D\dashv E$。

定义 4(术语定义项)　它是一个等价关系 $C\equiv D$，表示 C 用 D 来定义，C 称作前项，D 为后项，其中，C 是原子术语，D 是术语公式。如代码中 Mother≡Woman∧∃hasChild. Person。

术语定义集由一组满足以下条件的术语定义项组成，表示为 $T_D=\{C_1\equiv D_1, C_2\equiv D_2, \cdots, C_n\equiv D_n\}$，其中 $C_i\in T, D_i$ 为术语公式，D_i 中的术语均来自 T。

(1)对任何 i, j $(i\neq j, 1\leqslant i\leqslant n, 1\leqslant j\leqslant n)$，有 $C_i\neq C_j$。

(2)如果存在 $C'_1\equiv D'_1, C'_2\equiv D'_2, \cdots, C'_m\equiv D'_m$，且 C'_i 出现在中 $D'_{i-1}(1<i\leqslant m, m\leqslant n)$，则 C'_1 必不出现在 D'_m 中。

实例集 X 是个体的集合，实例分为特指实例与泛指实例。而实例声明集 X_D 则由类的实例声明与属性的实例声明组成：

(1)类的实例声明 $C(a)$，表示个体 a 属于类 C，如 Sex (male)。

(2)属性的实例声明 $P(a, b)$，表示个体 a, b 之间存在关系 P。

属性分配集将属性术语分配给类术语，记为 $A_A=\{C_1\hat{o}D_{11}\wedge D_{12}\wedge\cdots\wedge D_{1m_1}, C_2\hat{o}D_{21}\wedge D_{22}\wedge\cdots\wedge D_{2m_2}, \cdots, C_n\hat{o}D_{n1}\wedge D_{n2}\wedge\cdots\wedge D_{nm_n}\}$。其中，$C_i\in T$ 为原子类术语，D_i 可以是以下形式：

(1)$\exists P. D$ 或 $\exists P. \{z\}$：表示类术语具有属性 P，并且属性 P 某些值的类型为类 D 或者为个体 z；

(2)$\forall P. D$ 或 $\forall P. \{z\}$：表示类术语具有属性 P，并且属性 P 所有值的类型均为类 D 或者为个体 z；

(3)$\geqslant n, P$ 或 $\leqslant n, P$：表示类术语至少或最多拥有 n 个属性 P。

术语注释集可表示成 $T_c=\{t_c(t_1), \cdots, t_c(t_n)\}$，其中 $t_c(t_i)$ 是对原子术语 $t_i\in T$ 的语义进行自然语言描述。术语注释可借助语义字典来构造，如 WordNet。

术语约束集 T_R 由若干个术语关系(包含关系、等价关系、非交关系)组成，用来限定术语之间的关系。如代码中，∃hasSex. {male}⊣∃hasSex. {female}。

3　领域本体的逆向创建过程

3.1　逆向创建过程的 5 阶段

本文以关系模式为例，将领域本体逆向创建分为 5 个阶段，如图 1 所示。

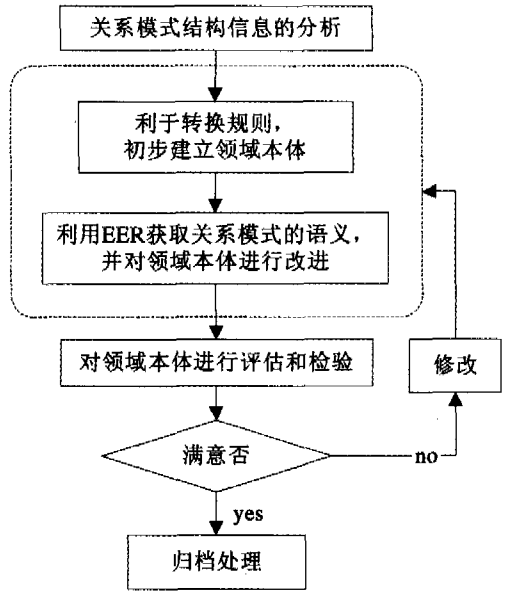


图 1　领域本体逆向抽取过程的流程图

(1)阶段 1：对关系模式的结构信息进行分析。数据库结构包含了一系列用来维护多表之间的联系及一致性的信息，如关系本身、关系的属性、属性的类型、主键、外键、关系之间的包含依赖等信息，这些信息将成为领域本体创建的重要依据，如

(a)关系模式本身是本体中类术语的侯选；关系模式的属性是本体中属性术语的侯选；包含依赖关系是建立本体中术语关系的重要依据；

(b)某些关系模式为了实现多表之间的数据连接、维护数据的一致性而创建了局部键(local key)，通常，一旦脱离了创建该键所在的数据库环境，这些局部键将变得没有意义。因此，在构造领域本体时，可以考虑将这些局部键剔除。

(c)包含依赖关系将来自不同表的信息组合为一个实体。关系模式通常将有关一个实体的信息分布在多张表中，以此避免数据冗余，并有利于数据的增删改。但从用户的角度来看，对同一实体的描述分布在不同的表中是不合理的。因此，在构造领域本体时，应考虑将分布在不同表中的信息重新组织到一个概念中。

(d)通过包含依赖关系来推断出实体间父类-子类关系。

(2)阶段 2：使用转换规则，将收集到的信息转换成领域本体的元素。

(3)阶段 3：根据扩展的实体-关系图(EER)，进一步获取关系模式的语义，以此改进领域本体。

(4)阶段 4：对转换进行评估和检验。检查是否所有的关系都转换成相应的领域本体的元素，领域本体是否在逻辑上保持一致性。

(5)阶段 5：如果领域本体通过评估和校验，则归档处理，

否则，返回阶段(2)、(3)，进行必要的修改。

3.2 相关概念及定义

3.2.1 关系模式的基本概念

(a)关系集 $R=\{r_1,r_2,\cdots,r_m\}$。

(b)属性集 $A_R=\{a_1,a_2,\cdots,a_n\}$。

(c)函数 Γ_a：$R\rightarrow 2^{A_R}$，即 $\Gamma_a(r_i)=\{a_{i_1},a_{i_2},\cdots,a_{i_s}\}$，返回关系 r_i 的属性集。

(d)函数 $\Gamma_k(r_i)=\{a_{i_{k_1}},a_{i_{k_2}},\cdots,a_{i_{k_t}}\}$，返回关系 r_i 的键，其中 $\Gamma_k(r_i)\subseteq\Gamma_a(r_i)$。

(e)集合 D_R 是指关系模式内部的数据类型，如字符型、整数型、布尔型等。

(f)函数 Γ_t：$A_R\rightarrow D_R$，返回关系模式中某个属性的类型。

(g)包含依赖集 I，包含了一组 R 上的包含依赖关系，包含依赖的定义如下。

定义 5(包含依赖) 给定关系集 $R=\{r_1,r_2,\cdots,r_m\}$，R 上的一个包含依赖是这样一个形式 $r_i[A_i]\subseteq r_j[A_j]$，其中，$A_i=\{a_{i_{l_1}},a_{i_{l_2}},\cdots,a_{i_{l_z}}\}\subseteq\Gamma_a(r_i)$，$A_j=\{a_{j_{l_1}},a_{j_{l_2}},\cdots,a_{j_{l_z}}\}\subseteq\Gamma_a(r_j)$，$|A_i|=|A_j|$，$\Gamma_t(a_{i_m})=\Gamma_t(a_{j_m})$，$(m=l_1,l_2,\cdots,l_z)$。$r_i[A_i]$、$r_j[A_j]$分别是关系 r_i 和 r_j 在属性 A_i 和 A_j 上的投影[9]。

(h)集合 I_C 为 I 的传递闭包。

3.2.2 关系与本体模型转换的有关函数与操作

(a)函数 Γ_r：$C\rightarrow R$，表示领域本体中的类术语与哪个关系模式对应。

(b)函数 Γ_t：$T_M\rightarrow T_R$，返回领域本体中某个属性术语的类型。

(c)函数 *set_class*：$R\rightarrow Boolean$，表示是否应该将某个关系模型转换成领域本体中的类术语。

(d)操作 *add_class*(r,c)：将类术语 c 加入术语集 T，并将 c 的自然语言描述加入术语注释集 T_C 中，使得 $\Gamma_r(c)=r$。

(e)操作 *add_one_attr*(c,a,t)：将类型为 t 的属性术语 a 加入术语集 T，并将 a 的自然语言描述加入术语注释集 T_C 中，若关系分配集 R_A 中存在 $c\hat{o}D$，则令 $D:=D\wedge\exists a.t\wedge=1,a$，否则在 R_A 中直接加入 $c\hat{o}\exists a.t\wedge=1,a$，使得 $a\in\Gamma_a(c)$，$\Gamma_t(a)=t$。

(f)操作 *add_some_attr_with_all*(c,a,t)：将类型为 t 的属性术语 a 加入术语集 T，并将 a 的自然语言描述加入到术语注释集 T_C 中，若关系分配集 R_A 中存在 $c\hat{o}D$，则令 $D:=D\wedge\forall a.t\wedge\geqslant 1,a$，否则在 R_A 中加入 $c\hat{o}\forall a.t\wedge\geqslant 1,a$，使得 $a\in\Gamma_a(c)$，$\Gamma_t(a)=t$。

(g)操作 *add_some_attr_with_part*(c,a,t)：将类型为 t 的属性术语 a 加入术语集 T，并将 a 的自然语言描述加入术语注释集 T_C 中，若关系分配集 R_A 中存在 $c\hat{o}D$，则令 $D:=D\wedge\forall a.t$，否则在 R_A 中加入 $c\hat{o}\forall a.t$，使得 $a\in\Gamma_a(c)$，$\Gamma_t(a)=t$。

(h)操作 *add_sub_class*(c_1,c_2)：建立类术语 c_1 和 c_2 之间的包含关系，即 $c_2\hat{o}c_1$，加入到术语约束集 T_R 中。

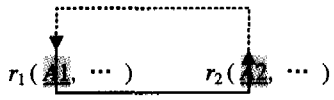
(i)操作 *merge*(r_1,r_2,r)：将关系 r_1 和 r_2 合并为一个新关系 r，使得 $\Gamma_a(r)=\Gamma_a(r_1)\cup\Gamma_a(r_2)$，$\Gamma_k(r)=\Gamma_k(r_1)\cup\Gamma_k(r_2)$，同时 r 将保留 r_1 和 r_2 原有的依赖关系，并令 $R:=R-\{r_1,r_2\}\cup\{r\}$。

(j)操作 *inverse*(a_1,a_2)：将属性术语 a_1,a_2 定义为逆关系，加入到术语约束集 T_R 中。

(k)操作 *add_class_from_attrs*(c,A)：将属性集 A 转换为类术语 c，并将 A 中的属性逐一分配给 c，即对 $\forall a\in A$，有 *add_one_attr*$(c,a',\Gamma_t(a)),a'\in\Gamma_a(c),\Gamma_t(a')=\Gamma_t(a)$。

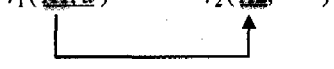
3.3 本体类术语的生成规则

规则 1 给定 $r_1\in R,r_2\in R$，其中，$A_1=\Gamma_k(r_1)$，$A_2=\Gamma_k(r_2)$，如果有 $r_1[A_1]\subseteq r_2[A_2]$，则 *merge*$(r_1,r_2,r)$，图例如下所示(为了描述清晰，图例中带有底纹的字符表示参与包含依赖关系的属性，下划线表示主键。)。规则(1)说明：如果关系 r_1 与关系 r_2 关于键 A_1 和键 A_2 存在包含依赖，并且 r_2 与 r_1 关于 A_2 和 A_1 也存在包含依赖，则说明 r_1 与 r_2 所代表的实体存在一对一的对应关系，r_1 与 r_2 是在说明同一个概念，因此将 r_1 与 r_2 合并为一个关系模式。另外，如果关系 r_1 与关系 r_2 关于键 A_1 和键 A_2 存在包含依赖，而 r_2 与 r_1 关于 A_2 和 A_1 不存在包含依赖，r_1 与 r_2 也可能是在说明同一个概念，此时，需要人工判断。

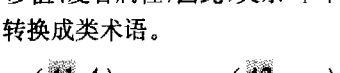


规则 2 给定 $r_1\in R$，其中，$\Gamma_k(r_1)=\Gamma_a(r_1)$，$A_{11}\cap A_{12}=\varnothing$，$A_{11}\cup A_{12}=\Gamma_a(r_1)$，如果存在 $r_2\in R,r_3\in R,A_2=\Gamma_k(r_2)$，$A_3=\Gamma_k(r_3)$，使得 $r_1[A_{11}]\subseteq r_2[A_2]$，$r_1[A_{12}]\subseteq r_3[A_3]$，则令 *set_class*$(r_2)=.T.$，*set_class*$(r_3)=.T.$；如果 *set_class*$(r_1)$已被赋值，则保留原值，否则令 *set_class*$(r_1)=.F.$，图例如下所示。规则(2)说明：这种情况说明关系 r_2 和 r_3 所代表的实体通过关系 r_1 建立起多对多的对应关系，因此，关系 r_1 不必转换成类术语，而 r_2 和 r_3 可以转换成类术语。

规则 3 给定 $r_1\in R$ 和 $A_1\subset\Gamma_a(r_1)$，如果 $\Gamma_a(r_1)=\Gamma_k(r_1)$，$|\Gamma_a(r_1)|=|A_1|+1$，并且存在 $r_2\in R,A_2=\Gamma_k(r_2)$，使得 $r_1[A_1]\subseteq r_2[A_2]$，则令 *set_class*$(r_2)=.T.$；如果 *set_class*$(r_1)$已被赋值，则保留原值，否则令 *set_class*$(r_1)=.F.$，图例如下所示。规则(3)说明：这种情况说明关系 r_1 中的属性 a 是关系 r_2 所代表的实体的一个多值属性，因此，关系 r_1 不必转换成类术语，而 r_2 可以转换成类术语。



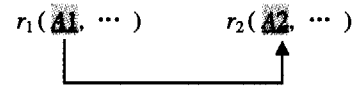
规则 4 给定 $r_1\in R$ 和 $A_1\subset\Gamma_a(r_1)$，如果 $\Gamma_a(r_1)=\Gamma_k(r_1)$，并且存在 $r_2\in R,A_2=\Gamma_k(r_2)$，使得 $r_1[A_1]\subseteq r_2[A_2]$，则令 *set_class*$(r_2)=.T.$；如果 *set_class*$(r_1)$已被赋值，则保留原值，否则令 *set_class*$(r_1)=.F.$，图例如下所示。规则(4)说明：这种情况说明关系 r_1 中的属性集 A 是关系 r_2 所代表的实体的一个多值、复合属性，因此，关系 r_1 不必转换成类术语，而 r_2 可以转换成类术语。



规则 5 给定 $r\in R$，如果 T 中不存在 c，使得 $r=\Gamma_r(c)$，则 *add_class*(r,c)。规则(5)说明：将关系 r 转换成本体的类术语 c，并加入到本体术语集 T。

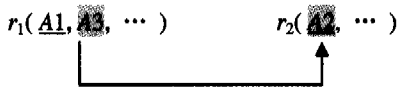
规则 6 给定 $r_1\in R,r_2\in R$，其中，$A_1=\Gamma_k(r_1)$，$A_2=\Gamma_k(r_2)$，如果有 $r_1[A_1]\subseteq r_2[A_2]$，$r_2[A_2]\not\subset r_1[A_1]$，并且 T 中

存在 c_1,c_2,使得 $r_1=\Gamma_r(c_1)$,$r_2=\Gamma_r(c_2)$,则 *add_sub_class*(c_1, c_2),图例如下所示。规则(6)说明:如果关系 r_1 和 r_2 关于键 A_1 和键 A_2 存在包含依赖关系,并且,反之不成立,那么关系 r_1 和 r_2 所表示的实体 c_1 和 c_2 之间存在继承关系,并加入到本体中术语约束集 T_R。

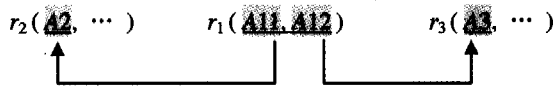


规则 7 给定 $r\in R$ 和 $a\in\Gamma_a(r)$,并且 T 中存在 c,使得 $r=\Gamma_r(c)$,则 *add_one_attr*(c,a',$\Gamma_t(a)$)。规则(7)说明:将关系 r 中的属性 a 转换成属性术语 a',并加入到本体中术语集 T 中,然后将 a' 的语义用自然语言描述并加入到术语注释集 T_C 中,最后将属性术语 a' 分配给相应的类术语 c,并加入到关系分配集 R_A 中。

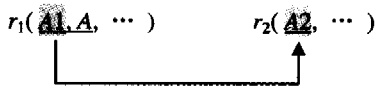
规则 8 给定 $r_1\in R$,$r_2\in R$,其中,$A_1=\Gamma_k(r_1)$,$A_2=\Gamma_k(r_2)$,$A_3\subseteq\Gamma_a(r_1)$,如果 $A_3\cap\Gamma_k(r_1)=\varnothing$,$r_1[A_3]\subseteq r_2[A_2]$成立,并且 T 中存在 c_1, c_2,使得 $r_1=\Gamma_r(c_1)$,$r_2=\Gamma_r(c_2)$,则 *add_one_attr*(c_1, a_{c_2}, c_2),如果 r_2 在 r_1 中是全参与,则 *add_some_attr_with_all*(c_2,a_{c_1},c_1),如果 r_2 在 r_1 中是部分参与,则 *add_some_attr_with_part*(c_2,a_{c_1},c_1)。然后 *inverse*(a_{c_1},a_{c_2}),图例如下所示。规则(8)说明:如果关系 r_2 的键 A_2 是关系 r_1 的外键,则关系 r_1 与 r_2 之间存在一对多的关系,首先建立 2 个属性术语:a_{c_1} 和 a_{c_2},并加入到本体中术语集 T 中,然后将属性术语 a_{c_1} 分配给的类术语 c_2,将属性术语 a_{c_2} 分配给的类术语 c_1,并将 a_{c_1} 和 a_{c_2} 设为互逆关系(或属性),然后加入到术语约束集 T_R 中。



规则 9 给定 $r_1\in R$,$r_2\in R$,$r_3\in R$,$A_2=\Gamma_k(r_2)$,$A_3=\Gamma_k(r_3)$,$\Gamma_k(r_1)=\Gamma_a(r_1)$,$A_{11}\cap A_{12}=\varnothing$,$A_{11}\cup A_{12}=\Gamma_a(r_1)$,$r_1[A_{11}]\subseteq r_2[A_2]$,$r_1[A_{12}]\subseteq r_3[A_3]$。如果 T 中存在 c_2、c_3,使得 $r_2=\Gamma_r(c_2)$,$r_3=\Gamma_r(c_3)$,不存在 c_1,使得 $r_1=\Gamma_r(c_1)$,则如果 r_3 在 r_2 中是全参与,则 *add_some_attr_with_all*(c_3, a_{c_2}, c_2),否则,*add_some_attr_with_part*(c_3,a_{c_2},c_2);如果在 r_2 中 r_3 是全参与,*add_some_attr_with_all*(c_2,a_{c_3},c_3),否则 *add_some_attr_with_part*(c_2,a_{c_3},c_3)。然后 *inverse*(a_{c_2},a_{c_3}),图例如下所示。规则(9)说明:如果关系 r_1 的键是其属性集 $\Gamma_a(r_1)$本身,而且可以分解成 2 个外键,分别参照关系 r_2 和 r_3,则 r_2 与 r_3 之间存在多对多的关系,分别建立 2 个属性术语:a_{c_2} 和 a_{c_3},并加入到本体中术语集 T 中,然后将属性术语 a_{c_2} 分配给的类术语 c_3,将属性术语 a_{c_3} 分配给的类术语 c_2。并将 a_{c_2} 和 a_{c_3} 设为逆关系,然后加入到术语约束集 T_R 中。

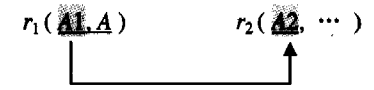


规则 10 给定 $r_1\in R$,$r_2\in R$,并且 $A_1\subset\Gamma_k(r_1)$,$A_2=\Gamma_k(r_2)$,$r_1[A_1]\subseteq r_2[A_2]$。如果 T 中存在 c_1, c_2,使得 $r_1=\Gamma_r(c_1)$,$r_2=\Gamma_r(c_2)$,则 *add_one_attr*(c_1,a_{c_2},c_2),如果 r_2 在 r_1 中是全参与,则 *add_some_attr_with_all*(c_2,a_{c_1},c_1),如果是部分参与,则 *add_some_attr_with_part*(c_2,a_{c_1},c_1)。然后 *inverse*(a_{c_1},a_{c_2}),图例如下所示。规则(10)说明:如果关系 r_1 的键的真子集 A_1 参照关系 r_2 的键 A_2,则关系 r_1 与 r_2 之间存在一对多的关系,分别建立 2 个属性术语:a_{c_1} 和 a_{c_2},并加入到本体中术语集 T 中,然后将属性术语 a_{c_1} 分配给的类术语 c_2,将属性术语 a_{c_2} 分配给的类术语 c_1,并将 a_{c_1} 和 a_{c_2} 设为逆属性,然后加入到术语约束集 T_R 中。



规则 11 给定 $r_1\in R$,$r_2\in R$,并且 $\Gamma_a(r_1)=\Gamma_k(r_1)$,$A_1\subset\Gamma_a(r_1)$,$A_2=\Gamma_k(r_2)$,$|\Gamma_a(r_1)|=|A_1|+1$,$\Gamma_a(r_1)=A_1\cup\{a\}$,$r_1[A_1]\subseteq r_2[A_2]$。如果 T 中不存在 c_1,使得 $r_1=\Gamma_r(c_1)$,存在 c_2,使得 $r_2=\Gamma_r(c_2)$,则如果 r_2 在 r_1 中是全参与,则 *add_some_attr_with_all*(c_2,a_{r_1},$\Gamma_t(a)$),如果 r_2 在 r_1 中是部分参与,则 *add_some_attr_with_part*(c_2,a_{r_1},$\Gamma_t(a)$)。图例如下所示。规则(11)说明:这种情况说明关系 r_1 中的属性 a 是关系 r_2 所代表的实体的一个多值属性,因此,关系 r_1 不必转换成类术语,而是将属性 a 转换成属性术语,加入到本体中术语集 T 中,然后并分配给 c_2。

规则 12 给定 $r_1\in R$,$r_2\in R$,并且 $\Gamma_a(r_1)=\Gamma_k(r_1)$,$A_1\subset\Gamma_a(r_1)$,$A_2=\Gamma_k(r_2)$,如果 $r_1[A_1]\subseteq r_2[A_2]$,并且 T 中不存在 c_1,使得 $r_1=\Gamma_r(c_1)$,存在 c_2,使得 $r_2=\Gamma_r(c_2)$,则首先 *add_class_from_attrs*(c_3,A),然后,对 $\forall a\in A$,*add_one_attr*(c_3,a',$\Gamma_t(a)$),最后,如果 r_2 在 r_1 中是全参与,则 *add_some_attr_with_all*(c_2,a_{c_3},c_3),如果 r_2 在 r_1 中是部分参与,则 *add_some_attr_with_part*(c_2,a_{c_3},c_3)。图例如下所示。规则(12)说明:这种情况说明关系 r_1 中的属性集 A 是关系 r_2 所代表的实体的一个多值、复合属性,因此,关系 r_1 不必转换成类术语。而是将属性集 A 转换成类术语 c_3,然后建立 c_2 与 c_3 之间的一对多关系。



3.4 关系模式与领域本体转换的流程

图 2 描述了从关系模式向领域本体的转换流程,其基本思路是:首先,将描述同一事物或概念的若干张表合并起来;然后,判断哪些表可以转换成类术语,哪些表不必转换成类术语;接下来,将能够转换成类术语的表转换为类术语,并建立类的层次结构;最后,建立属性术语,并分配给相应的类术语。

经过上述转换,将收集到的信息转换成领域本体的元素。然后,根据扩展的实体—联系图(EER),进一步获取关系模式的语义,以此改进领域本体。接着,对转换进行评估和检验,检查是否所有的关系都转换成相应的领域本体的元素,领域本体是否在逻辑上保持一致性。最后,如果领域本体通过评估和校验,则归档处理,否则对前面的步骤进行必要的修改。如何从 EER 中获取语义,如何进行本体检验,请参考文[7],此不赘述。

结束语 逆向获取的目的就是能够根据遗留系统中数据库的结构信息来构造领域本体。逆向获取方法侧重于对概念术语及其内部结构的识别,能够减少对领域专家的依赖,是对传统的正向创建方法的补充,对本体工程师具有一定的借鉴价值。在"CRM 中客户本体的建立研究"中,课题组对多个 CRM 系统的数据模式进行分析,从中获得大量关于客户知识

的术语描述，为客户本体的构建提供了很大的帮助[8]。

参 考 文 献

1 Gruber T R. Ontolingua: A translation approach to portable ontology specification [J]. Knowledge Acquisition, 1993, 5(2): 199～220

2 Guarino N. Formal ontology and information systems [A]. In: Proceedings of FOIS'98, Trento, Italy [C]. Amsterdam: IOS Press, 1998. 3～15

3 Uschold M, King M. Towards a methodology for building ontologies [A]. In: Proc. of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence [C], Montreal, Canada, August 1995

4 Grüninger M, Fox M S. The logic of enterprise modeling. Brown J, O'Sullivan D, eds. Reengineering the Enterprise, Chapman & Hall, 1995. 83～98

5 Gómez-Pérez A. Knowledge Sharing and Reuse [M]. In: Liebowitz J, ed. Handbook of Applied Expert Systems, CRC Press, 1998

6 Vipul K. Design and creation of ontologies for environmental information retrieval [C]. In: Proc. of the 12th International Conf. on Knowledge Acquisition, Modeling and Management, Banff, Canada, October 1999

7 王洪伟，蒋馥，吴家春. Extended ontology model and ontology checking based on description logics [J]. 上海交通大学学报(自然科学版)，2004，E-9(1)：34～41

8 王洪伟，蒋馥，吴家春. Study on formal ontology model: constructing customer ontology in CRM context [C]. In: Proc. of 9th Americas Conference on Information System, Tampa, Florida, USA, August 2003

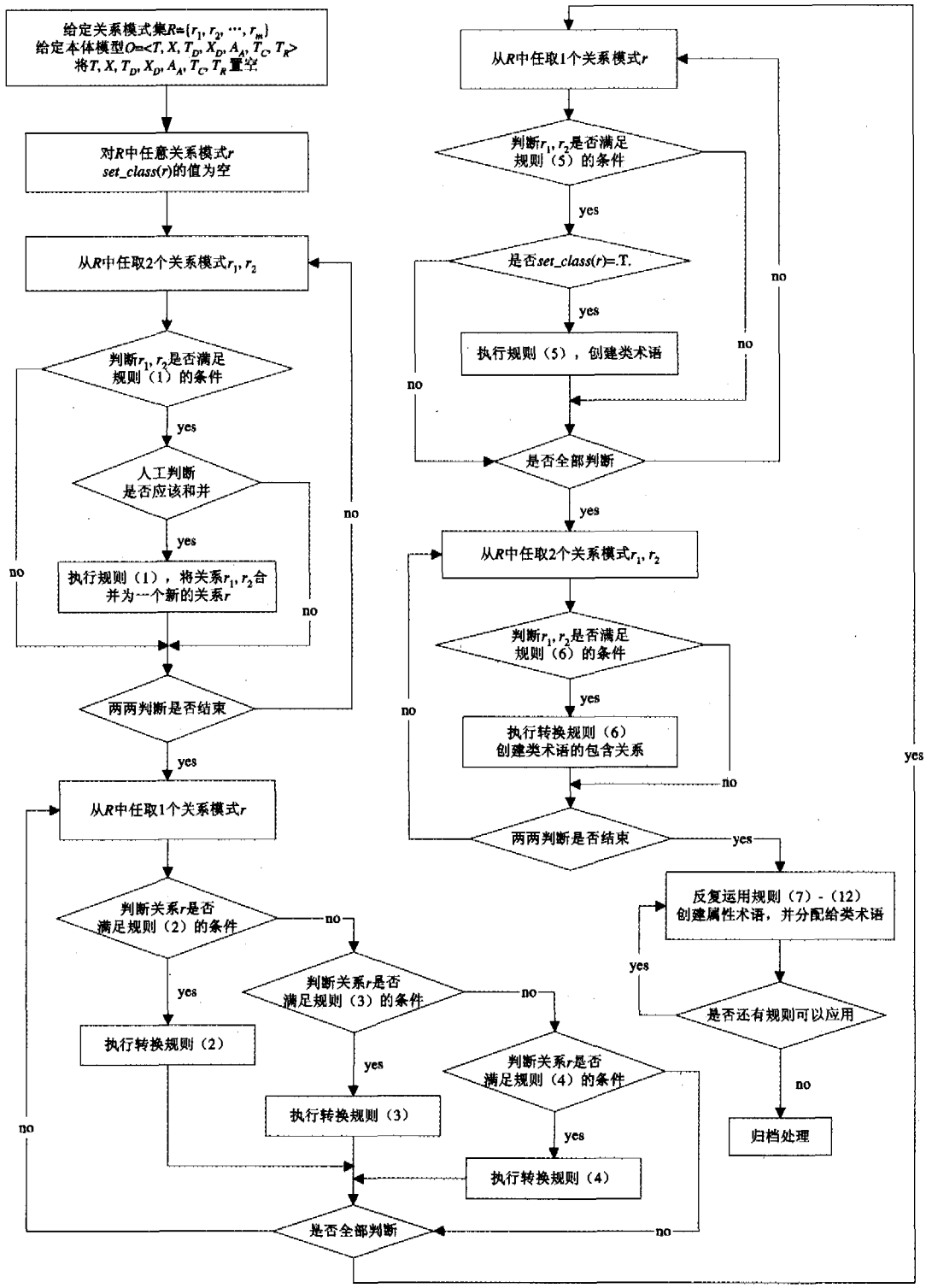


图 2　关系模式与领域本体转换的流程