18-22, 9 代数规范描述 语义

计算机科学 1994Vol. 21 №. 2

代数规范描述及其语义的研究*)

宋 群 聂承启

（江西师范大学计算机科学系 南昌 330027）

摘 要 In this paper, the theories and the concepts of algebraic specification and its semanteme for specification abstract data type are presented. The semantic basis and semantic meaning of algebraic specification are discussed. In the last part of this paper, some examples of modeling abstract data type defined accurately are given.

关键词 Algebraic Specification, Semantic Function, Algebraic Class Abstract Data type

用代数规范描述来描述抽象数据类型的基本思想是用它的标记和特征性质来说明抽象数据类型，它们的性质可用多类逻辑形式来表示，通常为受限的一价逻辑，例如等式逻辑。但也可以选用其它形式，象高阶等式逻辑等。抽象数据类型被刻画为计算结构的同构类。因此，从数学上来说，代数规范描述的理论就是可达多类代数的同构类的理论。

抽象数据类型规范的方法研究主要有二种：一是从给定的数据类型D着手，去找一个适合D的代数规范描述。二是从一个代数规范SP开始(比如对一些给定的非正式的软件或硬件问题)，试图去获得一个更详细的规范描述SP'来描述一个抽象数据类型。本文中，我们主要兴趣在后一种方法的研究。

1 公式与理论

多类代数的性质是通过多类一阶等式逻辑来表示的。

设$\Sigma=$（S，F）是一个标记，一个Σ-等式具有形式$t=_s t'$，其中t，t' $\in T(\Sigma, X)$是类$s\in S$中的项。（一阶）Σ-公式是Σ-等式由连接词$\neg$，$\wedge$和量词$\forall$，$\exists$构造而成。Σ-公式集WFF（Σ）是满足以下性质的最小集合：

（Ⅰ）每个Σ-等式在WFF（Σ）中；

（Ⅱ）若G，H$\in$WFF（Σ），则$\neg$G，（G$\wedge$H）$\in$WFF（Σ）；

（Ⅲ）若$x\in Xs$且G$\in$WFF（Σ），那么（$\forall$ x：S. G）$\in$WFF（Σ）；

*）江西省自然科学基金资助课题。聂承启 副教授，从事软件工程、知识工程、多处理系统和计算机应用基础的研究。92年在美国Oakland大学研究多处理系统。宋群 讲师，从事软件形式化开发方法的研究，现在德国攻读博士学位。

过程的基点，认识起点不应限制由此展开的认识过程。逻辑程序语义可以作为它所表达的认识进程{P.}实际指称的轨迹——{$M_{P_i}^{*}$}。从这个角度上看，我们更有理由接受那种“可想而不可及”的模型语义定义：这类语义模型（如稳定模型语义、良基模型语义）尽管可能无法有穷（或能行）生成，但确是良定的。（全文完）

参 考 文 献

[1] H. Przymusinska & T. Przymusinski, Semantic Issues in Deductive Databases and Logic Programs. Formal Techniques in Artificial Intelligence, R. B. Banerji (editor), 1990, pp. 321-367.

[2] J. W. Lloyd, Foundations of Logic Programming, Springer Verlag, New York, N. Y., first edition, 1984.

Σ-等式也称原子公式。其它的逻辑算符象∨，⇒和∃ 等常看作是以下缩写：

$(G \vee H) =_{def} \neg(\neg G \wedge \neg H)$,

$(G \Rightarrow H) =_{def} (\neg G \vee H)$

$(G \Leftarrow\Rightarrow H) =_{def} ((G \Rightarrow H) \wedge (H \Rightarrow G))$,

$(\exists x: S.G) =_{def} \neg(\forall x: S.G)$

出现在一个公式 G 中的约束变元集用 BV（G）表示，G 的自由变元集用 FV（G）表示，没有自由变元的Σ-公式称为Σ-句子，无任何变元的Σ-公式称为基础。

一个标记Σ的 Horn 从句是如下形式的Σ-公式：$(G_1 \wedge \cdots \wedge G_n) \Rightarrow G$，其中 $G_1, \cdots, G_n$ 是原子公式且 $n \geqslant 0$，因为原子公式就是Σ-等式，因此 Horn 从句也叫条件Σ-等式。

对任意Σ-代数 A，赋值 $V: X \rightarrow A$ 和Σ-公式 G，在 V 上 A 满足 G 的关系记为 $A, V \models G$，它根据以下归纳定义为：

（Ⅰ）$A, V \models t =_s t'$ 当且仅当 $V^*(t) = V^*(t')$；

（Ⅱ）$A, V \models \neg G$ 当且仅当（$A, V \models G$）不成立；

（Ⅲ）$A, V \models (G \wedge H)$ 当且仅当（$A, V \models G$）和（$A, V \models H$）；

（Ⅳ）$A, V \models \forall x: S.G$ 当且仅当，对所有的赋值 $V_x: X \rightarrow A$ 且 $V_x(y) = V(y), \forall y \neq x$ 有（$A, V_x \models G$）。

若$\forall V: X \rightarrow A$，$A, V \models G$，则说Σ-代数 A 满足 G，记为 $A \models G$。

一个Σ-公式 G 的全体闭合是Σ-句子 $\forall x_1: s_1, \cdots, x_n: s_n.G$，其中 $FV(G) = \{x_1: s_1, \cdots, x_n: s_n\}$。很显然，Σ-代数 A 满足 G 当且仅当它满足 G 的全体闭合。在一个Σ-代数类 K 中，若任何 $A \in K$ 均满足 G，则称Σ-公式 G 在 K 中是有效的，记作 $K \models G$。

在同构下满足关系是封闭的：若Σ-代数 B 与 A 同构，则 B 满足Σ-公式 G，当且仅当 A 满足 G。记 A 的同构类为［A］，则［A］$\models$ G 当且仅当 $A \models G$。一个Σ-等价$\sim \in C(\Sigma)$满足 G 当且仅当在所有～相关的Σ-计算结构中 G 成立。每一个Σ-代数类 K，有一个有关在 K 中有效的Σ-公式集 Th（K），称为 K 的理论。我们称 $Th_{EQ}(K)$ 为 K 的等式理论（即 K 中有效的全体Σ-等式集）。同样可定义 K 中有效的条件Σ-等式集 $Th_{CEQ}(K)$。

另一方面，每一个标记Σ和一个Σ-公式集 E，也有一个有关满足 E 的所有Σ-代数类 $Alg(\Sigma, E) =_{def} \{A \in Alg(\Sigma) \mid A \models G, \forall G \in E\}$。类似可以定义 $Gen(\Sigma, E) =_{def} \{A \in Gen(\Sigma) \mid A \models G, \forall G \in E\}$ 以及相对于 Gen（Σ，E）的抽象数据类型集 $C(\Sigma, E) =_{def} \{\sim^A \mid A \in Gen(\Sigma, E)\}$。在这种情形下 E 就分别称为 Alg（Σ，E），Gen（Σ，E）和 C（Σ，E）的公理集或公理化。

显然公理化不唯一，对任意Σ-公式集 E 有 $E \subseteq Th(Alg(\Sigma, E))$ 和 $E \subseteq Th(Gen(\Sigma, E))$ 成立。很容易看到：任意二个Σ-代数类 K，K′ 和Σ-公式集 E，E′ 有：

（＊）$K \subseteq K' \Leftarrow\Rightarrow Th(K') \subseteq Th(K)$（同构意义下）

（＊＊）$E \subseteq E' \Leftarrow\Rightarrow Alg(\Sigma, E') \subseteq Alg(\Sigma, E') \Leftarrow\Rightarrow Gen(\Sigma, E') \subseteq Gen(\Sigma, E) \Leftarrow\Rightarrow C(\Sigma, E') \subseteq C(\Sigma, E)$

且对任意Σ-公式集 E，我们有：

$E \subseteq Th(Alg(\Sigma, E)) \subseteq Th(Gen(\Sigma, E)) = Th(C(\Sigma, E))$

Alg（Σ，E）有一个重要性质，即它的理论可通过一个形式系统产生出来，即存在一个逻辑和非逻辑公理集和（有限）推理规则集，它们定义了一个二元关系 $E \vdash G$，这个关系成立当且仅当Σ-公式 G 可以利用逻辑和非逻辑公理以及推理规则由 E 推导出来。

“＝”作为非逻辑公理，它的反身性、对称性和传递性以及替换性质如下：

（Ⅰ）反身性：$\forall x: S.X =_s X, \forall s \in S$；

（Ⅱ）对称性：$\forall x, y: S.X =_s Y \Rightarrow Y =_s X, \forall s \in S$；

（Ⅲ）传递性：$\forall x, y, z: S.X =_s Y \wedge Y =_s Z \Rightarrow X =_s Z, \forall s \in S$；

（Ⅳ）替　换：$\forall x_1, y_1: S_1, \cdots, x_n, y_n: S_n, X_1 =_{s_1} Y_1, \cdots, X_n =_{s_n} Y_n \Rightarrow f(x_1, \cdots,$

x_n) $=f$ (Y_1, …, Y_n) $\forall f: S_1, \cdots, S_n \to s \in F$。

课题研究中，完全性理论是我们要求的重要结果之一。限于篇幅我们省略了证明过程，详细证明请参考文 [2]。

定理 1.1 一阶Σ-公式 G 可由Σ-句子集 E 推导出来，当且仅当 G 在 Alg(Σ, E) 中是有效的，即：$E \vdash G$ 当且仅当 Alg (Σ, E) $\models G$。

类似对于等式和条件等式公式也存在通过二元关系 $\vdash_{EQ}$和 $\vdash_{CEQ}$计算理论 Th_{EQ} (Alg (Σ, E)) 和 Th_{CEQ} (Alg (Σ, E)) 的形式系统。

Σ-等式的推理规则有二条：设 H 为（无量词）条件Σ-等式，G_1, G_2, G 为Σ-公式，u, u′, v, v′ $\in T$ (Σ, X)：

（Ⅴ）替换规则：若 H 是可导的，$x \in X_s$，$u \in T$ (Σ, X)$_s$，则 H [u/x] 可导；

（Ⅵ）切换规则：若 $G_1 \wedge u' = v' \wedge G_2 \Rightarrow u = v$ 是可导的且 $G \Rightarrow u' = v'$ 可导，则 $G_1 \wedge G \wedge G_2 \Rightarrow u = v$ 可导。

如果Σ-公式 G 可用公理 (i) — (iv) 和推理规则 (v) — (vi) 导出，就记为 $E \vdash_{EQ} G$，对于可见标记，这个形式系统是充分和完全的。为此我们阐明有关多类等式演算充分性和完全性的定理如下：

定理 1.2 设Σ是可见标记，E 为条件Σ-等式集，则 $E \vdash_{EQ} G$，当且仅当 Alg (Σ, E) $\models G$

［证明提示］ 根据Σ为可见标记的假定，并通过验证逻辑公理和推导规则 (i) — (vi) 的有效性易证充分性。为了证明完全性，我们考虑项代数 T (Σ, X) 的商代数 T (Σ, X) $/\sim^E$：X 是一个 Sort (Σ) 一类自由变元集，它的每个类都有无限个变元且$\sim^E$是 T (Σ, X) 上的等价关系，这个关系定义为：$\forall t, t' \in T$ (Σ, X)，$t \sim^E t'$ 当且仅当 $E \vdash_{EQ} t = t'$。商代数 T (Σ, x) $/\sim^E$ 是 Alg (Σ, X) 中的自由代数，即所有Σ-等式 $t = t'$，T (Σ, X) $/\sim^E \models t = t'$ 当且仅当 Alg (Σ, E) $\models t = t'$。这就可证 $\vdash$EQ 的完全性。

设 G 是Σ-公式，FV (G) $= \{x_1 : s_1, \cdots, x_n : s_n\}$，若 $t_i \in T$ (Σ)$_{s_i}$为基本项，$i = 1, \cdots, n$，则 G $[t_1/x_1, \cdots, t_n/x_n]$ 称为 G 的一个基本例示，为此有：

结论 1.3 设 E 为无量词Σ-公式集。

(1) Gen (Σ, E) $=$ Gen (Σ, E_{ground})，其中 $E_{ground} =_{def} \{G[t_1/x_1, \cdots, t_n/x_n] \mid G \in E$, FV (G) $= \{x_1 : s_1, \cdots, x_n : s_n\}$, $t_i \in T(\Sigma)_{si}$, $i = 1, \cdots, n\}$。

(2) 任意Σ-基本项 u, $v \in T$ (Σ)：Gen (Σ, E) $\models u = v$ 当且仅当 $E \vdash u = v$。

［证明提示］(1) 显然，一个Σ-计算结构满足无量词公式 G 当且仅当它满足 G 的所有基本例示；(2) 因为 $u = v$ 为基本等式，由 (1) 有 Alg (Σ, E) $\models u = v$ 当且仅当 Gen (Σ, E) $\models u = v$，再根据定理 1.2 便可直接推出。□

2 代数规范描述及其语义

一个规范是一个数据类型 D 的形式描述，它能够保证 D 实现的性质，从而可应用于 D 上程序的正确性证明。一个一阶公理规范 (Σ, E) 由标记和描述所要求性质的一阶公理组成，它的语义由标记Σ和 Alg (Σ, E) 给出。公理化实现的思想与数据类型自由表达的观点是相吻合的。但 Alg (Σ, E) 中大多数模型是不可达的，它们对抽象数据类型无任何意义。因此，对抽象的代数规范描述，我们将 Alg (Σ, E) 中的 Gen (Σ, E) 作为语义研究的基础。在 Gen (Σ, E) 中分别有三种主要的探讨：行为语义的初始代数探讨和终止代数探讨，以及自由代数探讨。

在自由探讨中，规范被看作是一种约束，这种约束可能针对许多不同的抽象数据类型。因此它适用于程序证明和程序开发，所有满足公理 E 的Σ-计算结构的类 Gen (Σ, E) 作为它的语义。然而，为了能够用一些公理规范精确地定义抽象数据类型，常需要有一个语义机构 M，M 从 Gen (Σ, E) 中选取一个计算结构 M (Σ, E)，它在同构下唯一。

初始代数探讨中的初始代数和终止代数探讨中的终止代数就是所谓的 M。

一个（简单）代数规范 SP=〈Σ，E〉由标记Σ和Σ-公式集 E 构成。若 E 是(条件)Σ-等式（或句子），则称为（条件）等式规范（或一阶规范）。

三种探讨中的三个语义函数定义为：

Mod (SP) $=_{def}$Gen (Σ，E)，自由语义；

I (SP) $=_{def}$ {I∈Gen (Σ，E) |I 是 Gen (Σ，E) 中初始元}，初始代数语义；

Z (SP) $=_{def}$ {Z∈Gen (Σ，E) |Z 是 Gen (Σ，E) 中的终止元}，终止代数语义。

因此，在三种探讨中，一个规范的语义是抽象的，它是在同构下封闭的计算结构类。Σ-等价与同构类之间的 1-1 对应可以归纳为等价的描述：

$\mathrm{Mod}_{\cong}(SP)=_{def}\{[A]\mid A\in \mathrm{Mod}(SP)\}$

结论 2.1（语义等价）设 SP=〈Σ，E〉和 SP′ =〈Σ′，E′〉是二个（简单）规范。则以下断言等价：

(1) Mod (SP) ⊆Mod (SP′)，

(2) $\mathrm{Mod}_{\cong}$ (SP) ⊆$\mathrm{Mod}_{\cong}$ (SP′)，

(3) C (SP) ⊆C (SP′))，

(4) Th (Mod (SP)) ⊇Th (Mod (SP′))。

[证明提示] (1) ⟺ (2) 因为在同构下 Mod (SP) 的封闭性；(2) ⟺ (3) 因为 $\sim^{T(\Sigma)}$和$\sim^{U(\Sigma)}$分别为 C (Σ) 的最小和最大元素； (3) ⟺ (4) 根据第一节中断言 (* *) 易得。□

若Σ=Σ′ 且 Mod (SP′) ⊆Mod (SP)，记为 SP′ ⊆SP，则规范 SP′ =〈Σ′，E′〉就称为规范 SP=〈Σ，E〉的求精。如果 SP′ ⊆ SP 且 SP⊆SP′，则称 SP 与 SP′ 等价。

初始和终止计算结构并非总是存在，即使它们存在也常不同构。但如果它们是同构的话，那么 Mod (SP) 就仅为一计算结构的同构类，即只有一个抽象数据类型，且这时称规范 SP 为单构的。

一般地，初始和终止计算结构可刻画如下：

引理 2.2 设 K 是Σ-代数类。

(1) 一个Σ-计算结构 I∈K 是 K 中的初始元当且仅当对所有基本Σ-项 t，t′ ∈T (Σ) 有以下条件成立：I⊨t=t′ ⟺∀ A∈K：A⊨t=t′

(2) 一个Σ-计算结构 Z∈K 是 K 中的终止元当且仅当对所有基本Σ-项 t，t′ ∈T (Σ) 有 Z⊨t=t′ ⟺∃ A∈K：A⊨t=t′。

[证明提示] 根据⊨的定义，(1) 和 (2) 分别是以下结论的直接结果：

$t\sim^{I}t'$ ⟺∀ A∈K：$t\sim^{A}t$，$t\sim^{Z}t'$ ⟺∃ A∈K：$t\sim^{A}t'$。□

作为该引理和结论 1.3(2) 的结果是，在规范 SP 的初等模型中一个等式成立当且仅当这种相等性可用 SP 的公理来"计算"，而在终止模型中仅有那些可以证明它们的非成立的等式才是假的。为此有：

定理 2.3 设 SP=〈Σ，E〉为规范且 E 为无量词公理集。

(1) 一个Σ-计算结构 I∈Mod (SP) 是 Mod (SP) 的初始对象当且仅当∀ t，t′ ∈T (Σ) 有以下条件成立：I⊨t=t′ ⟺$t\sim^{I}t'$ ⟺E ⊢t=t′

(2) 一个Σ-计算结构 Z∈Mod (SP) 是 Mod (SP) 的终止对象当且仅当∀ t，t′ ∈T (Σ) 有以下条件成立：Z⊨t≠t′ ⟺¬ ($t\sim^{Z}t'$) ⟺E⊢¬ (t=t′)

[证明提示] 根据引理 2.2 和结构 1.3 (2) 的结果可得。□

以后我们将用 spec SP≡signature Σ axioms E endspec 来表示规范 SP=〈Σ，E〉。如果 SP_1=〈〈S_1，F_1〉，E_1〉，则通过算子"extend by."来构造更大的规范，spec SP≡extend SP_1 by sorts S functions F axioms E endspec 就表示规范〈Σ，E_1∪E〉且Σ=〈S_1∪S，F_1∪F〉。再设 SP_2=〈〈S_2，F_2〉，E_2〉，则 $SP_1+SP_2=_{def}$ 〈〈S_1∪S_2，F_1∪F_2〉E_1∪E_2〉表示 SP_1 和 SP_2 的联合。

最后，为了具体论述，我们给出一些实例，并通过这些实例来了解有关语义。在下

面的实例中出现的Σ_{BOOLO}，Σ_{NATO}，N-B，PO 和 NO* 等，它们的定义在文[6]中已有说明。例：

（1）规范 BOOLO 由Σ_{BOOLO}构成，没有公理。

spec BOOLO≡signature Σ_{BOOLO} endspec。

Mod（BOOLO）中含有二个抽象数据类型：项代数 T（Σ_{BOOLO}）的抽象数据类型和单位代数 U（Σ_{BOOLO}）的抽象数据类型，显然它们分别为 BOOLO 的初始和终止模型，即 Mod（BOOLO）=I（BOOLO）∪Z（BOOLO）。如果在 BOOLO 中加入一个不相等公理 true≠false，我们就得到一个单构规范 BOOLM，它以 T（Σ_{BOOLO}）的抽象数据类型作为模型：

spec BOOLM ≡ signature Σ_{BOOLO} axioms true≠false endspec

这样，Mod（BOOLM）=I（BOOLM）=Z（BOOLM）=I（BOOLO）。

还可以使用普通布尔算子 not，and 和 or 对 BOOLM 进行扩张，得到规范 BOOL：

```
spec BOOL≡
  extend BOOLM by
    functions not: bool→bool
    .and., .or.: bool, bool→bool
    axioms not (true) =false, not (false) =true,
           true and x=x, false and x=false,
                 x or y = not (not (x) and not
                 (y))
endspec
```

规范 BOOL 是单构的。BOOL 的任一模型 M 的限制 M|Σ_{BOOLO}是 BOOLM 的模型，即 M|Σ_{BOOLO}≅T（Σ_{BOOLO}）。

（2）spec NATO≡signature Σ_{NATO} endspec

Mod（NATO）就是 Gen（（Σ_{NATO}，Φ）），I（NATO）是 T（Σ_{NATO}）的抽象数据类型也就是（等同于）N；Z（NATO）是Σ_{NATO}的单位代数的同构类。

对 BOOLM 和 NATO 进行适当地完善，可以得到一个单构的规范，N-B 可作它的规范代数（模型）：

```
Spec NAT1≡
  extend BOOLM+NATO by
    functions eq: nat, nat→bool
    axioms    eq(Succ(x),Succ(y))=eq(x,y),
              eq(Succ(X),zero)=false,
              eq(zero,Succ(X))=false;
              eq(x,x)=true
endspec
```

有 Mod（NAT1）=I（NAT1）=Z（NAT1）=[N-B]

（3）在下面定义的规范 LOOSE-SET 中，自然数有限集的计算结构 PO 是它的终止代数：

```
Spec LOOSE-SET≡
  extend NAT1 by
  Sort,set
  functions emty:→set
    insert:nat,set→set
    .ε. :nat,set→bool
  axioms(x ε emty)=false,
    (x ε insert(x,s))=true,
    eq(x,y)=false⇒(x ε insert(y,s))=(x ε s),
endspec
```

而 NO*（自然数有穷序列）的抽象数据类型为 LOOSE-SET 的初始元素，即 I（LOOSE-SET）=[NO*]，Z（LOOSE-SET）=[PO]。

但 Mod(LOOSE-SET)包含无限个抽象数据类型。事实上 Mod(LOOSE-SET)的同构类集不可数。为了说明这一点，我们考虑 PO 的载体集 PO_{SET}的划分。设 P：PO_{SET}→{0,1}为全函数；在$\Sigma_{SETNAT}\setminus\{\varepsilon\}$项集中定义一个等价关系$\sim^P$：$\forall$ t,t′ ∈T($\Sigma_{SETNAT}\setminus\{\varepsilon\}$)$_{set}$，

t $\sim^P$t′ 当且仅当 t≡t′ 或(PO⊨t=t′ 且 P(t^{PO})=1)

按照 LOOSE-SET 的公理集，由定义"n ε t=b"(n∈T(Σ_{NAT1})，t∈T(Σ_{SETNAT}))$_{SET}$，b∈{true,false})，关系$\sim^P$ 可以扩张为 LOOSE-SET 的一个一致关系，且存在无穷个划分 P：PO_{SET}→{0,1}。因此 Mod(LOOSE-SET)不可数。

在 LOOSE-SET 中加入二个公理，使得含 nat 中相同元素的项成为一致，就可得到一个单构规范：

```
spec SETNAT≡
    extend LOOSE-SET by
    axioms insert(x,insert(x,s))=insert(x,s)
      insert(x,insert(y,s))=insert(y,insert(x,s))
```

（转第 9 页）

一致和非矛盾的能被证据数据等最好地解释的假设集。据 Thagard[10]所言："一个假设和命题集相合适（贴切）是指假设解释命题集或命题集解释假设，或两者共同解释其它的命题，或者命题集提供相似的解释"。用以描述直接观察到的事物的命题具有独立的可接受性，而一个解释假设被接受仅当它在总体上比竞争命题具有更好的"合适性"。

Thagard 理论本质上由建立命题间"合适性"的七个关系构成，这些关系为：对称性、解释、相似、数据优先权、矛盾、可接受性和系统合适性。下面对"解释"和"相似"加以更为详细的介绍以便给出"合适性关系"的一个简单实例，有关完整的描述请参见文 [10]。

Thagard 对"解释"理论描述如下：

如果 P_1，…，P_m 解释 Q，那么（1）对任一 P_i，$P_i=P_1$，…，P_m，P_i 和 Q 相合适；（2）对任一 P_i，P_j，$P_i=P_1$，…，P_m，$P_j=P_1$，…，P_m，P_i 和 P_j 相合适；（3）在（1）和（2）中，合适的程度反比于命题 P_1，…，P_m 的数目。

对"相似"理论描述如下：

（1）如果 P_1 解释 Q_1，P_2 和 Q_2，P_1 相似于 P_2，Q_1 相似于 Q_2，那么 P_1 和 P_2 合适，Q_1 和 Q_2 合适；

（2）如果 P_1 解释 Q_1，P_2 解释 Q_2，Q_1 相似于 Q_2，但 P_1 不相似于 P_2，那么 P_1 和 P_2 不合适。

上述的这类关系如何编码到一个 NN 模型中，这样的 NN 模型如何运行呢？首先，须有一个简单的高级描述语言，能允许诸如（EXPLAIN (H_1，H_2) E_1）和（CONTRADICT (H_1) (H_2)）形式的表达式（该表达式表示假设 H_1 与 H_2 共同解释证据 E_1，但 H_1 和 H_2 互相矛盾）。这些表达式被编辑在一个 NN 网络中，每一个命题用一个单一的节点表示。如果两个命题合适，那么在它们之间存在一个表示兴奋、具有正权值的对称连接。如果两个命题不合适，那么它们之间存在一个抑制连接。数据优先权由来自一个特殊数据节点的兴奋刺激链来实现。

在时刻 t，激活值大于零指示命题的接受，整个命题系统的合适性用下列函数表征（这个函数是 NN 系统中能量函数的负值）：

$$H(t)=\sum_i\sum_j W_{ij}a_i(t)a_j(t)$$

其中 W_{ij} 是从节点 i 到节点 j 的权值，a_i（t）是时刻 t 节点 i 的激活值。运行网络意指去产生命题的一个合适集（假如这个集合是可得到的），这个集合由命题节点上激活值的一个稳定模型来表示，这个稳定的结合体比其它具有更糟解释合适性的可能假设集拥有更多的控制权力。Thagard[10]使用这种方法进行了科学解释和论证领域里的很多模拟实验。例如：Lavoisier 用氧气来反驳燃素理论的论点。至今为止，为了得到一稳定结果，最为复杂的应用包含 150 个节点和 210 次迭代过程。

（参考文献共 10 篇略）

（接第 22 页）

有 Mod (SETNAT) ＝I (SETNAT) ＝Z (SETNAT) ＝ [PO]。

初始代数规范一般比终止代数规范和自由规范需要的公理数要少。初始代数规范的优越性在于它们为真（条件）等式规范，而终止代数规范和单构的自由规范则需要附加的非等式公理或附加的语义假设，这方面的探讨可参考有关文献。在以后的有关代数规范描述的应用中，我们将总是假定规范 SP 是 BOOLM 的完善，即 true≠false 是 SP 仅有的非等式，所有其它的公理都是（条件）等式公式。

参 考 文 献

[1] J. A. Goguen, J. Meseguer,《Completeness of Many-Sorted Equational Logic》, 1981

[2] K. J. Barwise,《Studies in Logic and the Foundation of Mathemaics》Vol. 90, 1977

[3] P. Padawitz, M. Wirsing,《Completeness of Many-Sorted Equational Logic Revisited》, 1984

[4] P. Padawitz,《Computing in Horn Clause Theories》, Theoretical Computer Science 16, 1988

[5] M. Wirsing,《Algebraic Specification》, MIP-8914, 1989

[6] 宋群，聂承启，杨茵，《抽象数据类型的代数方法研究》，《江西师大学报》3 期，1993