何幻

Programming is about ideas,

languages are just a way to express them.

语言背后的代数学（六）：Henkin模型

2018-02-04 | Math

回顾

为了解释简单类型化演算$\lambda^{\rightarrow}$中项，

我们介绍了Σ代数，

用Σ代数中的载体A^{σ}来解释基本类型σ，

用载体上的函数集$A^{\sigma\rightarrow\tau}=\{f|f:A^{\sigma}\rightarrow A^{\tau}\}$来解释类型为$\sigma\rightarrow\tau$的所有函数。

但只是做这些对应关系，还是不够的，

我们还得证明，这样的解释是“足够多的”，

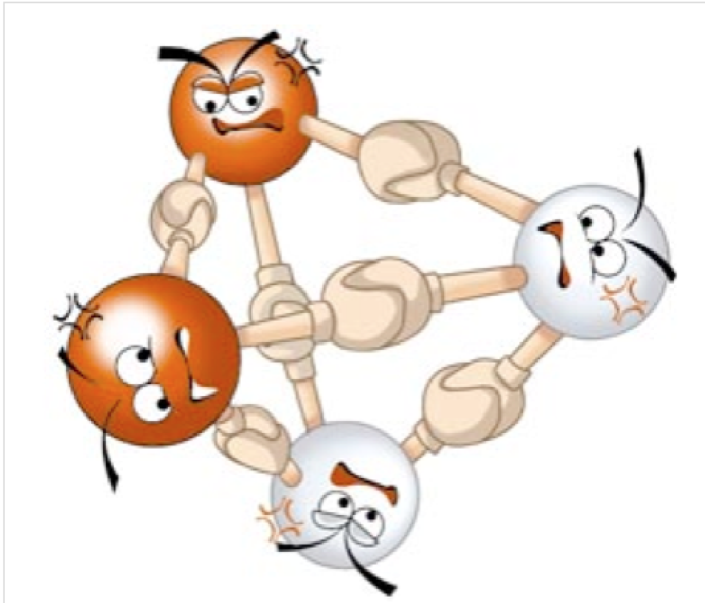
以保证每一个合法$\lambda^{\rightarrow}$项的解释，都在这个Σ代数中。

尤其是使用集合上的函数，来解释具有不动点的λ项时，

$A^{\sigma\rightarrow\tau}$的条件应当适当放宽一些，它不一定恰好是函数集$\{f|f:A^{\sigma}\rightarrow A^{\tau}\}$。

为此我们需要更抽象的，从语义角度定义函数是如何作用到它的参数上的。

1. 作用结构



从形式系统的角度来看，λ项的“应用”，是推导规则的一部分，

包括类型推导规则，$\frac{\Gamma\vdash t_1:T_1\to T_2 \quad \Gamma\vdash t_2:T_1}{\Gamma\vdash t_1\ t_2:T_2}$，

还包括求值规则，$(\lambda x:T.t)\ v\to[x\mapsto v]t$。

项的“应用”，我们可以定义为Σ代数上的一个**作用结构**（applicative structure）。

假设$\lambda x:\sigma.t$是一个类型为$\sigma\to\tau$的λ项，

我们可以把它“应用于”类型为σ的项$v:\sigma$，

我们有，$(\lambda x:\sigma.t)\ v=([x\mapsto v]t):\tau$。

因此，Σ代数上的一个**作用结构**$App^{\sigma,\tau}$，

指的是这样的一个映射，$App^{\sigma,\tau}:A^{\sigma\to\tau}\to A^{\sigma}\to A^{\tau}$，

它将集合$A^{\sigma\to\tau}$中的一个函数，以及集合A^{σ}中的一个元素，

映射成集合A^{τ}中的一个元素。

一个有效的作用结构，必须具有**外延性条件**（extensional），

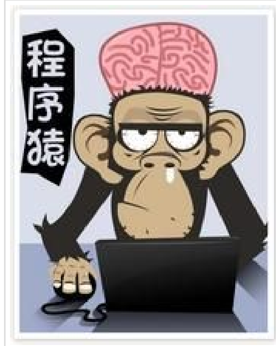
即，$\forall f,g\in A^{\sigma}\to\tau$，

如果对于任意的$d\in A^{\sigma}$，都有$App\ f\ d=App\ g\ d$，则必有$f=g$。

它指出集合$A^{\sigma}\to\tau$的两个函数，如果在$App^{\sigma,\tau}$下的作用效果相同，

那么它们必须是同一个函数。

2. 项的唯一解释



有了满足外延性条件的作用结构之后，

我们就可以归纳的定义出**含义函数**（meaning function）$\mathscr{A}[\![\cdot]\!]$ 了，

（1）$\mathscr{A}[\![\emptyset \vdash c:\sigma]\!]\eta = Const(c)$

（2）$\mathscr{A}[\![x:\sigma \vdash x:\sigma]\!]\eta = \eta(x)$

（3）$\mathscr{A}[\![\Gamma, x:\sigma \vdash M:\tau]\!]\eta = \mathscr{A}[\![\Gamma \vdash M:\tau]\!]\eta$

（4）$\mathscr{A}[\![\Gamma \vdash MN:\tau]\!]\eta = App^{\sigma,\tau}\ \mathscr{A}[\![\Gamma \vdash M:\sigma \to \tau]\!]\eta\ \mathscr{A}[\![\Gamma \vdash N:\sigma]\!]\eta$

（5）$\mathscr{A}[\![\Gamma \vdash \lambda x:\sigma.M:\sigma \to \tau]\!]\eta =$ ，存在唯一的$f \in A^{\sigma\to\tau}$，使得，

$\forall d \in A^{\sigma}$，$App\ f\ d = \mathscr{A}[\![\Gamma, x:\sigma \vdash M:\tau]\!]\eta[x \mapsto d]$

其中，η是满足指派Γ的环境，

$Const$是一个映射，将项常量映射到所有A^{σ}并集的元素上，

使得，如果$c:\sigma$，则$Const(c) \in A^{\sigma}$。

由于以上几个等式覆盖了所有的λ项，因此这样定义的含义函数是完全的。

并且由于它为每一个λ项都指定了确定的语义，因此它给出的解释方式也是唯一的。

它称为**Henkin模型**。

Henkin模型是可靠的，

设$\mathscr{A}$是$\lambda^{\to}$的任意Henkin模型，

$\Gamma \vdash M:\sigma$是可证的，η是一个满足指派Γ的环境，

则$\mathscr{A}[\![\Gamma \vdash M:\sigma]\!]\eta \in A^{\sigma}$ 。

即，如果一个类型断言是可证的，则它在语义上也成立。

（关于完全性的讨论，略）

3. 例子

我们可以对具有单一类型nat的λ演算，定义它的Henkin模型，
令A^{nat}为自然数集，$A^{\sigma\to\tau}$为所有从A^{σ}到A^{τ}的函数集合，
这称为**自然数上的完全集合论函数体系**，
（full set-theoretic function hierarchy over the natural number）。

我们通过$App\ f\ x=f(x)$，把函数$f\in A^{\sigma\to\tau}$，作用到参数$x\in A^{\sigma}$上。
下面我们得出$\lambda x:nat\to nat.\lambda y:nat.xy$的语义。

由于该项是封闭项，选择什么样的环境η都是无关紧要的。
根据上文“含义函数”$\mathscr{A}[\![\cdot]\!]$ 的归纳定义，我们有，
$\mathscr{A}[\![\emptyset\vdash\lambda x:nat\to nat.\lambda y:nat.xy]\!]\eta=$
唯一的$f\in A^{(nat\to nat)\to nat\to nat}$，使得，
$\forall h\in A^{nat\to nat}$，$App\ f\ h=\mathscr{A}[\![x:nat\to nat\vdash\lambda y:nat.xy]\!]\eta[x\mapsto h]$

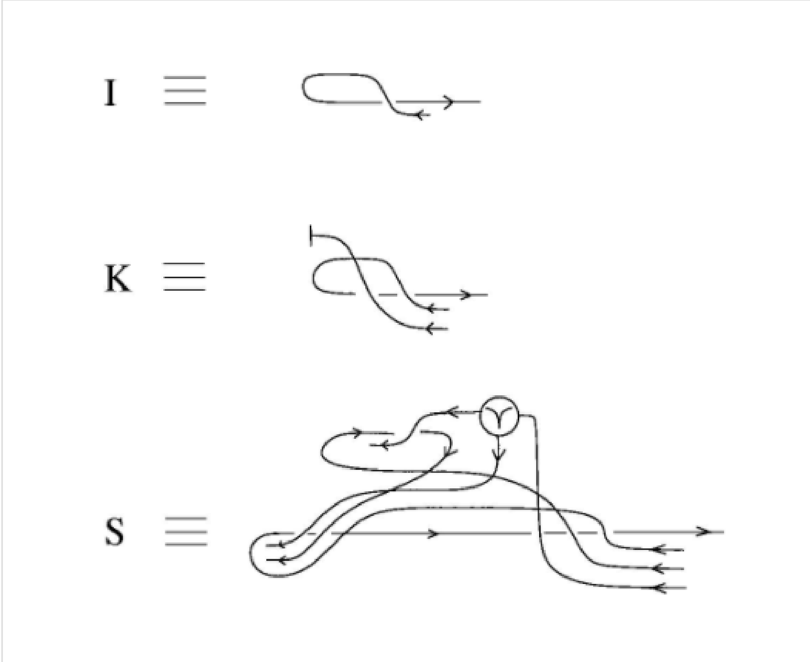
然后我们再来看下，$\mathscr{A}[\![x:nat\to nat\vdash\lambda y:nat.xy]\!]\eta[x\mapsto h]=$，
唯一的$g\in A^{nat\to nat}$，使得，
$\forall n\in A^{nat}$，$App\ g\ n=[\![x:nat\to nat,y:nat\vdash xy]\!]\eta[x\mapsto h][y\mapsto n]$
即，唯一的$g\in A^{nat\to nat}$，使得，$\forall n\in A^{nat}$，$App\ g\ n=App\ h\ n$。
那么这个唯一的$g\in A^{nat\to nat}$，实际上就是h了。

其中，$App\ g\ n=App\ h\ n$是因为，
$App\ g\ n=[\![x:nat\to nat,y:nat\vdash xy]\!]\eta[x\mapsto h][y\mapsto n]$
$=[\![h:nat\to nat,n:nat\vdash hn]\!]\eta$
$=App\ h\ n$

综合以上两个步骤，$\mathscr{A}[\![\emptyset\vdash\lambda x:nat\to nat.\lambda y:nat.xy]\!]\eta=$，
唯一的$f\in A^{(nat\to nat)\to nat\to nat}$，使得，
$\forall h\in A^{nat\to nat}$，$App\ f\ h=h$。
因此，$f\in A^{(nat\to nat)\to nat\to nat}$的语义是一个恒等函数。

我们从语义上证明了以下等式，
$\emptyset\vdash\lambda x:nat\to nat.\lambda y:nat.xy=\lambda x:nat\to nat.x$。

4. 环境模型条件和组合模型条件



以上定义的作用结构$App^{\sigma,\tau}$称为满足**环境模型条件**（enviroment model condition），
依赖了环境η这一附加概念。
它使得每一个合法的λ项，都有模型中一个唯一确定的数学对象与之对应。

由于λ项有λ“抽象”和λ“应用”双重复杂性，
所以，建立一个合理的解释也比较麻烦。
在《你好，类型》系列文章中，
我们介绍过组合子逻辑，我们知道可以将任意的λ项转换成对应的CL项，反之亦然。

因此，如果存在模型可以解释所有的CL项，那么使用它也就可以解释所有λ项了。
CL项比λ项更为简洁，它不包含λ“抽象”，只包含K和S这两个组合子。

类似于λ项的“应用”，
对于K和S的“组合”，我们同样可以定义Σ代数上的一个**作用结构**。

给定Σ代数，对任意类型ρ,σ,τ，如果存在元素$K_{\sigma,\tau}\in A^{\sigma\to(\tau\to\sigma)}$，
$S_{\rho,\sigma,\tau}\in A^{(\rho\to\sigma\to\tau)\to(\rho\to\sigma)\to\rho\tau}$，
满足下列对合适类型x,y,z的等式条件，
$K_{\sigma,\tau}xy=x$，$S_{\rho,\sigma,\tau}xyz=(xz)(yz)$，
我们就称，该作用结构满足**组合模型条件**（combinatory model condition）。

由于所有CL项都可以表示成K和S的“组合”，
因此，满足组合模型条件的作用结构，可以唯一解释所有CL项。

可以证明，一个满足外延性条件的作用结构，

如果它满足环境模型条件，当且仅当它也同样满足组合模型条件。

总结

本文介绍了Henkin模型作用结构所满足的条件，环境模型条件和组合模型条件，
它们是等价的，有了它们我们才能给出λ项的可靠解释，
即，任何合法的λ项都有唯一解释，且在语法上可证的λ项，在语义上也成立。

下文我们开始学习范畴论，为理解笛卡尔闭范畴打好基础。

参考

你好，类型（三）：Combinatory logic
程序设计语言理论基础

代数"> 语言背后的代数学（五）：Σ代数
语言背后的代数学（七）：数学结构