1 涵义, 指称, 语义

理论计算尚非科学. 许多基本概念亟待澄清, 并且当前该领域的研究遵循一种"婚礼蛋糕"范式: 例如, 语言设计让人想到Ptolemy天文学—不断需要更加深入的修正. 然而, 也存在一些有限的主题, 例如复杂度理论和指称语义学, 它们相当远离这种批判.

在这样的情况下, 方法论式的评论极其重要, 因为我们不得不将方法论视为*战略*而将具体的结果视为具有*战术*性质.

我们尤其感兴趣的东西可在1900年代的逻辑漩涡的源头找到, 由Frege, Löwenheim, Gödel等名字刻画. 不熟悉逻辑学史的读者应该参考[vanHeijenoort].

1.1 逻辑中的涵义和指称

让我们从一个例子开始. 存在一个乘法的标准过程, 它由输入27和37产生结果999. 对于这个事实我们可以言称什么?

最初的尝试是言称我们拥有了一个*等式*

$$27 \times 37 = 999$$

这个等式在数学主流中以言称两边*指称*相同的整数且$\times$是Cantor的图的意义下的一个*函数*而获得了含义. (译注: 这里"整数"的原文是"integer", 又有原注, 全文的*integer*将表示*natural number*: 0, 1, 2, ...)

这是指称性的方面, 无疑是正确的, 然而它忽略了基本的点.

存在一个有限的*计算*过程表明这两个指称是相等的. 言称27×37*等于*999是一种滥用 (这并非什么廉价的哲学—而是一个具体的问题), 因为如果我们拥有的这两个东西真是*相同*的, 那么我们就不会感到陈述它们的相等性的需要了. 具体地说, 我们在问一个*问题*, 27×37, 然后得到了一个*答案*, 999. 这两个表达式具有不同的*涵义*, 而且我们必须做什么 (编制证明或进行计算, 或是至少查询百科全书) 来表明这两个*涵义*具有相同的*指称*.

关于$\times$, 将其称为一个(作为图的)函数是不正确的, 因为加载了乘法程序的计算机器无法容纳下一个无限的图. (译注: 这句话是在说*实无限*是不可能容纳于一个经典计算机器之中的, 然而*潜无限*的确是可以的, 比如惰性数据结构*流*. 关于流, 读者首先可以参考论文*CONS should not Evaluate its Arguments*. 当然, 许多编程语言中的常见过程就已经表达了潜无限.) 因此, 我们不得不总结道, 我们面对的是与这个涵义之问相关的一种*有限*的动力学.

尽管指称在很早的阶段就被建模, 涵义则被推向了*主观主义*, 导致当前的数学对于涵义的处理或多或少沦为*句法*操作. 这在我们所要讨论的主题的本质之下并非*先验*, 而我们可以期待在接下来的几十年里找到一种对于计算的处理, 它结合了指称语义学 (数学的清晰性) 与句法 (有限的动力学) 的优点. 本书显然坐落于传统之上, 这种传统基于不幸的当前状况: 在*无限的静态的指称*与*有限的动态的涵义*的对立之中, 指称性的一方要远比另一方先进.

于是乎, 逻辑中由Frege指出的最根本的一个区分是: 给定一个句子A, 存在两种看待它的方式:

- 作为*指令*的序列, 确定了其*涵义*, 例如$A \vee B$的意思是"A*或*B", 等等.
- 作为由这些操作找到的*理想结果*: 此即其*指称*.

"指称 (denotation)", 与"记号 (notation)"相对, 是*被指称*的什么, 而不是*进行指称*的什么. 例如一个逻辑句子的指称是$\mathbf{t}$ (true, 真) 或$\mathbf{f}$ (false, 假), 而$A \vee B$的指称可由A和B的指称通过析取的真值表获得.

拥有相同涵义的两个句子当然拥有相同的指称, 这是显然的; 但是两个拥有相同指称的句子很少拥有相同的涵义. 例如, 取一个复杂的数学等价$A \Leftrightarrow B$. 两个句子拥有相同的指称 (它们同时为真) 但肯定拥有不同的涵义, 不然的话表明这种等价的意义何在?

这个例子允许我们引入一些成组的想法:

- 涵义, 句法, 证明;

- 指称, 真值, 语义, 代数操作.

这是逻辑中的根本对立. 虽然话是这么说, 两方的地位完全不对称!

1.1.1 代数传统

这个传统 (早在Frege的时代之前就由Boole开始) 基于对Ockham的剃刀的激进应用: 我们相当轻易地舍弃了涵义, 只考虑指称. 澄清这种对于逻辑的肢解的合理性的是其可操作性的一面: *it works!*

建立了这种传统的主导性地位的基本转折点在于1916年Löwenheim的定理. 如今, 人们可以将模型论视为这种业已古老的认识论选择所带来的丰富回报. 实际上, 从指称的角度, 即从操作的结果的角度, 如此考虑逻辑, 我们发现了一种有些特殊的代数, 但它允许我们检视对于更加传统的代数而言并不熟悉的操作. 实际上, 避免局限于等式性变体而考虑一般的可定义结构是可以的. 因此, 模型论常以甜美的方式给代数的想法和方法注入了活力.

1.1.2 句法传统

另一方面, "全然忘记指称而专注于涵义"是不可能的, 这出于简单的原因, 即涵义包含指称, 至少是隐式地包含. 因此这并非对称的情况. 实际上几乎不存在统一的句法观点, 因为我们从未能够赋予神秘的涵义以一种操作性的含义. 关于涵义唯一可感知的现实在于其被写下的方式, 即形式化; 但是形式化仍然是一种不够理想的研究对象, 不具备真切的结构, 就像一片*soft camembert*.

这难道意味着纯粹句法的方法毫无讨论的价值吗? 当然不是, 1934年Gentzen的著名定理表明在句法层面上逻辑具有某些深远的对称性 (由切消表达). 然而这些对称被句法的不完美之处遮掩了. 换句话说, 它们不是句法的对称, 而是涵义的对称. 但要想更进一步, 我们必须要将那些对称表达为句法的性质, 而结果并不是很美丽.

那么, 总结我们对于这种传统的观点, 它总是在寻找其根本概念, 也就是说, 涵义和句法之间的操作性区别. 或是把话说得更具体些, 它意在寻找深刻的句法的几何形状上的不变量: 其中可以找到涵义.

被称为"句法性" (因为没有更加高贵的名字了) 的传统, 从没能达到其对手的高度. 近些年来, 也就是代数传统繁荣发展的时期, 句法传统不值一提, 而且无疑将因为缺少问题和方法论而消失一二十年. 这个灾难因为计算机科学 (伟大的句法操纵装置) 而得以避免, 其提出了一些非常重要的理论问题.

其中一些问题 (例如关于算法复杂度的) 似乎更多地需要逻辑的字面而非逻辑的精神. 另一方面, 一切和程序的正确性和模块性有关的问题都深刻地诉诸于句法传统, 诉诸于证明论. 我们被引导至从可追溯到1930年的Herbrand的根本性定理开始对于证明论进行修订. 这个修订给那些一度被认为永远固定下来了的领域带来了新的光亮, 那里曾在很长一段时间内盛行着墨守成规.

在句法逻辑传统与计算机科学之间的交流中, 人们可以在计算的一侧等待着新的语言和新的机器. 但是在逻辑的一侧 (也就是本书的主作者所在的领域), 人们终于可以期望用上一直被残忍忽视的概念基础了.

1.2 两种语义传统

1.2.1 Tarski

这种传统以极端的陈词滥调为人所知: 联结词"$\vee$"被翻译为"或", 诸如此类. 这种解释没有告诉我们关于逻辑联结词的特别突出的东西: 它显然的抱负缺乏是其可操作性的潜在理由. 我们只关心句法的句子 (封闭表达式) 的指称, **t**或**f**.

1. 对于原子句子, 我们假定其指称已然知晓; 例如:
   - $3+2=5$具有指称**t**.
   - $3+3=5$具有指称**f**.
2. 表达式$A \wedge B$, $A \vee B$, $A \Rightarrow B$以及$\neg A$的指称可藉由真值表得到:

A	B	$A \wedge B$	$A \vee B$	$A \Rightarrow B$	$\neg A$
t	**t**	**t**	**t**	**t**	**f**
f	**t**	**f**	**t**	**t**	**t**
t	**f**	**f**	**t**	**f**	
f	**f**	**f**	**f**	**t**	

3. $\forall \xi.A$的指称是**t**当且仅当对于每个在解释的论域中的a, $A[a/\xi]$为**t**. 类似地, $\exists \xi.A$为**t**当且仅当对于某个a而言$A[a/\xi]$为**t**. (原注: $A[a/\xi]$是"A其中所有ξ的(自由)出现均被a替换"的元记号. 若要形式地定义这个东西, 我们需要小心绑定变量.)

再一次, 这个定义从逻辑的角度而言是滑稽可笑的, 但对于其目的而言全然足够了. 模型论的发展证明了这点.

1.2.2 Heyting

Heyting的想法更少为人所知, 但很难想象有比这原初想法的才气和之后发展的庸碌之间更大的差距了. 其目标是极其雄心壮志的: 不是对于指称建模, 而是对于证明建模.

我们不是问"何时一个句子A为真?", 而是问"什么是A的一个证明?". 提到证明, 我们的理解并非句法的形式转写, 而是内在固有的对象, 书写形式仅能给出其模糊的倒影. 我们采取的观点是, 我们作为证明所写下的仅仅是对于某个东西的描述, 它已经自成过程. 因此, 对于我们极其雄心壮志的问题 (并且其也是一个重要的问题, 如果我们计算性地阅读它) 的回答不能是一个形式系统.

1. 对于原子句子, 我们假定我们已经内蕴地明了其证明是什么; 例如, 纸笔计算可以充当"$27 \times 37 = 999$"的证明.
2. 一个对于$A \wedge B$的证明是一个序对(p, q), 其包含一个对于A的证明p和一个对于B的证明q.
3. 一个对于$A \vee B$的证明是一个序对(i, p):
   - $i = 0$, 且p是一个对于A的证明, 或
   - $i = 1$, 且p是一个对于B的证明.
4. 一个对于$A \Rightarrow B$的证明是一个函数f, 其将每个对于A的证明p映射至对于B的证明$f(p)$.
5. 一般$\neg A$被当作$A \Rightarrow \bot$处理, 其中$\bot$是一个没有可能的证明的句子.
6. 一个对于$\forall \xi.A$的证明是一个函数f, 其将定义的论域的每个点a映射至对于$A[a/\xi]$的证明$f(a)$.
7. 一个对于$\exists \xi.A$的证明是一个序对(a, p), 其中a是定义的论域的一个点而p是$A[a/\xi]$的一个证明.

例如, 句子$A \Rightarrow A$由恒等函数证明, 其将A的每个证明p与相同的证明关联起来. 另一方面, 我们该如何证明$A \vee \neg A$呢? 我们必须找到一个对于A的证明或是一个对于$\neg A$的证明, 而在一般情况下这并不可能. 因此, Heyting语义与另一种逻辑, 即Brouwer的直觉主义逻辑相关, 之后我们将遇到它.

不可否认的是, Heyting语义具有相当的原创性: 它不通过逻辑运算自身来解释逻辑运算, 而是通过抽象构造. 现在我们可以看出来这些构造不过只是类型化 (即模块化) 的程序而已. 但是该领域的专家从中看出了非常不同的东西: 一种数学的函数式方法. 换言之, 证明的语义将表达了数学的相当本质.

那是非常异想天开的: 事实上, 一方面我们拥有Tarski传统, 其寻常但忠实 ("$\vee$"的意思是"或", "$\forall$"的意思是"对于所有"), 矫饰皆无. 它也没有成为基础的前景, 因为其若要成为基础, 那么人们就需要基于更原始的东西给出其解释, 而更原始的东西本身也需要它自己的基础. Heyting传统是原创性的, 但其从根本上说有着相同的问题—顺带一提, Gödel的不完备性定理向我们保证它不会是例外. 如果我们希望通过证明A的行为来解释A, 我们就得面对证明的定义用到了两次量词的事实 (分别是$\Rightarrow$和$\forall$的情况). 而且在$\Rightarrow$的情况下, 不能说f的定义的论域是特别好理解的!

既然$\Rightarrow$和$\forall$的情况是成问题的 (从荒谬的基础性角度出发), 有人提议给第4条和第6条增添附加内容, "连带一个f具有此性质的证明". 当然了这没有解决任何问题, 而不得不给出的关于此附加内容有何意义的拜占庭式的讨论—没有一点数学实质的讨论—只会使Heyting的想法 (我们一再强调其为逻辑的基石之一) 失去信誉.

我们将遇到Heyting的想法在Curry-Howard同构里发挥作用. 它也出现在可实现性里. 在这两种情况下, 基础性的矫揉造作都被除去了. 这允许我们好好运用这在未来可能产生伟大应用的想法.