演绎面向对象数据库语言中的方法研究*

王 浩 高 隽 张莫成

(合肥工业大学人工智能应用研究室 合肥 230009)

摘要 方法和方法继承是面向对象的核心概念.本文介绍了约束演绎对象语言 CDOL(constraint deductive object language)中方法的概念及定义,方法是作为规则定义的,它既可作为对象的接口,又可用来实现计算的抽象与共享,方法的继承和多态是通过类型机制和合一机制来实现的.

关键词 演绎面向对象数据库,演绎查询语言,方法,方法继承.

中图法分类号 TP311.13

演绎面向对象数据库 DOOD(deductive object-oriented database)研究是当前数据库领域的一个前沿课题,它将面向对象数据库(OODB)和演绎数据库(DDB)这两个方面的研究成果结合起来,在统一的理论框架下,将面向对象的核心概念结合到演绎方法中去.DOOD 的研究开始于对复杂对象增加演绎推理的研究,它从传统的 DDB 理论出发,在 DDB 的关系数据模型中增加对集合、元组和表等数据项的支持,进而明确结合了对象标识(Oid)、继承等面向对象概念.由于传统 DDB 的查询语言是基于 Horn 子句的,因此没有与过程语言中方法相对应的概念,也没有对多态的支持.因而所提出的面向对象演绎查询语言,很多都忽略了方法及方法继承.M. Kifer 等提出的 F-LOGIC[1]使用了非基 ID 项(Noground ID-term)作为方法定义的标志.在 F-LOGIC 中,一个子类可以是超类的一个实例,类可以作为对象来使用,这种概念上的重叠,就混淆了一个类定义的方法是作为其子类上的方法还是此类中单个对象的方法之间的区别,在对象和类之间定义方法也有一些限制,例如,关系模型中象连接这样的操作也无直观的定义.IQL[2]和 ILOG[3]提出了对象标识符创建的概念,但也仅讨论了数据继承,没有清晰的方法概念,在文献[4]中,Abiteboul 提出了方法是满足某种类型限制的函数,并通过示例讨论了方法继承和重载,但并未给出理论保证.另一方面,HILOG[5]讨论了具有一阶语义的高阶语法,虽然没有直接讨论 OO 特征,但已有了定义方法的机制,并且参数化多态可以容易地实现.不过,HILOG 没有说明如何实现继承.

本文首先简要介绍我们在文献[6]基础上提出的一个约束演绎的对象语言 CDOL(constraint deductive object language),然后讨论 CDOL 中方法的概念、定义以及方法的继承性.

1 CDOL 简介

CDOL 是一个基于规则的具有约束处理能力的对象查询语言,其数据模型集成了面向对象和面向值两方面的特征.[7]CDOL 是一个类型化语言,语言中的常量和变量都是带有类型的.其类型定义如下:

(1) 基本类型(如 Integer, Real, String)是类型.

(2) 类 C 为类型,这里 C 为一类名.

(3) $[A_1:\tau_1,\ldots,A_n:\tau_n]$是一个类型(元组类型,简记为$[\tau_1,\ldots,\tau_n]$),这里 $n\geqslant 0$,A_i 为属性名,τ_i 为类型.

(4) $\{\tau\}$是一个类型(集合类型),这里 τ 为类型.

设 σ 为一个由类到类型的映射.

在 CDOL 中定义一个特殊的类 ALL,为所有其它类的根,并定义类层次的一个偏序关系 is-a.设 $C1$,$C2$ 为类名,且 $C1$ is-a $C2$,则根据下面定义,有 $C1\leqslant C2$.

类型 τ_1 是类型 τ_2 的子类型,记为 $\tau_1\leqslant\tau_2$,当且仅当下列条件成立:

---

* 本文研究得到国家教委博上点专项基金资助.作者王浩,1962 年生,博士,讲师,主要研究领域为面向对象方法,智能 CAD,智能数据库.高隽,1963 年生,讲师,主要研究领域为模式识别,人工智能.张莫成,1926 年生,教授,博士导师,主要研究领域为人工智能应用,知识工程,智能数据库.

本文通讯联系人:王浩,合肥 230009,合肥工业大学计算机与信息系

本文 1997-01-24 收到原稿,1997-05-04 收到修改稿

(1) τ_1 为基类型或类名,且 $\tau_1 - \tau_2$;

(2) τ_1 为基类型或类名,且 $\sigma(\tau_1) \leqslant \tau_2$;

(3) τ_1, τ_2 为类名,且 $\sigma(\tau_1) \leqslant \sigma(\tau_2)$;

(4) τ_1 为元组类型$[A_i:\tau_{1,i}](1 \leqslant i \leqslant p)$,$\tau_2$ 为元组类型$[A_k:\tau_{2,k}](1 \leqslant k \leqslant q)$,$q \leqslant p$,且$\forall\ k \exists\ i: A_i = A_k, \tau_{1,i} \leqslant \tau_{2,k}$;

(5) τ_1 为集合类型$\{\tau'_1\}$,τ_2 为集合类型$\{\tau'_2\}$,且 $\tau'_1 \leqslant \tau'_2$.

在CDOL中引入id-项作为文字中的标识符及方法的定义,其定义如下:

(1) 常量Oid是类型C的一个id-项,C为一类名.

(2) 若$t_1, \ldots, t_n$分别为类型$\tau_1, \ldots, \tau_n$对应的项,f是类型为$\tau_1 \times \ldots \times \tau_n \to \tau$的函数符,$\tau$是一个类$C$对应的类型,则$f(t_1, \ldots, t_n)$是类型为$\tau$的id-项,称为函数id-项.id-项的值可由特殊的操作符$*$获取.

CDOL中的项、原子公式、文字可定义如下:

项:(1) 具有指定类型的常量、变量和id-项是项.

(2) 若t为类型τ的项,则$*t$为类型τ'的项,τ'是将τ中的类名换成其相应类型的结果,我们称之为展开项.

(3) 若$t_1, \ldots, t_n$分别为类型$\tau_1, \ldots, \tau_n$对应的项,则$[A_1:t_1, \ldots, A_n:t_n]$是类型为$[\tau_1, \ldots, \tau_n]$的项,称为元组项.

(4) 若$t_1, \ldots, t_n$类型τ项,则$\{t_1, \ldots, t_n\}$为类型$\{\tau\}$的项,称为集合项.

{},[]和$*$称为值构造符,不含变量的项即是基项.

原子公式:(1) id-项是原子公式.

(2) 如果t是一个id-项,具有类型$\tau = [\tau_1, \ldots, \tau_n]$(或$\tau = \{[\tau_1, \ldots, \tau_n]\}$),且$t_1, \ldots, t_n$分别为类型$\tau_1, \ldots, \tau_n$的项,则$t[t_1, \ldots, t_n]$(或$t(t_1, \ldots, t_n)$)是原子公式,$t$称为此原子公式的谓词id-项,$t_1, \ldots, t_n$称为参数项.

文字:一个原子公式是文字,称为正文字.若L为一正文字,则$\neg L$为负文字.

CDOL具有约束表达和处理能力,约束是数学上的一个关系,形式为:$R(t_1, \ldots, t_n), n > 0$

这里R是一个表示约束的关系,$t_1, \ldots, t_n$是参与约束的项.

约束系统是一个有限的约束集合:　　$R_1, \ldots, R_n, \quad n \geqslant 0$

其中每个R_i都是一个约束,一个约束系统被满足当且仅当其中每一个约束都被满足且与每一约束出现的次序无关.

CDOL系统中规则具有如下形式:　　$L: -L_1, \ldots, L_m, \{R_1, \ldots, R_n\}$

这里L为正文字,称为规则头;$L_1, \ldots, L_m$为文字集,称为规则体;$R_1, \ldots, R_n$称为该规则的约束系统.

当n和m为0时,规则转化为事实.规则头为空时,表示一个查询.CDOL程序是由一个有限的规则集组成.本文下面的讨论不涉及约束求解问题,有关这方面的问题及CDOL的语义理论见文献[8].

2 方法的概念与定义

2.1 规则作为方法

在CDOL中,对象的结构部分由标识符Oid和一个复杂的状态组成,根据对象有无封装,有两种不同的方式来存取对象状态:(1) 如果对象是封装的,则定义在此对象类上的一组方法作为其接口.(2) 如果对象不需要封装,则可直接存取其状态,如使用$*$操作符.

方法有时主要用来实现计算的抽象和共享.在CDOL中,一条规则即是一个方法,它用来获取对象的属性值或从对象中导出新的信息.在类层次结构中,方法既可以定义在某个(些)特定的对象上,也可以定义在几个类/类型之上.我们将定义在几个类/类型之上的且具有多态性质(包括参数多态和继承多态)的方法称为通用方法(Generic Method),而把定义在特定对象上的方法称为具体方法(Concrete Method).

例1:在Datalog中,下面的程序用来计算关系r传递闭包,并将结果存放在关系q中.

$$q(X,Y): -r(X,Y)$$

$$q(X,Y): -r(X,Z), q(Z,Y)$$

查询$q(X,Y)$?可看成是传递给数据库的一个消息,而响应此消息的方法($q(X,Y)$)则由合一算法选中,关系r的传递闭包即是方法返回的结果.在Datalog中,规则不是通用方法.对上面的例子,方法只是定义在关系r上,如果有另一个关系s,并期望求得它的传递闭包,就必须另外定义一个方法(规则).即在Datalog中谓词名(关系名)是一个常量,不能实现任何形式的共享.

在CDOL中,规则左边其谓词项是常量Oid's的规则就是具体方法.通过在id-项上使用函数符(即函数id-项),我们可以定义通用方法.规则头部的函数id-项中的变量的作用是传递信息到规则体中,变量的类型指定了此方法输入

参数的类型.

2.2 通用方法

在CDOL中,每个函数符均具有相应的类型和元数,函数 f 的类型的形式定义如下:

$$f:\tau_1\times\tau_2\times\ldots\times\tau_n\rightarrow\tau$$

这里 n 是 f 的元数,$\tau_1,\tau_2,\ldots,\tau_n$ 是其参数类型,τ 是返回的对象类型.特别地,一个具有类型 τ 的对象标识符 o 可看成是一个常量函数(0元函数),即 $o:\rightarrow\tau$.

因此,当一条规则以某一函数id-项开头时,这条规则即定义了一个方法.而方法收到一条消息时(即方法调用),就计算出一个值并赋给结果对象.结果对象的标识符由规则头部的函数id-项表示,其中变量已被实例化了.

定义1. 通用方法是一个规则(或规则集),这个(些)规则左边以具有类型为 $:\tau_1\times\tau_2\times\ldots\times\tau_n\rightarrow\tau$ 的函数id-项 $f(t_1,t_2,\ldots,t_n)$ 开始.规则体是方法的实现,函数id-项 $f(t_1,t_2,\ldots,t_n)$ 是此方法的接口,$\tau_1\times\tau_2\times\ldots\times\tau_n\rightarrow\tau$ 称为方法的标记(Signature).

由于谓词可看成是一个常量函数id-项,因而CDOL中的所有规则都是方法.

例2:我们将例1程序用CDOL写成一个通用方法,

$$trans\text{-}closure(R)(X,Y):-R(X,Y)$$

$$trans\text{-}closure(R)(X,Y):-R(X,Z),trans\text{-}closure(R)(Z,Y)$$

其标记为

$$trans\text{-}closure:\tau\rightarrow\tau$$

这里 τ 是一个类名;R 是类型为 τ 的变量;$trans\text{-}closure(R)$ 是一个谓词项,表示此方法的接口.如果消息 $trans\text{-}closure(r)$ 传递到数据库,方法 $trans\text{-}closure(R)$ 将对此作出响应.结果对象的Oid即是 $trans\text{-}closure(r)$,其值就是此方法计算所得的 r 的传递闭包.

标记为 $f:\tau\rightarrow\tau'$ 的方法即是定义在类 τ 上的方法,对每个类可用此方式定义一个方法集.为达到封装的目的,对一个对象状态的存取规定必须通过使用定义在其类上的方法.另一方面,方法也可实现计算抽象和程序的模块性.由于数据库中并非所有对象均需要封装,有些类只是一些具有共同性质的对象的组合,在这种情况下,方法实现的是代码共享.

在许多情况下,方法需要检索几个对象(不属于同一个类),在F-Logic中,必须在其中的某个类上定义方法并以其余的类作为参数(在F-Logic中方法是作为类的属性定义的),因而显得不自然.[9]而CDOL中的具有 n 元的方法就可解决此问题.

3 方法继承

设有一方法,其接口为 $f(t_1,t_2,\ldots,t_n)$,标记为 $\tau_1\times\tau_2\times\ldots\times\tau_n\rightarrow\tau$,其中 τ 为类名.现有一消息(查询)$f(t'_1,t'_2,\ldots t'_n)$ 送到数据库,其中 $t'_1,t'_2,\ldots,t'_n$ 分别为类型 $\tau'_1,\tau'_2,\ldots,\tau'_n$ 的项,且 $f(t'_1,t'_2,\ldots t'_n)$ 的类型为 τ',则此消息是调用具有标记为 $:\tau'_1\times\tau'_2\times\ldots\times\tau'_n\rightarrow\tau'$ 的方法.方法 $f(t_1,t_2,\ldots,t_n)$ 是否会对此消息作出反应?

在CDOL中,我们规定当满足下列类型要求时,此消息即可发送给方法 $f(t_1,t_2,\ldots,t_n)$.

(1) $\tau'_1\leqslant\tau_1,\ldots,\tau'_n\leqslant\tau_n$;

(2) $\tau\leqslant\tau'$.

即消息的参数类型必须是方法参数类型的子类型,而方法的结果类型则是消息的子类型.从直观上讲,第1个条件表明,定义在超类型上的方法可以在子类型上使用;第2个条件表明,结果对象只能是超类型的对象.

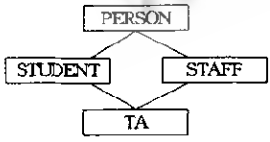

图1 类层次

对于定义在一个类 C 上的方法 $f(t)$(即标记为 $\tau\rightarrow\tau'$),对于其子类 C',由于有 $\sigma(C')\leqslant\sigma(C)$,则 $f(t)$ 可应用到其子类对象上,如对图1所示的类层次,定义在PERSON类上的方法可用于STUDENT类,而如果一方法输出类型为STUDENT类,则结果对象可看成是PERSON类的一个实例对象,而不能作为TA类的实例对象.

例3:对于图1定义的类层次,设类PERSON上定义了一个计算年龄的方法:

$$age(P:PERSON)(N:int):-P(Name:S,Birthyear:Y),N=Curr\_year-Y$$

其中假设Curr_year为系统变量表示当前年份,则对于STUDENT类中的一个实例wang,由于其类型是PERSON类型的子类型,我们可以用此方法求得wang的年龄,设wang对象值为[Name:"Wang",Birthyear:1970,Course-taken:C-wang],其中C-wang表示wang所学课程的集合,则查询age(wang)(N)?将调用方法age,并转换成具体的计算:

$$age(wang)(N):-wang(Name:"Wang",Birthyear:1970),N=Curr\_year-1970$$

规则体中 wang(Name:"Wang",Birthyear:1970)是作为 PERSON 类的对象而忽略属性 Course-taken 代入方法体中的.

在 CDOL 中可实现方法多态,多态分为参数多态和继承多态.参数多态又称为过载(Overload),即方法名相同时,不同的标记指定不同的方法,当一个消息发送到数据库时,消息的标记(类型)将决定哪个方法对此作出响应,不同的参数类型将产生不同的运算.继承多态又称为重载(Overriding),即在超类定义的方法可在子类重新定义,这可在进行方法和消息类型检查时,寻找"最佳"的匹配类型来实现.在上面的例子中,若在 STUDENT 类中重载了方法 age,此时在类层次中有两个方法:age(P:PERSON)(N:int)和 age(S:STUDENT)(N:int),而消息 age(wang)(N)与 age(S:STUDENT)(N:int)是最佳类型匹配,因为 wang 的类型等于 σ(STUDENT).

4 结束语

本文讨论了面向对象的演绎语言中的方法和方法继承等问题,引入了函数 id-项和类型机制.函数 id-项和方法可以实现对象的封装,方法作为接口来存取对象状态.在某些情况下,封装可以打破(如 * 运算),这使得面向值和面向对象数据有机地集成.方法作为规则,其方法名为函数 id-项,它既可以作为方法的调用,又可作为结果对象的标识,因而集成了函数程序设计、面向对象程序设计和逻辑程序设计的特征.类型/子类型在方法继承中起了重要作用,而类型检查(匹配)则实现了方法的多态.在 CDOL 中,一个消息(查询)的谓词项中允许出现变量,这在变量取值域是无限时(如是类型而不是类),将会出现二义性结果,需要进一步的研究.进一步的工作还包括对在对象类型及类/子类层次上的推理研究,它可使计值更为有效,并最终简化查询结果.

参考文献

1 Kifer M, Lausem G. F-logic: a higher-order language for reasoning about objects, inheritance and scheme. In: Proceedings of ACM SIGMOD on Management of Data. Portland, USA. ACM, 1989. 134～146

2 Abiteboul S, Kanellakis P. Object identity as a query language primitive. In: Proceedings of ACM SIGMOD on Management of Data. Portland, USA, ACM, 1989. 159～173

3 Hull R, Yoshikawa M. ILOG: declarative creation and manipulation of object identifiers (extenced abstract). In: International Conference on VLDB(very large data bases). Brisbane, Australia: Morgan Kaufmann Publishers, 1990. 455～468

4 Abiteboul S. Towards a deductive object-oriented database language. In: Proceedings of the 1st International Conference on Deductive and Object-oriented Database. Kyoto, Japan: Elsevier Science Publishers, 1989. 453～472

5 Chen W, Kifer M, Warren D. Hilog: a first-order semantics for higer-order logic programming constructs. In: Proceedings of the 2nd International Workshop on Database Programming Languages. San Mateo: Morgan-Kaufman Publishers, June 1989. 1090～1114

6 张莫成,李修华. CO-LOGIC:一种支持约束演绎 OODB 语言的多类型逻辑.计算机科学,1996,23(4):70～73
(Zhang Dian-cheng, Li Xiu-hua. CO-LOGIC: a many typed logic which supports constraint deductive OODB language. Computer Science, 1996,23(4):70～73)

7 王浩.面向值和面向对象集成的数据模型.合肥工业大学学报,1996,19(4):100～105
(Wang Hao. The data model integrated value-oriented and object-oriented features. Journal of Hefei University of Technology, 1996,19(4):100～105)

8 王浩.智能化 CAX 环境核心数据库的模型及语言的研究[博士论文].合肥工业大学,1996
(Wang Hao. The research of data model and language of the kernel database which supports intelligent Cax environment[Ph. D. Thesis]. Hefei University of Technology. 1996)

9 Zdonik S B, Maier D. Fundamentals of object-oriented database. In: Zdonik S B, Maier D eds. Readings in Object-Oriented Database System. San Mateo: Morgan Kaufmann Publishers, 1990. 1～32

Research of the Method in Deductive Object-Oriented Database Language

WANG Hao　GAO Jun　ZHANG Dian-cheng

(*Institute of Artificial Intelligent Application　Hefei University of Technology　Hefei*　230009)

Abstract　Method and method inheritance are the kernel concepts of object-oriented. This paper introduces the concept and definition of methods in CDOL (constraint deductive object language). Methods are defined by rules. They are used as interface of objects, but they can also be used to achieve abstraction of computation and sharing. The inheritance and polymorphism of methods can be realized through typing and unification.

Key words　Deductive objected-oriented database, deductive query language, method, method inheritance.

Class number　TP311.13