国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第14卷第3期 1992年9月 Vol. 14 No. 3

结合逻辑和面向对象范例的一种新模型（SCKE）*

金 芝 胡守仁

（电子计算机系）

摘 要 逻辑语言和面向对象语言是两种引人注目的知识程序设计语言。如何结合它们各自的优点以支持复杂的知识处理，已成为许多研究者关注的问题。本文首先提出了结合逻辑和面向对象范例的一种新模型SCKE，其特点是在统一的逻辑语义和对象语义解释的基础上，同时支持逻辑语言的描述性特征和面向对象语言的结构化、信息隐藏、继承等性质。本文推广了逻辑语言的Herbrand解释，并由此探讨了SCKE模型的形式语义。文中讨论了实现该模型的基于预编译的元级扩充方法，并论述了SCKE模型的主要特点。

关键词 逻辑语言，面向对象语言，模型，Herbrand解释。

分类号 TP18

逻辑语言和面向对象语言是两种引人注目的知识程序设计语言。面向对象语言的核心是把知识组织成对象的集合，对象间的相互作用通过消息传送来完成，对象组织成一种层次结构，且下层对象可以继承其上层对象的属性。由于面向对象语言具有模块化结构、封装功能、继承和动态链接等特性，特别适合于构造大型可重用知识系统。但是现有的面向对象语言说明能力弱，不具备演绎推理能力，不适于描述说明性知识和支持知识系统的增量式开发。

另一方面，逻辑语言是一种描述性语言。它通过自动搜索、一致化匹配和回溯来求解问题，具有演绎推理能力，语义清晰，表达能力强，支持知识的表示和知识系统的增量式开发。但是，逻辑程序的平坦化结构妨碍了它对大型知识系统的支持。

结合逻辑与面向对象范例以支持复杂的知识处理，已成为许多研究者们关注的课题。从目前的研究状况来看，该课题的研究大致分为两个分支：基于顺序逻辑程序(Prolog)[3][7][9][12][15][16]和基于并行逻辑程序[6][8][13][18]。

本文提出了结合逻辑和面向对象范例的一种新模型SCKE(Structured Communicating Knowledge Entities)，其特点是在统一的逻辑语义和对象语义解释的基础上，同时支持逻辑语言的描述性特征，和面向对象语言的结构化、信息隐藏、继承等性质，用一种自然而有效的方式开发了这两种语言范例的优点。本文还在逻辑语言的Herbrand解释基础上，探讨了逻辑对象的语义解释，从而刻划了SCKE模型的语义。开发SCKE模型的目的是支持大型知识系统的结构化、系统性开发，并以此建立知识的并行处理环境。

* 国家高技术发展计划资助项目
** 1991年10月3日收稿

1 SCKE 模型

SCKE 模型是合成逻辑和对象的一种新型集成式语言范例，它在标准逻辑程序语义基础上，引入面向对象范例中的如下典型概念。

(1) 类＝模块

面向对象范例的核心是“模块≡类”[2]。其中，模块是控制名范围的实现层结构，类是管理对象过程定义的程序抽象。我们合成逻辑与对象范例的总体原则是利用低层模块机制封装对象和类，即模块提供了对象或类代码周围的“隔火墙”，语言解释器则支持穿越模块边界的名约束功能。

在 SCKE 模型中，程序由对象组成，每个对象都包含一组独立的相关 Prolog 子句。类是同一类对象建立的模板。类的语法描述如下所述：

```
〈class〉 ::=class ( 〈name〉).
                〈interface〉
                〈method_clause〉
                〈instvari_clause〉
          endclass.
〈interface〉 ::=nil
〈interface〉 ::= 〈super_spec〉 | 〈interface〉
〈interface〉 ::= 〈meta_spec〉 | 〈interface〉
〈super_spec〉 ::=super ( 〈name〉).
〈meta_spec〉 ::=meta ( 〈name〉).
〈name〉 ::= 〈atom〉
```

其中，〈name〉为一个原子，表示类的名字，〈interface〉表示该类与其它类的关系，〈super_spec〉指出类之间的继承关系，〈meta_spec〉指明元类/类关系。〈method_clause〉中定义的方法用 Horn 子句表示，即形为：

$$A:-B_1, \cdots, B_n$$

的逻辑蕴含式，其中 A，B_i(i＝1，…，n)为原子公式，子句头 A 表示方法的选择模式。如果一个方法有多种定义，可用回溯寻找正确的求解方法。

(2) 消息＝目标

独立的 Prolog 程序模块(类)之间可通过发送消息进行通讯。向某一对象发送一条消息可解释为请求该对象证明一个目标。消息的语法形式为：

Object::Goal

其中，中缀谓词“::/2”可以任意地嵌入程序中，其语义为：“在对象 Object 定义的环境中证明目标 Goal。”若证明成功，则该消息目标为真，否则回答“no”。

将方法表示为 Horn 子句，方法的调用解释为目标证明，具有如下特性：

(a) 具有描述性风格，有助于建立基于知识的系统；

(b) 一致化功能使目标参数既可作为方法操作的输入参数，又可作为输出参数，具有灵活的操作调用入口；

(c) 方法可有多种定义，回溯用于寻找正确的求解方法；

(d) 方法语法与 Prolog 程序完全一致，利于编程。

(3) 继承性

在面向对象的系统中，继承关系用于共享相关对象组之间的知识。在 SCKE 中，组成对象理论的公理集分为两部分：共享部分和私有部分，其中共享部分包含方法定义，私有部分包含实例变量。对象间的继承关系用 super 表示。一个对象的继承链为一个类序列：

$$C=C_1, C_2, \cdots, C_i, \cdots, C_n$$

其中，C 为该类本身，C_n 为对象系统的根对象。对于所有的 $i(1\leqslant i\leqslant n-1)$，在 C_i 的类定义中存在外部接口说明 super(C_{i+1})。

按 Prolog 的语义，带有继承链的对象理论中，可见（或可访问）的方法包括：(a)如果方法是该对象类中的一个子句头，则该方法在该对象中可见；(b)如果方法在该对象类继承链上的直接后继类中可见，则该方法在该对象中也可见。

多重继承通过回溯支持。一个类对象可以有多个超类，这时该对象的继承链有多条，多继承通过回溯依次在这些链上搜索完成。定义超类的顺序决定了继承链的搜索顺序。

(4) 元级机制

SCKE 模型中的一个重要概念，是将对象的问题求解策略作为元级程序设计的一个典型应用[8]。为了支持这个概念，SCKE 严格区分目标级对象和元级对象，并提供两者之间的交互机制。一个元级对象通过目标级类定义中的说明语句：

meta_class(〈M_class_name〉)

与该目标级对象相连。当一个目标级对象与一个元级对象联系在一起时，该目标级对象每次被激活时都自动与其元级对象通讯，并由元级对象控制目标级对象的行为。

由于这种元级机制允许程序员定义控制目标级对象的推理策略，便于融合多种知识表示风格和推理方法，并支持多模式推理间的自然切换。

考虑下面简单的对象程序：

```
class(S_entity).
        super(S_entity).
endclass.
class(animal).
        super(S_entity).
        like(eating).
        mode(walk):-no_of_leg(X),integer::lesseq(2,X).
endclass.
class(bird).
        super(animal).
        no_of_leg(2).
endclass.
class(human).
        super(animal).
        no_of_leg(2).
```

```
        agelimit(200):-property(common_human).
        like(woman):-sex(male).
endclass.
class(john).
        super(human).
        property(common_human).
endclass.
class(integer).
        super(S_entity).
        lesseq(X,Y):-X<Y;X=Y.
endclass.
```

其继承树如图1. 其中根对象的反馈超类关系主要是为整个对象系统提供一致的语义，继承机制自动识别根对象并终止继承行为。

2 SCKE 的逻辑语义

一个面向对象的逻辑程序是逻辑对象的一个层次结构，其中每个对象含有一组有限的确定子句。作为一种程序风格，它对逻辑程序的扩充主要在于消息和继承两个方面，即一个逻辑对象可通过发送消息与其它逻辑对象通讯，并继承其超类对象的操作。从逻辑程序的观点出发，继承可看作为逻辑理论的组合。当对象 obj 接受到消息 obj::G 时，G 实际上是在 obj 以及其继承链上所有类对象定义的公理集上求解。另外，消息可解释为目标，向一个对象发送消息即为用与该对象相关的确定子句求解某个目标。因此，基于逻辑程序的过程语义，面向对象逻辑程序 P 的过程语义可用如下定义描述。

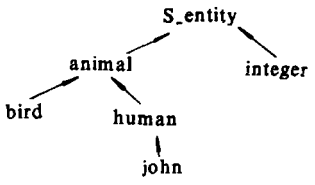


图1

定义 (SCKE 的归结)

程序 P 自顶到底导出目标 G 的过程由目标子句序列：$G_0=G$，G_1，……，以及 mgu 序列：θ_1，θ_2，……组成，其中 G_{i+1} 由 G_i 用 $mgu\theta_{i+1}$ 导出。若导出过程有限，且最终目标为空子句，即 $G_n=\square$，称该导出是成功的，此时记 $P\models_k G(k=\theta_1\theta_2\cdots\theta_n)$。

这里，SCKE 的导出规则为：

(1) $\rightarrow \{true,\_\}$

(2) $\{A_1O_iO_j\}_k$，$\{B,\ O_iO_j\}_l \rightarrow \{A\wedge B,\ O_iO_j\}_{kl}$

(3) $\{Bk,\ O_iO_j\}_l \rightarrow \{G \mid G$ 为一个原子，$O_iO_j\}_{kl}$

当存在对象 O_i 到 O_j 继承链上定义的子句变体$(H':-B)$，且 $k=mgu(G,H')$。

(4) $\{G,\ O_iO_s\}_l \rightarrow \{G,\ O_iO_j\}_l$，当 $O_j\neq$根对象，且 O_s 是 O_j 的超类。

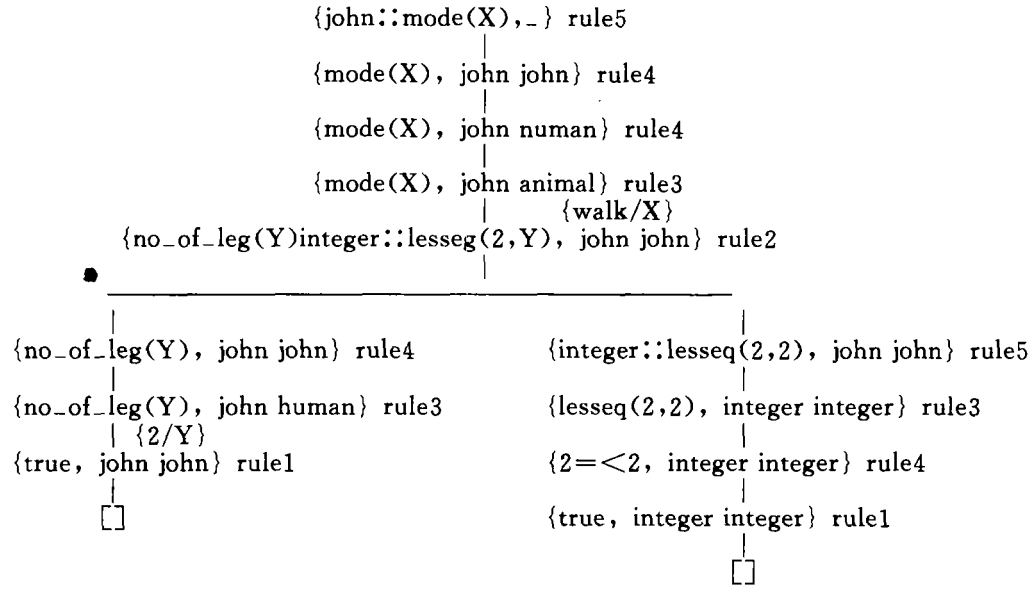
(5) $\{A,\ O_jO_j\}_- \{O_j::A,\ \_\}_l$

其中，O_i，O_j，O_s 表示 P 中的逻辑对象。

考虑上节中对象程序，若发送消息：

```
john::mode(X)
```

则其导出过程为：



逻辑对象既可隐式地（继承）又可显式地（消息）访问“外部世界”，即对单个逻辑对象来说，封闭世界假设可以不再有效。这种开放逻辑对象也不能用简单的最小 Herbrand 模型来解释。本文提出“可见条件 Herbrand 模型”，并用它描述逻辑对象的描述性语义。下面为简单起见，假设面向对象逻辑程序 P 中所有对象都只含基础子句。

定义 （可见条件 Herbrand 模型 VCH）

给定程序 P 中的一个对象 O，对 O 的每个继承链 $O_0=O$，O_1，…，O_{root}，其中 O_{i+1} 是 O_i 的超类。令 O′包含 O 中所有可见子句，即 $|O'|=|O_0|\cup|O_1|\cup\cdots\cup|O_{root}|$，Msg(O′)为出现在 O′中消息的集合 O′，H 是 Msg(O′)的子集，则可见条件 Herbrand 模型 VCH $(O)^H$ 为 O″的最小 Herbrand 模型，其中 O″为对 O′进行如下操作所得。

(1) 删除所有属于 H 的消息目标；

(2) 如果存在子句体中含有消息 Obj::A，且该消息不属于 H，则删除该子句。

如上节例中的 human 对象，其可见子句集 O′为：

```
no_of_leg(2).
agelimite(200):-property(common_human).
like(woman):-sex(male).
like(eating).
mode(walk):-no_of_leg(X),integer::lesseq(2,X).
```

子句体中消息的集合为

Msg(O′)={integer::lesseq(2,X)}

假设 H= {integer::lesseq(2,X)}，按定义得 O″为一个标准逻辑程序：

```
no_of_leg(2).
agelimit(200):-property(common_human).
like(woman):-sex(male).
```

```
like(eating).
mode(walk):-no_of_leg(X).
```

O'的最小 Herbrand 模型为：

$$M=\{no\_of\_leg(2),like(eating),mode(walk)\}$$

开放逻辑对象 human 的指称可由如下可见条件 Herbrand 模型集表示：

$$VCH(human):-\{\{no\_of\_leg(2),like(eating)\}^{\{\}},\ \{no\_of\_leg(2),like(eating),mode(walk)\}^{\{integer::lesseq(2,X)\}}\}$$

3 SCKE 实现

为了扩展 Prolog 以支持 SCKE 模型中的概念，提出如下三种方法：

(1) 在 Prolog 上开发一个元级解释器[13]，快速构造 SCKE 原型，以模拟 SCKE 的程序描述和行为。

(2) 扩展现有的 Prolog 解释器，如 GKD—Prolog 解释器，以求在语言解释执行一级直接嵌入面向对象风格，为逻辑程序语义和面向对象语义提供一个统一的解释机制。

(3) 在 Warren 抽象机[4]一级定义一组抽象指令，以提供合成语言编译器。这不仅有利于提高程序的执行效率，而且有助于对支持逻辑和面向对象风格体系结构的研究。

目前，我们已完成第一部分的工作，其它两方面的工作也正在进行中。这里主要介绍元级解释器的实现。如同第二节中所述，要在元级支持对象的概念，首先必须用模块组织谓词以达到封装对象的目的，简单地说，就是限制谓词作用的范围。一种方法就是将模块 M 中的谓词 P 命名为唯一全局名 "M:P"。如果所有的子句都用全局谓词名描述，则可在一般非模块化 Prolog 上完成模块化功能。

我们已经实现了一个名扩展预处理器[12]。它用上述方法将模块化程序转化为一般的 Prolog 程序。其内部谓词作为全局可见谓词，转换过程中保持不变。对于 Prolog 中的元逻辑谓词，如 call，clause，assert 等，因为它们在执行时可能改变运行环境，程序变换时不作处理，仅将它们与各自的元解释器相连，程序运行时再进行动态变换后解释执行。另外还提供了动态创建模块的功能以支持对象的动态创建。

有了模块系统的支持，我们可以较方便地在元级嵌入面向对象风格，主要工作包括消息和继承的实现。消息 "M::P" 意指请求模块 M 证明目标 P. 由于 M 可以继承其超类的方法，实际上对 P 的证明也是由继承机制完成的。"::/2" 的简单实现可表示为：

```
::(Object,Message):-
    inherit(Message,Object,Object).
```

继承机制实际上就是面向对象逻辑系统的解释机制。它用对象继承链上的组合公理集证明相应的目标。其简化的实现过程如下：

```
inherit (Message,Object,Object):-
    Object:Message.
inherit(Message,Object,Objectl):-
    Object≠Objectl,
    Objectl:clause(Message,Body),
    inherit(Body,Object,Object).
```

```
inherit(Message,Object,Object):-
        super(Object,Objectl),
        Object≠Objectl,
        inherit(Message,Object,Objectl).
```

当遇到循环超类关系（根对象）时，继承过程自动终止。

4 结 论

本文提出了一种新颖的合成逻辑和面向对象风格的方法，与其它方法相比，它具有如下主要特点：

(1) 采用结构化语法形式，支持大型知识系统的模块化设计和结构化开发；

(2) 完美地结合了逻辑程序描述性语言的优点和面向对象范例的典型概念；

(3)对两种程序设计风格探讨了统一的逻辑语义。这也有助于研究面向对象范例本身的语义基础；

(4)元类结构允许用户定义目标类的表示结构和计算方法。这有利于支持多模式正交知识处理系统的开发。

为了提高元级解释的效率，我们还将部分计算技术[5]应用于元解释器，并获得良好的效果。

参 考 文 献

1 Goldberg A, Robson D. Smalltalk－80: The Language and its Implementation. Addison Wesley, 1983

2 Meyer B. Genericity versus Inheritance. Proc OOPSLA'86, Sept, 1986

3 Zaniolo C. Object－Oriented Programming in Prolog. Proc International symposium on Logic Programming. Atlantic City, IEEE, 1984

4 Warren David H D. An Abstract Prolog Instruction Set, Technical Note 309, SRI, October, 1983

5 Deng Tieqing, Jin Zhi, Wu Quanyuan. Further Researches on the Partial Evaluation of Prolog Programs. IPMU'90, France, 1990

6 Davison E. Polka: A Parlog Object Oriented Language. Technical Report, Dept of Computing, Imperial College, 1988

7 Stabler E P. Object－Oriented Programming in Prolog. AI Expert, October, 1986

8 Shapiro E, Takeuchi A. Object Oriented Programming in Concurrent Prolog. New Generation Computing, Springer Verlag, 1983;1(1)

9 Mizoguchi F, Ohwaha H, Katayama Y. LOOKS: Knowledge Representation System for Designing Expert System in a Logic Programming Framework. Proc of the International Conf on Fifth Gen Comp Sys, 1984

10 Hu Yunfa, et al. GKD－Prolog/Wick Reference Manual. CIT Technical Report, 1989

11 Floyd J W. Foundations of Logic Programming, Springer－Verlag, 1984

12 Jin Zhi, Deng Tieqing, Hu Shouren. PROLOG Module System－the Compiling Techique Based on the Meta－level Extension. Proc of the third National Conf on Logic Programming and AI programming, Zhang Jia Jie, China, 1991

13 Jin Zhi, Hu Shouren. A New Approach towards Combining Logic－with Object－Oriented Paradigm for Parallel Knowledge Processing. Proc of the Int'l Conf for Young Computer Scientists, Beijing, China, 1991

14 Fukunaga K, Hirose S. An Experience with a Prolog－based Object－Oriented Language. Proc of Object－Oriented Prog Sys, Lang and Applic'86(OOPSLA), ACM Sigplan Notes 21, 1986

15 Kahn K, et al. Vulcan: Logical Concurrent Objects, in Research Directions in Object－Oriented Programming.

MIT Press, 1987
16 Aiello L, Levi G. The uses of meta-knowledge in AI Systems, ECAI-84, Pisa, September, 1984
17 Mello P, Natali A. Programs as Collections of Communicating Prolog Units. Proc of ESOP'86, Springer-Verlag, Lecture Notes in Computer Science 213, 1986
18 Mello P, Natali A. Objects as Communicating Prolog Units, Proc of ECOOP'87, Bigre+Globule 54, 1987
19 Chikayama T. Unique Features of ESP. Proc FGCS'84, ICOT, Tokyo, 1984
20 Yoshida T, et al. A'UM-Parallel Object-Oriented Language upon KL1, ICOT, 1987
21 金芝，胡守仁. 一个新一代知识处理系统原型. 全国人工智能与智能机学术会议论文集. 北京，1991

SCKE: A New Model for Combining Logic with Object-oriented Paradigm

Jin Zhi Hu Shouren
(Department of Computer Science)

Abstract

Nowadays, the paradigms, namely the object-oriented language and the logic programming language, are two attractive knowledge programming languages. However, a generally accepted answer to the question "How to combine the logic-with the object-oriented paradigm for complex knowledge processing?" is still to be given. In this paper, a new model SCKE has been proposed towards combining the logic-with the object-oriented paradigm of computing. It is intended to introduce the concepts that are typical for the object-oriented systems in the logic-oriented paradigm, without losing its advantages as a declarative language. An extensive Herbrand interpretation for SCKE model has been defined to interpret uniformly the logic semantics and the object-oriented semantics. SCKE model has been implemented on SUN separately by the following three methods: precompiling-based meta-level extension, interpreting execution and compiling execution. Finally, the features of SCKE model have been discussed.

Key words logic programming, object-oriented paradigm, model, Herbrand interpretation