第一章 PCF语言模型

BY ALAN

1 前言

在这一章将给出一个简单的能对可计算函数进行编程的示例语言，称之为PCF，该语言基于类型化的lambda演算, 以自然数和布尔值(真值)作为基本类型, PCF可以构造值的对偶(pair)和定义递归函数，

接下来研究纯类型化的lambda演算, 然后给PCF添加栈和数等数据结构, 由于PCF的类型系统并不如我们期望的那样灵活, 我们只考虑带有多态函数(polymorphism)和类型说明的系统。

2 λ项

lambda演算框架提出了定义变量(veriables), 函数(function)与函数调用(application)这三个基础概念, 下面我们就根据这三点来展开讨论。

- x: 变量就类似于数学中的未知数
- $\lambda x\colon \mathrm{nat}.x$: 函数在lambda演算中是匿名的
- $(\lambda x\colon \mathrm{nat}.x)\,1$: 函数是匿名的, 所以调用只能写成这样

2.1 变量约束

如果一个变量只出现在λ项中, 那么表明这个变量是约束的, 否者这个变量就是自由的。λ抽象本身就是一个约束算子, 也就是说, 在λ项$\lambda x\colon \mathrm{nat}.M$中变量$x$仅仅作为一个占位符, 所以$\lambda x\colon \mathrm{nat}.x$和$\lambda a\colon \mathrm{nat}.a$其实是等价的。我们将这种等价关系称之为$\alpha$等价。

2.2 函数的域

在lambda演算中, 一个函数的定义域可以由形参给出的相应类型而给出, 例如下面这个函数:

$$\lambda x\colon \mathrm{nat}.x$$

上面这个函数就是自然数的等同函数，记法$x\colon \mathrm{nat}$指出该函数的定义域是nat，也是自然数类型,由于在x:nat的情况下, 函数的值也是x, 所以值域也是nat。

理解λ项的一种方式是通过与其他数方法相比较, 还可以使用$x\colon \mathrm{nat} \mapsto x$来描述任何$x\colon \mathrm{nat}$到其自身的函数，在程序语言中定义等同函数的一种更加通用的方式是写成:

$$\mathrm{Id}\,(x\colon \mathrm{nat}) = x$$

2.3 函数的调用

在λ记法中, 函数的调用是比较古怪的, 它遵循下面这两条规则:

1. 函数的调用是左结合的, 比如(MN)P, 将函数M作用在实参N上, 然后把所得的结果函数作用在P上。

2. 第二个规则是, 让函数的作用域尽可能的大, 也就说$\lambda x\colon \sigma\ldots$可以先在类型$\sigma$后面插入一个左括号, 而相应的右括号可以远远的放到可以形成合适的语法表达式上就可以了,比如$\lambda x\colon \mathrm{nat}.\mathrm{MN}$应该理解为$\lambda x\colon \mathrm{nat}.(\mathrm{MN})$而不是$(\lambda x\colon \mathrm{nat}.M)\,N$