F-logic 语言表达能力的研究*

田增平 王宇君 曲云尧 施伯乐

(复旦大学计算机系 上海 200433)

摘要 F-logic 语言是一种基于框架逻辑的逻辑数据语言,它在表示面向对象的特征方面具有较强的能力,但是讨论其操作复杂对象能力的工作还不多见. 本文比较了以 COL(complex object language)为代表的逻辑数据语言与 F-logic 语言操作复杂对象的能力. 通过两对保持语义的变换,能够将 F-logic 程序及其 Herbrand 解释与 COL 程序及其 Herbrand 解释互相转变,且保持程序在相应解释下的真值性质. 最后,讨论了否定的影响. 本文的工作说明:在不考虑 F-logic 语言 oid 生成影响的情况下,F-logic 语言与 COL,LDL1 和 ELPS 等逻辑数据语言在操作复杂对象方面具有相同的能力.

关键词 逻辑数据语言,F-logic 语言,复杂对象语言.

中图法分类号 TP311.13

众所周知,关系数据模型的建模能力具有一定的局限性. 扩充关系和面向对象等数据模型能够直接描述和操作复杂结构的数据(复杂对象),具有更强的语义建模能力. 基于这些数据模型的代数、演算及逻辑数据语言的表达能力反映了其相应模型的建模能力,是近年来数据库领域研究的重要问题之一.

传统的逻辑数据语言(LDL 和 DATALOG 等)采用不同的形式扩充描述和操作复杂对象的能力,形成各种新语言. 其中有对 DL(logic database language)扩充集合谓词的 LDL1[1];对 DATALOG 扩充有限集合和量词的 ELPS[2];对 DATALOG 扩充数据函数且带类型的 COL(complex object language)[3~5]以及文献[6]中带类型的逻辑数据语言. 已经证明,上列各种语言在描述和操作复杂对象方面具有相同的能力.[2,3,6]

F-logic 是面向对象数据库的逻辑基础,它与面向对象数据库的关系如同一阶谓词逻辑同关系数据库的关系一样. 基于 F-logic 的逻辑数据语言在支持语义建模(描述面向对象特征)方面,比 C-logic 语言[7]、O-Logic 语言[8]及上列各种语言的能力强. 但是,其描述和操作复杂对象的能力还没有同上列语言进行过比较,这正是本文工作的出发点.

由于 ELPS,LDL1 和 COL 在描述和操作复杂对象方面具有相同的能力,本文选择了复

---

* 本文研究得到国家自然科学基金资助.作者**田增平**,1968 年生,博士,主要研究领域为对象多媒体数据库技术及数据库理论. **王宇君**,1970 年生,博士,讲师,主要研究领域为数据库理论及约束数据库技术. **曲云尧**,1961 年生,博士,副教授,主要研究领域为数据库理论及并发控制技术. **施伯乐**,1935 年生,教授,博士导师,主要研究领域为数据库理论及应用.

本文通讯联系人:田增平,上海 200433,复旦大学计算机系

本文 1996-10-16 收到修改稿

杂对象语言COL作为代表,将其与F-logic语言[9]进行了比较.通过两对保持语义的构造性语法和语义变换,能够将F-logic程序及其Herbrand解释与COL程序及相应的Herbrand解释互相转换且保持程序在相应解释中的真值性质.由于两种语言对等词在分层程序中的影响的处理方法不同,两对变换不能保持程序的分层性质.

本文第1、2节分别讨论无否定时F-logic程序及其Herbrand解释与COL程序及相应的Herbrand解释之间的相互变换及有关性质,第3节讨论否定的影响,第4节是总结.

1 F-logic程序到COL程序的变换

本节讨论不带否定的F-logic程序到无否定的COL程序的语法变换、语义变换及性质.

定义 1.1. 对于两种逻辑语言L和L',如果存在两个变换Φ和Ψ分别把L语言的程序及其Herbrand解释转换为L'语言的程序及相应的Herbrand解释,且任给L语言的程序P及其Herbrand解释H,$H\models P$当且仅当$\Psi(H)\models\Phi(P)$,则称Φ和Ψ把L语言嵌入L'语言.

在本节和下一节中,我们将证明F-logic语言和COL语言可互相嵌入.

1.1 语法变换(Φ)

语法变换把F-logic的语法单元变换为COL的语法单元,包括类型系统的定义、平F文字变换、规则变换、附加公理和程序变换.

(1)类型系统定义

所有的id项视为具有类型ID的值.

(2)平F文字的变换

(2.1)类层次F项

定义谓词$SUBCLS$,类层次F项$Q:P$变换为$SUBCLS(Q,P)$,类型为$ID\times ID\rightarrow Boolean$;

(2.2)数据F项

$$P[,\ldots,FunM_i@Q_{i,1},\ldots,Q_{i,k}\rightarrow T_i,\ldots,SetM_j@R_{j,1},\ldots,R_{j,n}\rightarrow\{S_{j,1},\ldots,S_{j,m}\},\ldots],$$

对其组成原子的函数方法定义单值函数$FunM_i$:

$FunM_i(P,Q_{i,1},\ldots,Q_{i,k})=T_i$,类型为$ID\times\ldots\times ID\rightarrow ID$;

对于组成原子的集合方法定义多值函数$SetM_j$:

$SetM_j(P,R_{j,1},\ldots,R_{j,n})\ni S_{j,i},1\leqslant i\leqslant m$,类型为$ID\times\ldots\times ID\rightarrow ID$;

数据F项变换为其组成原子变换结果的合取;

(2.3)基调F项

$$P[,\ldots,FunM_i@Q_{i,1},\ldots,Q_{i,k}\Rightarrow\{A_1,\ldots,A_h\},\ldots,SetM_j@R_{j,1},\ldots,R_{j,n}\Rightarrow\Rightarrow\{B_1,\ldots,B_m\},\ldots],$$

对各方法定义多值函数$SFunM_j$和$SSetM_j$:

$SFunM_i(P,Q_{j,1},\ldots,Q_{j,k})\ni\varnothing$, $SSetM_j(P,R_{j,1},\ldots,R_{j,n})\ni\varnothing$,

$SFunM_i(P,Q_{i,1},\ldots,Q_{i,k})\ni A_j,1\leqslant j\leqslant h$, $SSetM_j(P,R_{j,1},\ldots,R_{j,n})\ni B_i,1\leqslant i\leqslant m$,

类型为$ID\times ID\times\ldots\times ID\rightarrow\{ID\}$;

基调F项变换为其组成原子变换结果的合取;

(2.4)平P项视为c原子,类型为$ID\times\ldots\times ID\rightarrow Boolean$;

(2.5)对于 $t[\ ]$,引入谓词 OBJ. 它变换为 $OBJ(t)$,类型为 $ID \rightarrow Boolean$;

(2.6)若 γ 的变换为 γ',则负文字 $\sim\gamma$ 变换为 $\sim\gamma'$.

(3)对规则 r 分别变换其头和体中的文字

(4)附加公理(AM(P))

为了使变换后的程序仍保持 F-logic 程序的某些语义信息(如类层次 F 项的传递性),对 F-logic 程序 P 的变换结果需加入适当的公理(其中各变量均具有 ID 类型).

(4.1)类层次公理

- $SUBCLS(X,X) \leftarrow$
- $SUBCLS(X,Z) \leftarrow SUBCLS(X,Y) \& SUBCLS(Y,Z)$
- $X=Y \leftarrow SUBCLS(X,Y) \& SUBCLS(Y,X)$

(4.2)等价公理

- $X=X \leftarrow$
- $X=Y \leftarrow Y=X$
- $X=Z \leftarrow X=Y \& Y=Z$
- $W[Y] \leftarrow W[X] \& X=Y$, $W[X]$:为含 X 的公式

(4.3)类型公理

对每个基调 F 项的多值函数 SM:

- $SM(X_1,\ldots,X'_i,\ldots,X_n)\exists\ Y \leftarrow SM(X_1,\ldots,X_i,\ldots,X_n)\exists\ Y \& SUBCLS(X'_i,X_i)$
- $SM(X_1,\ldots,X_n)\exists\ Y \leftarrow SM(X_1,\ldots,X_n)\exists\ Y' \& SUBCLS(Y',Y)$

对于每个函数方法的单值函数 $FunM$ 及其基调的多值函数 $SFunM$:

- $SUBCLS(Y,Z) \leftarrow FunM(X_1,\ldots,X_n)=Y \& SFunM(X_1,\ldots,X_n)\exists\ Z$
- $SFunM(X_1,\ldots,X_n)\exists\ \varnothing \leftarrow FunM(X_1,\ldots,X_n)=Y$

对于每个集合方法的多值函数 $SetM$ 及其基调的多值函数 $SSetM$:

- $SUBCLS(Y,Z) \leftarrow SetM(X_1,\ldots,X_n)\exists\ Y \ \& \ SSetM(X_1,\ldots,X_n)\exists\ Z$
- $SSetM(X_1,\ldots,X_n)\exists\ \varnothing \leftarrow SetM(X_1,\ldots,X_n)\exists\ q$

(4.4)一致性约束

- $Y_1=Y_2 \leftarrow FunM(X_1,\ldots,X_n)=Y_1 \ \& \ FunM(X_1,\ldots,X_n)=Y_2$
- $OBJ(X) \leftarrow$

(5)程序 P 的变换

$\Phi(P)=AM(P) \cup P'$,P' 为 P 中各规则的变换集.

1.2 语义变换(ψ)

语义变换从 F-logic 程序的 Herbrand 解释构造相应 COL 程序的 Herbrand 解释,包括 H 域构造和 H 解释构造.

(1)H 域构造

若 F-logic 程序 P 的 H 域为 U^f,则 $\Phi(P)$ 的 H 域为 $\psi(U^f)$. 它是以 U^f 为常量集,由集合和元组构造子构成的带类型集合.

(2)H 解释构造

若 F-logic 程序 P 的 H 解释为 H^f,则 $\Phi(P)$ 的 H 解释 $\psi(H^f)=\{a' \mid a \in H^f, a' \text{为} a \text{依} \Phi$

的变换结果}.

1.3 Φ 和 ψ 变换的性质

本小节通过一些命题、引理和定理证明 Φ 和 ψ 是保持程序语义的变换(证明略).

命题 1.1. B^f 的子集 H 在"$\models$"下封闭(即形成 H 解释),当且仅当它具有下列性质:

①"="形成 H 的同余关系;

②类层次形成偏序关系;

③基调 F 项满足:

(1)类继承:

- $p[FunM\ @q_1,\ldots,q_k\Rightarrow\{s\}],r::p\in H$, 则 $r[FunM@q_1,\ldots,q_k\Rightarrow\{s\}]\in H$;
- $p[SetM\ @q_1,\ldots,q_k\Rightarrow\Rightarrow\{s\}],r::p\in H$, 则 $r[SetM@q_1,\ldots,q_k\Rightarrow\Rightarrow\{s\}]\in H$;

(2)超类-1:

- $p[FunM\ @q_1,\ldots,q_i,\ldots,q_k\Rightarrow\{s\}],q'_i::q_i\in H$, 则
  $p[FunM\ @q_1,\ldots,q'_i,\ldots,q_k\Rightarrow\{s\}]\in H$;
- $p[SetM\ @q_1,\ldots,q_i,\ldots,q_k\Rightarrow\Rightarrow\{s\}],q'_i::q_i\in H$, 则
  $p[SetM\ @q_1,\ldots,q'_i,\ldots,q_k\Rightarrow\Rightarrow\{s\}]\in H$;

(3)超类-2:

- $p[FunM\ @q_1,\ldots,q_k\Rightarrow\{r\}],r::s\in H$, 则 $p[FunM\ @q_1,\ldots,q_k\Rightarrow\{s\}]\in H$;
- $p[SetM\ @q_1,\ldots,q_k\Rightarrow\Rightarrow\{r\}],r::s\in H$, 则 $r[SetM\ @q_1,\ldots,q_k\Rightarrow\Rightarrow\{s\}]\in H$;

(4)良类型:

- $p[FunM\ @q_1,\ldots,q_k\rightarrow q]$, $p[FunM\ @q_1,\ldots,q_k\Rightarrow\{r\}]$, 则 $q::r\in H$;
- $p[SetM\ @q_1,\ldots,q_k\rightarrow\rightarrow\{q\}]$, $p[SetM\ @q_1,\ldots,q_k\Rightarrow\Rightarrow\{r\}]$, 则 $q::r\in H$;
- $p[FunM\ @q_1,\ldots,q_k\rightarrow q]$, 则 $p[FunM\ @q_1,\ldots,q_k\Rightarrow\{\}]\in H$;
- $p[SetM\ @q_1,\ldots,q_k\rightarrow\rightarrow\{q\}]$, 则 $p[SetM\ @q_1,\ldots,q_k\Rightarrow\{\}]\in H$;

④ 数据 F 项

若 $p[FunM\ @q_1,\ldots,q_k\rightarrow r_1]\in H$, $p[FunM\ @q_1,\ldots,q_k\rightarrow r_2]\in H$, 则 $r_1=r_2\in H$;

⑤基 id 项 t,则 $t[\ ]\in H$;

⑥每个基 F 项 t,$t\in H$,当且仅当其所有组成原子属于 H.

命题 1.2. 若基 c 文字 $\varphi\in H^c$,则对任何求值 θ_H,有 $H^c\models\theta_H(\varphi)$.

引理 1.1. 任给一基 F-logic 文字 φ,H 解释 H^f,则 $H^f\models\varphi$ 当且仅当对任何求值 θ_H,$\psi(H^f)\models\theta_H(\Phi(\varphi))$.

引理 1.2. 任给一 F-logic 程序 P 及其 H 基的子集 H^f,则 H^f 为 P 的 H 解释当且仅当 $\psi(H^f)\models AM(P)$.

定理 1.1. 对于任一 F-logic 程序 P 及其 H 解释 H^f,$H^f\models P$ 当且仅当 $\psi(H^f)\models\Phi(P)$.

推论 1.1. 变换 Φ 及 ψ 将 F-logic 语言嵌入 COL 语言.

2 COL 程序到 F-logic 程序的变换

本节讨论不带否定的 COL 程序的语法及语义到 F-logic 程序及其语义上的变换.

2.1 语法变换(φ)

语法变换把COL语言的语法单元变换为F-logic语言的语法单元,包括类型系统变换、类型约束公理、原子的变换、规则变换和程序变换.

定义 2.1. 类型和值的变换项定义为

若$t\equiv a$,为原子类型或原子类型值,则其变换项$t'\equiv a$;

若$t\equiv[t_1,\ldots,t_n]$,则其变换项$t'\equiv TP_n(t_1',\ldots,t_n')$;

若$t\equiv\{t_1,\ldots,t_n\}$,则其变换项$t'\equiv ADD(t_n',ADD(\ldots,ADD(t_1',\varphi)\ldots))$;

若$t\equiv F(t_1,\ldots,t_n)$,则其变换项$t'\equiv F(t_1',\ldots,t_n')$.

其中F和TP_n是n元对象构造子,ADD是二元对象构造子.

(1)类型系统变换($TY(P)$)

COL语言隐含的类型说明在F-logic中引入谓词$TYPE$来表示.

(1.1)对程序P中每个原子类型T_i及常量a,$TYPE(a,T_i)$为真当且仅当$a\in dom(T_i)$;

(1.2)对每个k元组类型$[T_1,\ldots,T_k]$,引入k元对象构造子TP_k,$TYPE(TP_K(a_1',\ldots,a_k'),TP_K(T_1',\ldots,T_k'))$为真当且仅当$(a_1,\ldots,a_k)\in dom([T_1,\ldots,T_k])$. $TP_K(T_1',\ldots,T_k')$称为$TP_K(a_1',\ldots,a_k')$的类型约束项;

(1.3)对集合类型引入二元对象构造子ADD,对于k类集合$\{T_1,\ldots,T_k\}$,$TYPE(ADD(a_n',ADD(\ldots,ADD(a_1',\varphi)\ldots)),ADD(T_k',ADD(\ldots,ADD(T_1',\varphi)\ldots)))$为真,当且仅当$\{a_1,\ldots,a_n\}\in dom(\{T_1,\ldots,T_k\})$. $ADD(T_k',ADD(\ldots,ADD(T_1',\varphi)\ldots))$称为$ADD(a_n',ADD(\ldots,ADD(a_1',\varphi)\ldots))$的类型约束项;

(1.4)对每个数据函数$F(a_1,\ldots,a_k)$,引入k元对象构造子F. 若F的取值类型为T_f,则$TYPE(TP_K(a_1',\ldots,a_k'),T_f')$为真. T'_f称为$TP_K(a_1',\ldots,a_k')$的类型约束项.

(2)类型约束公理($TM(P)$)

这组公理的目的在于将COL语言隐含的类型关系在非类型语言F-logic语言中用公理的形式表示.

(2.1)对COL程序中的k元组类型

$\cdot\ TYPE(TP_k(X_1,\ldots,X_k),TP_k(Y_1,\ldots,Y_k))\leftarrow TYPE(X_1,Y_1)\&\ldots\&TYPE(X_k,Y_k)$

(2.2)对COL程序中的k类集合类型

$\cdot\ TYPE(ADD(X_n,ADD(\ldots,ADD(X_1,\varphi)\ldots)),ADD(Y_k,ADD(\ldots,ADD(Y_1,\varphi)\ldots)))\leftarrow TYPE(ADD(X_1,\varphi),ADD(Y_k,ADD(\ldots,ADD(Y_1,\varphi)\ldots)))\&,\ldots\ldots\&TYPE(ADD(X_n,\varphi),ADD(Y_k,ADD(\ldots,ADD(Y_1,\varphi)\ldots)))$

$\cdot\ TYPE(ADD(X,\varphi),ADD(Y_k,ADD(\ldots,ADD(Y_1,\varphi)\ldots)))\leftarrow TYPE(ADD(X,\varphi),ADD(Y_1,\varphi))$

$\vdots\qquad\qquad\vdots\qquad\qquad\vdots$

$\cdot\ TYPE(ADD(X,\varphi),ADD(Y_k,ADD(\ldots,ADD(Y_1,\varphi)\ldots)))\leftarrow TYPE(ADD(X,\varphi),ADD(Y_k,\varphi))$

(2.3)ADD的性质

$\cdot\ ADD(X,ADD(Y,S))=ADD(Y,ADD(X,S))$,集合元素的次序无关.

(3)原子变换

(3.1)对形如$F(t_1,\ldots,t_n)\exists\ t$的$c$原子,$F$视为$n$元对象构造子,$\exists$作为谓词.它变换为

$F(t_1',\ldots,t_n')\exists\ t'\cap TYPE(t_1',T_1')\cap\ldots\cap TYPE(t_n',T_n')\cap TYPE(t',T')\cap TYPE(F(t_1',\ldots,t_n'),T_F')$,

其中T_i', T', T_F'分别为t_i', t'及F值的类型约束项;

(3.2)对于形如$f(t_1,\ldots,t_n)=t$的c原子,f视为n元对象构造子.它变换为

$f(t_1',\ldots,t_n')=t'\cap TYPE(t_1',T_1')\cap\ldots\cap TYPE(t_n',T_n')\cap TYPE(t',T_f')\cap TYPE(f(t_1',\ldots,t_n'),T_f')$,

其中T_i'和T_f'分别为t_i'和f的类型约束项;

(3.3)形如$P(t_1,\ldots,t_n)$的c原子变换为

$P(t_1',\ldots,t_n')\cap TYPE(t_1',T_1')\cap\ldots\cap TYPE(t_n',T_n')$,

T_i'为t_i'的类型约束项.

上列变换中由$TYPE$谓词形成类型约束式,含有$\exists$、$=$及非$TYPE$谓词的文字称主变换式;

(3.4)负c原子$\sim\varphi$变换为

$\sim\varphi\hat{}\ \wedge\gamma$, 其中$\varphi\hat{}$为$\varphi$的主变换式,$\gamma$为$\varphi$的类型约束式.

(4)规则变换

$r\equiv A\leftarrow B_1\&\ldots\&B_n$变换为$r'\equiv A\hat{}\leftarrow B_1\hat{}\&\ldots\&B_n\hat{}\&\varphi_1\&\ldots\&\varphi_n\&\gamma$.

其中$A\hat{}$和$B_i\hat{}$分别为A和B_i的主变换式,φ_i和γ分别为B_i和A的类型约束式.

(5)程序P的变换

$\varnothing(P)=TY(P)\cup TM(P)\cup P'$,$P'$为$P$中规则的变换集.

2.2 语义变换(Ψ)

语义变换从COL程序的Herbrand解释构造相应F-logic程序的Herbrand解释.

(1)H域的构造

给定COL程序P的H域U^c,$\varnothing(P)$的H域$\Psi(U^c)$由下列规则构造:

(1.1)若原子常量$a\in U^c$,且类型为T,则a, $T\in\Psi(U^c)$;

(1.2)若元组类型项$[a_1,\ldots,a_n]\in U^c$,且类型为$[T_1,\ldots,T_n]$,则$TP_n(a_1',\ldots,a_n')$, $TP_n(T_1',\ldots,T_n')\in\Psi(U^c)$;

(1.3)若集合类型项$\{a_1,\ldots,a_n\}\in U^c$,且类型为$\{T_1,\ldots,T_n\}$,则$ADD(a_n',ADD(\ldots,ADD(a_1',\varphi)\ldots))$, $ADD(T_n',ADD(\ldots,ADD(T_1',\varphi)\ldots))\in\Psi(U^c)$;

(1.4)若F为数据函数,取值类型为T_F,且$F(a_1,\ldots,a_n)\in U^c$,则$F(a_1',\ldots,a_n')$, $T_F'\in\Psi(U^c)$.

(2)H解释的构造

(2.1)若$t'\in\Psi(U^c)$,则$t'=t'$, $t'[\ ]$, $TYPE(t',T')\in\Psi(H^c)$,T是t的类型;

若$ADD(s,ADD(T,\varphi))\in\Psi(U^c)$,则$[ADD(T,ADD(s,\varphi))=ADD(s,ADD(T,\varphi))]\in\Psi(H^c)$;

(2.2)若$R(t_1,\ldots,t_n)\in H^c$,则$R(t_1',\ldots,t_n')$, $TYPE(t_i',T_i')\in\Psi(H^c)$;

(2.3)若$[F(t_1,\ldots,t_n)\exists\ t]\in H^c$,则$F(t_1',\ldots,t_n')\exists\ t'$, $TYPE(F(t_1',\ldots,t_n'),T_F')$,

$TYPE(t_i',T_i')$, $TYPE(t',T')\in\Psi(H^c)$;

(2.4)若$[f(t_1,\ldots,t_n)=t]\in H^c$, 则 $f(t_1',\ldots,t_n')=t'$, $TYPE(f(t_1',\ldots,t_n'),T'_f)$, $TYPE(t'_i,T'_i)$, $TYPE(t',T')\in\Psi(H^c)$;

(2.5)若$[F(t_1',\ldots,t_n')\exists\ t']\in\Psi(H^c)$, 则$[F(t_1',\ldots,t_n')=\{a\mid[F(t_1',\ldots,t_n')\exists\ a']\in\Psi(H^c)\}']\in\Psi(H^c)$;

(2.6)若$[t_1=t_2]$, $[t_2=t_3]\in\Psi(H^c)$, 则$[t_1=t_3]\in\Psi(H^c)$;

(2.7)若 $t_1=t_2\in H^c$, 则$[t_2=t_1]\in\Psi(H^c)$;

(2.8)对含有项 t 的公式 $\varphi[t]$, 若 $\varphi[t]\in\Psi(H^c)$ 且 $t=h\in\Psi(H^c)$, 则 $\varphi[h]\in\Psi(H^c)$.

2.3 变换∅和 Ψ 的性质

下列命题、引理和定理能够保证∅和 Ψ 是保持程序语义的变换(证明略).

命题 2.1. 任给 COL 程序 P 及其 H 解释 H^c, 则 $\Psi(H^c)$为$\varnothing(P)$的 H 解释.

引理 2.1. 基 c 文字 φ 在 COL 程序 P 的 H 解释 H^c 中被满足, 当且仅当$\varnothing(\varphi)$在 $\Psi(H^c)$下为真.

引理 2.2. 给定 COL 程序 P 及其 H 解释 H^c, $\Psi(H^c)\models TYPE(a',T')$当且仅当 U^c 中 a 具有类型 T.

引理 2.3. 给定 COL 程序 P 及其 H 解释 H^c 和一个 F-logic 基代入 $V:Var(\varnothing(P))\to\{\sigma'_i(a'_{i,k}),T'_i\}$, $\Psi(H^c)\models V(TY(P)\cup TM(P))$当且仅当 $V':Var(P)\to\{\sigma_i(a_{i,k})\}$是类型合适的 COL 基替换($Var(P)$:$P$ 中的变量, $\sigma'_i(a'_{i,k})$:含有 $a'_{i,k}$的项, T_i 为 $\sigma_i(a_{i,k})$ 的类型).

定理 2.1. 对任一 COL 程序 P 及其 H 解释 H^c, $H^c\models P$ 当且仅当 $\Psi(H^c)\models\varnothing(P)$.

推论 2.1. 变换∅和 Ψ 将 COL 嵌入 F-logic 语言.

3 否定的影响

允许在规则体中使用负文字,会对变换程序的分层性质带来一定的影响. 对于局部分层的 F-logic 程序和分层的 COL 程序有下列否定结果.

命题 3.1. ①若 P 是局部分层的 F-logic 程序, 则 $\Phi(P)$不一定是分层的 COL 程序;②若 P 为分层的 COL 程序, 则$\varnothing(P)$不一定是局部分层的 F-logic 程序.

①由分层程序和局部分层程序的定义而得. ②对分层的 COL 程序 P, 由于等词的原因, 可使 $\varphi(P)$不再是局部分层的 F-logic 程序. 例如:若 P 是 $f=q\leftarrow\sim T$, 为分层 COL 程序, 但是 $\varphi(P)$:$f'=q'\leftarrow\sim\varphi(T)$不是局部分层的 F-logic 程序.

对带等词和否定的逻辑程序, F-logic 采取较严格的方法, 绝对限制局部分层的程序中等词依赖负文字;而 COL 采取较宽松的处理办法, 在分层程序中, 仅仅不允许等词环形依赖否定文字, 因而导致了上列结果.

4 结束语

对于只含有集合和元组两种构造子的逻辑数据语言的表达能力, Kuper 证明了 ELPS 与 LDL 等价[2], Abitrboul 断言 ELPS, LDL, COL 及文献[6]中所定义的代数和安全演算具有相同的能力.[4,5]本文的工作说明, F-logic 语言在不考虑对象构造子影响的情况下, 在操

作复杂对象方面，与上列各种逻辑数据语言具有相同的能力. 否定带来的影响主要是由于对等词的处理方法不同引起的. 关于带否定情况的等词处理仍是逻辑数据语言研究的问题之一.

参考文献

1 Beeri C *et al*. Sets and negation in a logic database language(LDL1). In: Proceedings of the 6th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Diego, USA, ACM, 1987. 21～37.

2 Kuper M. On the expressive of logic programming languages with sets. In: Proceedings of the 7th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. Austin, USA, ACM, 1988. 10～14.

3 Abiteboul S *et al*. COL: a logic-based language for complex objects. In: Schmidt J W, Ceri S, Missikoff eds, Advances in Database Technology——EDBT'88, Venice, Italy, Springer-Verlag, 1988. 271～293.

4 Abiteboul S *et al*. A rule-based language with functions and sets. ACM TODS, 1991,**16(1)**:1～30.

5 Abiteboul S *et al*. Data functions, datalog and negation. In: Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data, Chicago, USA, ACM, 1988. 143～153.

6 Abiteboul S *et al*. On the power of languages for the manipulation of complex objects. Technical Report, INRIA, 1993. 1～76.

7 Chen W *et al*. C-logic for complex objects. In: Proceedings of the 8th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Philadelphia, USA, ACM, 1989. 369～378.

8 Kifer M *et al*. A logic for programming with complex objects. J. Comput. Syst. Sci. 1993,**47(1)**:77～120.

9 Kifer M *et al*. Logic foundation of object-oriented and frame based language. Technical Report 92/23, Department of Computer Science, SUNY at Stony Brook, New York, 1992.

ON THE EXPRESSIVE POWER OF F-LOGIC LANGUAGE

TIAN Zengping WANG Yujun QU Yunyao SHI Baile

(*Department of Computer Science Fudan University Shanghai* 200433)

Abstract F-logic language is a logic database language based on frame logic. It is powerful in expressing object-oriented features. However, there was little work in discussing its capability of manipulating complex objects. In this paper, the authors compare the capability of F-logic with that of logic database languages represented by COL (complex object language). Through two pairs of semantic preserving transformations, F-logic programs and their Herbrand interpretations can be transformed into COL programs and their corresponding Herbrand interpretations, and vice versa. Also, the effects of negation are discussed. The results of this paper indicate that, without consideration of the effects of OID generating, F-logic language has the same power in manipulating complex objects as COL, LDL1(logic database language), and ELPS.

Key words Logic database language, F-logic language, complex object language.

Class number TP311.13