计算机科学1992 Vol.19 No.5

逻辑中的元级概念

王献昌*)

"道"可道，非常"道"
"名"可名，非常"名"

——引自：老子《道德经》**)

摘　要

语言和思维的主要作用是将主观规定的性质归于事物，然后断言之。对语言和思维的这一研究构成了元语言的基础。属性本身亦是一类事物，故也可以把元语言看作是对象语言。这个事实具有重要的影响，并成为逻辑和AI领域内许多研究的基础。本文将综述这方面工作。

一、引　言

元级指"关于（aboutness）"，即从某物回顾，观察该物的内在因果性，并对该物及其内在因果性作出断言。元级层的概念经常出现在形式逻辑领域中，借助元级语言，可以表达逻辑推导和形式证明。元级语言既可以是自然语言，也可以是形式语言，甚至可以是对象语言本身。当元语言与语言相同时，术语反射（reflection）看起来更为贴切，因为这具有一定自处理（self-dealing）、自省（introspection）的味道。本文的大部分内容将说明如何获取这种自省能力和各种方法所存在的问题。

特别地，一个合式公式（wff）的可证明性等同于对象（该wff）的一种属性。尽管合式公式有其它的属性，如可信任，必要性，但我们将主要讨论可证明性。

假定T是语言L上的理论（为便于说明，我们取L、T为一阶语言），于是合式公式P(a)直观地解释为："P陈述了事物a的某事"。这样L自身已至少有某种"谁有关谁"的含义，即我们认为P是谈论a的，故它处在a所处领域的元级层次上。确实，传统的解释I支持上述观念：a解释为I中论域的元素，而P为论域上的关系。为了更准确地表达事实：P是用来规定事物a的一种属性，我们用Attrib（P，a）来表示。现在P的作用发生了变化：P不再是一个谓词，而是一个参数或项了，即P从"谈论a的某属性"变为被谈论的对象，故更合适的写法应当是用"'P'"而不是P来描述。即Attrib（P，a）应当写为Attrib（'P'，a）。显然'P'不同于P，'P'是不同于P的符号（称为P的名）并通过某些公理先验地和P建立了联系。一般说来，P的名字'P'的引入称为P的实体化（reification，源于拉丁语，意为thing，即事物），P变为一个被用于讨论的项'P'。

既然Attrib（'P'，a）是用来谈论P(a)的，Attrib（'P'，a）一定比P（a）具有更强的元级含义。Attrib可以是语言L的符号，

*）现为北京航空航天大学博士后。　**）编译者借以说明AI的理想与实现之间的矛盾。

也可以不属于L。但一般说Attrib 属于元级语言M，这里并不排除M＝L。

元语言的研究历史始于 Hilbert 的规划：对于算术的形式化以及证明这种形式化的若干定理，这些定理是有关形式证明和逻辑结论的断言，故都是些元定理。第一个将元定理与自身语言L结合在一起并获得成功的是Godel〔1931〕，其关键的一步便是用'P'来表示P。看来AI 中围绕使用元语言而引起的困难主要取决于名字的引入，而不是引入可证明性这些形式概念。

元语言M中的一个典型命题是：

L的合式公式α在理论T中是可证明的。

这里M是汉语，L是一阶语言。一般记上述断言为"T⊢α"，但T⊢α并不是一阶语言L中的公式，而只不过是上述命题的一种简写罢了。

Godel 决定在语言L中引入一个新的谓词符号 Thm 来表达⊢。从而导致L的公式含有 Thm('α')，这里'α'为L的合式公式α的名字，故'α'必然属于L。名字的加入导致了L中更多合式公式的出现，反过来又需要给这些新的合式公式规定名字，因此必须借助递归方式。

在L内通过"命名"来表达L中元概念的过程称为元数学的算术化。之所以这样讲是由于 Godel 用'P'命名表达式P的实质是给P赋一个整数，称为P的 Godel 数。有关算术的公理则用于研究L中表达式的证明性质。因此依据L中的Thm，T中可证明性概念便得到了一定的形式表达。

但 Thm 能完全表达理论T的可证明性吗？若是，显然希望P在T中的可证明性与 Thm('P') 在T中的可证明性是一样的。故

T⊢P　iff　T⊢Thm('P')

这就是 Godel 给出的有关 Thm 的定义。该定义可以用二条推理规则来表示：

P ⊢ Thm('P')　　和

Thm('P') ⊢ P

称上述二规则为 Thm 的正规则。在第三节我们将看到有关 Thm 的情况并非这么简单。

本文是这样安排的：在第二节研究反射问题中的正问题，在第三节研究反射问题中的负问题；元级表达可证明性概念遇到了来自上述二方面的困难，将涉及半可判定性和半不可判定性。在某种意义下，我们将给出一些解决办法。在第四节，我们站在更高的层次上用形式体系的环境观点（即 engine（引擎）观点）来展望逻辑和元级。我们将谈及情态逻辑，看来能表达内构性质的元级特征。最后，讨论情态逻辑是如何处理意义（meaning）的。

二、反射中的正问题

主要困难是如何建立有关符号和名字间合适的对应关系。我们已经看到在 Attrib 和 Thm 中引进名的必要；也看到尽管Thm试图描述它与可证明性间的关系，但并没有强到足以把握算术中的真句子。Thm能完全把握可证明性吗？在讨论这个问题之前，让我们进一步研究符号及其命名的更深刻含义。

最直观的建议是：不应将P与'P'的关系看作为 Thm('P')表达了P的可证明性，而应将P与'P'的关系看作为：P的意义是由'P'给出的。一个熟悉的例子是：

'雪是白的'为真 iff 雪是白的　或

'雪'意指　雪

与特定的理论T无关，语言L应具备如下约定的解释：'P'为P的名字仅当它们之间有某种形式联系，而谓词'True'提供了'P'与P联系的纽带。

True ('P') ⟷ P

Believes ('P') ⟷ Thm ('P')

Necessary ('P') ⟷ …

Probable ('P') ⟷ …

Consider ('P') ⟷ …

Should-do ('P') ⟷ …

Evase ('P') ⟷ …

Introspect-for ('P') ⟷ …

等。

上述这类概念经常出现在有关元级体系和反射的文献中。除了给出 True, Believes 的定义外，我们尽可能避开对这些概念进行公理化定义。注意，我们采用了Konolige〔1982〕对Believes 的定义，即一个智能体 (agent) 相信P的含义是指该智能体能证明P。

现在我们分析上述定义所存在的问题。首先是 Tarski〔1936〕以说谎者悖论 (Liar Parodox) 的方式提出的，Tarski 指出，谓词 True 的定义：

True ('P') ⟷P

在包含有充分的算术理论T中是不协调的！这令人惊诧，并导致了人们对 AI 基础的疑惑。因为一个智能体知道某良构式α的含义是基于α为真这个明显的直觉之上的。事实上，α的知识也经常被看作为α是已被验证的真信念。但正如 Gettier 在1963年所发现的，这种观点也成问题。

对 Tarski 形式的一种简单变形将导致T的一致性，这是由 Ferferman 与 Perlis 分别在1984年1985年发现的。

True ('α') ⟷ α*

这里*是公式的一种变形操作，与Gilmore 在1974年给出的操作相似，*将所有形如¬True ('β')的子公式都用True ('¬β')代替，直到不出现形式为¬True ('β')的子公式为止。

对一个理性的智能体进行形式化描述的另一问题是由 Montague〔1963〕发现的。若Bel 代表信念，公理格式为：

Bel ('α') ⟶α

推理规则为：

α ⊢ Bel('α')

同样在充分的算术理论中是不相容的。尽管有一些乐观的结果，但上述事实无疑对知识的形式化表示产生了消极影响。首先，让我们讨论 Montague 理论是如何成为一个有关信念的"更合理"的形式理论的。这涉及到二种观点。一种观点是，语言L及理论T都是智能体所拥有的，故每一公理、定理都应是智能体所相信的信念，而 Bel 则是智能体"反省"其自身信念的一种途径；另一观点是：语言L和理论T是我们分析智能体的工具，故T是描述智能体的理论，T的定理是可以建立的事实，Bel('α')则是表达智能体相信α的一种途径。下面所讨论的内容适用于上述二种观点。

设站在我们面前的是一个理想的智能体R，假定R有自己的谓词 Bel 以记录某些表达式是可信的。由于R是理想的，故R不应当发生差错，即R只可能相信真的命题；由于R是理想的，故R是否也应当知道（或相信）这一点呢？即R应当有这样一个关于自身推理的事实，"它所相信的都是真的"，这就是 Montague 的第一公理：

Bel ('α') ⟶α

但这和 Bel 的解释：Bel ('α') ⟷ Thm ('α') 相悖。因为 Lob〔1955〕证明，Thm ('α') →α是给定理论T（其中 Thm为Godel谓词）的定理 iff α在T中是已可证明的，所以一般来讲，不能有Thm ('α') → α，即不能将 Thm 与 Montague 公理格式中的 Bel 等价看待。但 Montague 的公理格式似乎又是真的，故需要进一步分析Montague 的推理规则 α / Bel ('α') 的含义。

该推理格式解释为：若能证明α，智能体R则一定相信α。该推理规则看起来也是无害的，即不会导致矛盾。但它和 Montague 公理一起，却将我们引入了一个尴尬的境界——我们所希望的理想智能体是不存在的。

但若用 Bel α 代替Bel ('α')，对Montague 的公理格式和规则作相应替换后，所得出的模态理论便一致了。这是否能说明模态逻辑而不是一阶逻辑能表达一个理想的智能体呢？

首先，若放弃对智能体某种理想的标准，则建立的这种形式体系便无法令人信

服。无论如何，Montague 体系中的问题表明，智慧的一个显著特征是对错误性的察觉，即它能意识到它所相信的未必是真的，故我们倾向于采用 Montague的负公理格式：

$$(\exists x)\ (Bel(x) \wedge \neg True(x))$$

至于模态逻辑，des Rivieres 和Levesque〔1986〕证明，对一致的模态理论作某种转换，可使原理论在一阶理论中得到表达。尽管这种转换以牺牲直观为代价，但它说明在表达元级方面，模态并不比一阶走多远。

其次，就一阶逻辑本身而言，Perlis〔1987〕证明，若将 Feferman 对 True 处理的方法施用于 Bel，则也可以消除 Montague 理论的矛盾。Asher 和 Kamp〔1986〕也分析了上述问题，但方法却基于 Gupta〔1982〕和 Herzberger〔1982〕对 Truth悖论的研究。

与 Montague 悖论相类似，Thomason〔1980〕给出另一悖论：若一个智能体R相信一个合适的算术理论，且R的信念满足如下条件：

1. $Bel('\alpha') \longrightarrow Bel('Bel('\alpha')')$
2. $Bel('Bel('\alpha') \longrightarrow \alpha')$
3. 对任意有效的α，$Bel('\alpha')$
4. $Bel('\alpha \rightarrow \beta') \rightarrow (Bel('\alpha') \rightarrow Bel('\beta'))$

则在R相信所有的合式公式意义上，R是不一致的。这说明我们所依据的“直观”在形式框架下是多么不可靠。

对 Thomason 悖论的分析如下：R的第二个公理格式 $Bel('Bel('\alpha') \rightarrow \alpha')$ 使R拥有一个“完美”的信念，由此导致了R产生涉及自身的悖论。

Thomason 和 Montague 的工作告诉我们：不能将某种直观上正确的自我推理卷入语言本身。“如果R是相当聪明的，R就应当明白这一点，R也应当以某种方式表达自身的信念”，这就是 Feferman 和 Perlis 所讨论的悖论产生的实质。

Richard Weyhrauch 指出，一个全能推理者的基本含义还应和有关负自省的假设相一致。若一个合式公式α不是智能体R的定理，则R不仅不相信α，而且也应能证明它为什么不相信α。在有关负自省的讨论中，我们将详细地讨论这个问题。

三、反射中的负问题

再看元谓词 Thm，Thm 试图捕获可证明性。当讨论的是元谓词的否定时，我们称之为“反射中的负问题”。我们不仅希望一个理想的智能体告诉我们什么是它所知道的，也能够告诉我们什么是它所不知道的。前者涉及正例规则：

$$P \mid Thm('P'),\ Thm('P') \mid P$$

这二条规则允许理论确定哪些事实上是定理的实例，但却不能确定哪些不是定理的实例。然而我们希望一个智能体有能力证明一个负的合式公式，如$\neg Thm('\alpha')$，$\neg Bel('\alpha')$，……，这就是 Weyhrauch 所建议的有内省能力的理想智能体的标准。值得注意的是，一致性检查与此类问题相关，因为一致性检查等价于判定特定公式的不可证明性，一个智能体若能通过内省方式得出自身的一致性，将导致负自省问题的部分解决。

一般说来，这类问题是不可判定的。尽管 Thm 的真实例是半可判定的，但却不是完全可判定的。即不能确定什么时候一个合式公式不是定理。其差别在于判定前者的时间是不限界的，在有限时间里我们并不知道所有可能的证明都得到了证实。当然对于持有神喻（oracle）的理想推理者，这将不成问题。我们的兴趣则在于寻找好的计算性质。

AI领域中负反射的例子有：

$\neg Bel('\alpha') \longrightarrow Bel('Bel('\alpha')')$ 负自省

$\vdash\!\!\!\!\!\!\backslash\ \alpha \Rightarrow \vdash \neg\alpha$ 失败视为否定规则

$\alpha : Consis\beta / \beta$ 默认规则

$\alpha \rightarrow Bel('\alpha') : Consis(\neg Bel('\alpha'))$
/¬ α自知规则

但如何构造¬Bel ('α') 呢？一个有一线希望的技术是由McCarthy (1980, 1984) 给出的，称为Cir (Circumscription) 技术。通过与语法相关的内部模型，Cir给出了一致性检查的捷径。由于是半可判定的，故 Cir 还强不到产生所有负自省的能力。但它已能捕捉相当一类有趣的常识处理，尤其是我们希望通过对 Bel 进行约束得出¬Bel 来。Cir 的优点是：在大多情况下，它具备 Moore 在 AE (Autoepistemic，自知) 推理中所提出的可废除能力 (defeasiable)；若 Bel ('α') 得不到证明，则承诺¬Bel ('α') 将是安全的。大多文献所讨论的智能体都具备这个性质。

四、情态逻辑 (situated logic) 中的问题

首先看二个断言：

Logic does nothing
Engines do all

上述断言有如下二重性：

(1) 经常说逻辑能（或不能）做这个或那个，而实际上"做"是引擎 (engin) 的职能，却不是逻辑语言的职能，引擎可以使用逻辑。

(2) 当引擎使用逻辑时，它总是在包括控制策略和外部事件的某实时环境下工作。对逻辑的应用总要依赖于引擎的情态，借助这种情态，逻辑才能与真实世界发生联系。

在 Kowaski 1979年提出的"算法＝逻辑＋控制"（中译文见《计算机科学》№.5, 1980)中，"做"是控制机制的任务，逻辑则提供这种机制所使用的素材。尽管控制机制的设计并不一定和逻辑有必然联系，但我们对二者间所具备的关系却感兴趣。特别是那种对控制的形式描述能使系统推理得以简化的情况，如：有关系统的某些公式或规则将失效；这些公理、规则全有效；判定它包含矛盾将花费很多时间；判定它包含矛盾将是不可能的；……。可以看到，有关自省的思想大多产生于控制领域中。

"逻辑＋控制"的提法已有一定历史，上述论述也不新奇，因为只有嵌入到这种系统中时，逻辑及其形式定理才能成为推理系统的元素。这种思想在〔McCarthy和Hayes 1969〕，〔Green 1969〕有关情态逻辑的论述中，在〔Newell 和 Simon 1972〕的产生式系统中，在〔Fagine 和 Halpern 1985, Levesque 1984, Drapkin 和 Perlis 1986〕的限界推理中是自明的；同样在默认推理中也是内含的，因为默认理论的本质是在有穷知识之上对真正矛盾的假设进行检查。近年来颇受重视的元知识也适用于引擎的情态，这是很多智能行为的关键。有关情态的一个很好的例子是由 Drapkin 和 Perlis在1986年讨论的，最初源于〔McDermott 1982〕，称为 Dudley 和 Nell 问题：

Nell 被绑在铁轨上，面临被迎面开来的火车碾死的危险。Dudley 必须拯救她。问题的关键是，Dudley 必须意识到在作规划的过程中，Nell 将越来越危险。但 Dudley 并不能料到大多情况下已经想像出的结果。

总之，情态逻辑在 AI 领域中将发挥越来越重要的作用。

五、有关"关于"的问题

"反射"的涵意之一是有关问题符号处理的语义内容之再现，即存在一个被反射的对象，它通过某些符号获得了问题的意义(meaning)。如何获得意义构成了"关于 (aboutness)"或"引用 (reference)"中的主要问题。简言之，一个程序是怎样将意义赋于符号的？若能建立这样的程序，我们便可以说此程序反映了较高的水准。

有关"意义"的论述很多，大多是由哲学家写的，至少有三个阵营。一些二元论者如 Popper〔1985〕认为，意义并不全是一套概念机制，而是一种关于物理行为的双重现象，因此在一般的科学讨论中是难以被理解的。一些机能主义者如 Dennett〔1978〕认为，

意义仅仅是一个有用的术语，一个智能体使用意义去推断另一智能体的功能。一些经验论者如 Thagard〔1986〕则认为有可能存在一种内在的现象能明确而清晰地产生有关智能体的真实的语义属性；但如何寻找这种特殊的语义属性却有待研究，这种观点成为最近 AI 界的一个主要观点。

Sloman〔1986〕和 Steels〔1986〕指出，在计算行为中有一种关于内部符号表示的重要的上下文相关现象，即程序的执行在某个方面是以非常直接的方式与环境事件相联系的，亦即与执行时的硬件和软件本身的内部状态相联系的。这是一种物理现象，因而也是一种环境，尽管相对于计算机而言不是外部的。还有，当从一个存储寄存器或其它（虚拟）内存地址读出（拷贝）一个值的时候，所得拷贝对于原件（original）有一种物理上的关系（同一或等价），于是可称之为是与原件有关的。这样，符号与符号化之间具有通常是微弱的联系，而就这些内部情形而言似乎又要强一些。

注意，Sloman 和 Steels 的看法也可用推理实体来陈述，该实体自身中既含符号也含符号化的对象（指称）。这表明，显式地引入现象表示也许指首先在事实上生成行为表示。这样，讲"那条件"就把注意力吸引到了一条牛以及对"那条牛"的陈述上。即"牛"既用作一种动物也用作一个字。符号的这种双重作用有什么优点呢？其优点在于对再思能力作出解释，而不丢失最初的观念。人们可能考虑外面那个地方被假设的东西是否真是一条牛，同时清醒地知道只不过用"那条牛"描述过它而已。由此，反省可以赋与认可的能力和处理可能的错误或假设的能力。要点在于词"牛"通常应该对程序而不仅仅是程序员进行解释。为此，对"牛"的使用应该受程序的控制，并可被程序在认为合适的时候转变其意义。比如程序考虑在它前面放置的是否真是一条牛，或者当它认为使用"牛"可能引起混淆时，可决定使用"四条腿的反刍动物"，并由此声明"在我面前的那个东西是一个四条腿的反刍动物"。两种表示（"那个东西"和"四条腿的反刍动物"）得到了匹配。这种对论域中一个实体的假设，如象在更为一般化的"那个东西"这个表达式中所反映的那样，似乎部分地抓住了 Sloman 和 Steels 所提出的思想。语法上的表示与这一假设具有内在联系，而且这种联系本身可以在另一个表示中被反映（引用），这一事实可能导致其自身的灵活性和纠错能力。

六、结论

在形式逻辑中，反射有很多用处，无疑在智能系统中占有重要的地位。本文指出了诸多论题间的联系，从模态逻辑到证明论，从悖论到意义。依作者看，我们仅仅接触到整个冰山的一角。逻辑中有关元级反射的更进一步结果将随着对智能行为的更深入研究而崭露出来。

有必要补充的是，反射首先是由 McCarthy 在1979年强调指出的，并由 Creary〔1979〕, Attardi 和 Simi〔1981〕, Haas〔1981, 82〕得以深入研究。在数理逻辑方面对反射作出贡献的有Church, Currey 以及在λ-演算领域的工作者们，并且在程序语义方面找到了应用，见Stoy〔1977〕, Barendregt〔1984〕。

鸣谢 非常感谢师弟史晓东、陈晓华、王怀民、贲可荣、王戟等同志，他们多次与本人交流，提出了不少宝贵意见。

编译自«Meta-Level Architecture and Reflection» P. Maes, D. Nardi(eds.), 1988, pp37-49