

λC 中逻辑概念的编码

The encoding of logical notions in λC

读书笔记

许博

1 类型理论中的谬论 (absurdity) 与否定 (negation)

在章节 5.4 中, 通过编码蕴含式 $A \Rightarrow B$ 为函数类型 $A \rightarrow B$, 模拟蕴含式的行为, 包括它的导入和消解规则。因为 λP 是 λC 的一部分, 所以 λC 中同样拥有最小谓词逻辑。

本章将处理更多的接词 (connective), 比如否定 (\neg), 合取 (\wedge) 和析取 (\vee)。这些在 λP 中不能表示, 但在 λC 中存在非常优雅的方式去编码这些概念。

将否定 $\neg A$ 看作蕴含式 $A \Rightarrow \perp$, 其中 \perp 是“谬论 (absurdity)”, 也可以称为“矛盾 (contradiction)”。因此 $\neg A$ 被解释为“ A 蕴含了谬论”。为了这个目标, 我们需要谬论的编码:

I. 谬论, *Absurdity*

命题“谬论”或 \perp 的一个独特的性质是: 如果 \perp 为真, 则每一个命题都为真。

每一个命题都为真, 则存在一个接收任意一个命题 α 然后返回 α 的一个成员的函数, 而这个函数的类型为 $\Pi\alpha : *. \alpha$ 。因此“如果 \perp 为真, 则每一个命题都为真”可以表述为“如果存在 $M : \perp$, 则存在 $f : \Pi\alpha : *. \alpha$ ”。因此, 在类型理论中, 定义 \perp 为 $\Pi\alpha : *. \alpha$ 。

\perp -消解 (\perp -elimination) 规则:

$$(\perp\text{-elim}) \frac{\perp}{A}$$

因为 $\perp \equiv \Pi \alpha : *. \alpha$, 则 $s_1 = \square$ 且 $s_2 = *$, 所以 \perp 存在于 $\lambda 2$ 中, 且 $\perp : *$ 。

II. 否定, *Negation*

定义: $\neg A \equiv A \rightarrow \perp$ 。

$A \rightarrow \perp$ 是 $\Pi x : A. \perp$ 的简写, 其中 $A : *$ 且 $\perp : *$, 所以 $(s_1, s_2) = (*, *)$ 。但因为 \perp 存在, 至少 $\lambda 2$ 才能够表示否定。

\perp -导入 (\perp -introduction) 规则:

$$(\perp\text{-intro}) \frac{A \quad \neg A}{\perp}$$

或:

$$(\perp\text{-intro}) \frac{A \quad A \Rightarrow \perp}{\perp}$$

而 (\neg -intro) 和 (\neg -elim) 规则可以以 (\Rightarrow -intro) 和 (\Rightarrow -elim) 规则替换, 前者为后者的特殊情况。

需要注意的是, 尽管 (\perp -intro) 和 (\neg -elim) 都是 (\Rightarrow -elim) 的特殊情况, 但两者具有不同的目的, 前者是为了获得 \perp , 而后者则告诉了我们如何使用一个否定 $\neg A$ 。

2 类型理论中的合取和析取

I. 合取, *Conjunction*

合取 $A \wedge B$ 为真当且仅当 A 和 B 都为真。在 $\lambda 2$ 中, 合取的表示为:

$$A \wedge B \equiv \Pi C : *. (A \rightarrow B \rightarrow C) \rightarrow C$$

这是一个合取的所谓“二阶”编码, 比如 $A \wedge B \equiv \neg(A \rightarrow \neg B)$ 的一阶编码更为通用, 因为后者只在经典逻辑中有效。

$\Pi C : *. (A \rightarrow B \rightarrow C) \rightarrow C$ 可以读作: 对于所有的 C , (A 蕴含 (B 蕴含 C)) 蕴含 C 。若将 A, B, C 都看作是命题, 则可以解释为: 对于所有命题 C , 如果 A 和 B 共同蕴含 C , 则 C 取决于自身。条件“如果 A 和 B 共同蕴含 C ”也即“ A 和 B 都为真”是多余的, 而为了使条件成立, A 和 B 必须为真。这里我认为, C holds on its own 意为 $C \rightarrow C$, 也即第二个 C 由第一个 C 蕴含, 而 $C \rightarrow C$ 恒为真, 所以条件是多余的, 但条件需要满足, 所以 A 和 B 都需要为真。