9 以定义扩展 λC
Extension of λC with definitions

读书笔记

许博

1 λC 扩展到系统 λD_0

本章在 λC 的基础上扩展通常意义上定义的形式化版本，也即所谓的描述性定义（descriptive definitions）。扩展后的系统 λD_0 尚不能完全支持公理以及公理概念的表示，相应的扩展会在下一章引入 λD 时说明。

为给出 λD_0 的合适的描述，首先扩展表达式的集合。λD_0 中的表达式与 λD 中相同，因此记集合为 $\mathcal{E}_{\lambda D}$。

假设除了之前定义的变量集合 V 以外，还有常量的集合 C。使用符号 $a, a_1, a_i, a', b, ...$ 作为常量的名字，正如我们使用 $x, x_1, x_i, x', y, ...$ 作为变量的名字一样。另外，还假设变量和常量来自不相交的集合，而 $*$ 和 $\square$ 是特殊符号，不属于 V 和 C：

$V \cap C = \emptyset,\ * \neq \square,\ *, \square \notin V \cup C$

定义 1.1（$\mathcal{E}_{\lambda D}$）

$\mathcal{E}_{\lambda D} = V|\square| * |(\mathcal{E}_{\lambda D}\mathcal{E}_{\lambda D})|(\lambda V : \mathcal{E}_{\lambda D}.\mathcal{E}_{\lambda D})|(\Pi V : \mathcal{E}_{\lambda D}.\mathcal{E}_{\lambda D})|C(\overline{\}})$

其中 $\overline{\}}$ 中的上划线表示这是一个 }-表达式的列表。

引入“环境，environment”表示一个定义的列表。

定义 1.2（λD_0 中的描述性定义；环境）

(1) 在 $\mathcal{E}_{\lambda D}$ 中，一个（描述性）定义具有形式

$\overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$

其中所有的 $x_i \in V, a \in C$，并且所有的 $A_i, M, N \in \mathcal{E}_{\lambda D}$

(2) 一个环境 Δ 是一个有限（空或非空）的定义列表。

使用诸如 $\mathcal{D}, \mathcal{D}_i, ...$ 等符号作为元名称表示定义。一个长度为 k 的环境可以被表示为如 $\Delta \equiv \mathcal{D}_1, ..., \mathcal{D}_k$。

关于定义，区分以下元素:

定义 1.3（定义中的元素）
令 $\mathcal{D} \equiv \overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$ 是一个定义。则：
- $\overline{x} : \overline{A}$ 是 $\mathcal{D}$ 中的上下文
- a 是 $\mathcal{D}$ 中被定义的常量，$\overline{x}$ 是参数列表
- $a(\overline{x})$ 是 $\mathcal{D}$ 中的 $definiendum$
- $M : N$ 是 $\mathcal{D}$ 中的语句，M 是 $definiens$ 或 $\mathcal{D}$ 的主体，N 是 $\mathcal{D}$ 的类型。

2 以定义扩展推定

回顾 λC 中的推定，具有如下形式：

$\Gamma \vdash M : N$

但在 $\lambda \mathrm{D}_0$ 中，这样的一个推定可能会依赖一些定义，因此我们在推定之前添加环境，使用元符号“;”分割环境与推定，因此包含定义的推定具有新的形式：

定义 2.1（包含定义的推定；扩展后的推定）

$\Delta ; \Gamma \vdash M : N$,

其中 Δ 是一个环境，Γ 是一个上下文以及 $M, N \in \mathcal{E}_{\lambda D}$。

其含义为：“在环境 Δ 和上下文 Γ 中，M 具有类型 N”。

因此 $M : N$ 由在其头部的列表 Δ 和 Γ 修饰：
(1) 环境 Δ 绑定了 $M : N$ 中出现的常量，
(2) 上下文 Γ 绑定了 $M : N$ 中出现的自由变量。

在整个推定中，存在依赖关系，先出现的变量或常量可能会出现在之后出现的部分中，而后出现的变量或常量则不会出现在之前出现的部分中，尽管前后可能存在相同的名称，但并非表示的不同。

与上下文相同，使用 $\Delta, \mathcal{D}$ 表示在 Δ 右边以 $\mathcal{D}$ 进行扩展。

因为暂且不考虑递归定义，因此在一个定义当中，被定义的常量只出现一次。

再给出修改后的全部推到规则之前，将先引入规则 (def) 和 $(inst)$，前者导入新的定义到已存在的环境中，而后者则是定义的实例化规则。

3 用于添加定义的规则

首先，描述如何扩展一个推定中的环境 Δ，它已被接受并且为正确的：

(i) $\Delta; \Gamma \vdash K : L$

在其中添加一个新的并且良构的定义，需要保证添加的定义本身是良构的，考虑如下一个新的定义：

$\mathcal{D} \equiv \overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$

期望将其添加至 Δ 的尾部。

因为 Δ 中定义的常量，可能会出现在 $\mathcal{D}$ 中，为了使 $\mathcal{D}$ 是可接受的，需要 $M : N$ 在上下文 $\overline{x} : \overline{A}$ 以及环境 Δ 中是可推导的。

因此我们需要一个条件：

(ii) $\Delta; \overline{x} : \overline{A} \vdash M : N$

从而我们得到了规则 (def)：

定义 3.1（用于添加一个定义到一个环境中的推到规则）

令 a 是一个未在 Δ 中定义的新名字，且 $\mathcal{D} \equiv \overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$

$$(def)\ \frac{\Delta; \Gamma \vdash K : L \quad \Delta; \overline{x} : \overline{A} \vdash M : N}{\Delta, \mathcal{D}; \Gamma \vdash K : L}$$

4 用于实例化定义的规则

在实例化定义时，实例化一个参数可能会改变声明列表中的类型，因为定义中的上下文中后面出现的类型，可能依赖之前的声明，考虑一个具有如下形式的定义：

$\mathcal{D} \equiv x_1 : A_1, ..., x_n : A_n \triangleright a(x_1, ..., x_n) := M : N$

对于每个变量 x_i，使用表达式 U_i 进行实例化。对于 U_1，实例化 x_1 时，需要满足条件 $U_1 : A_1$。而对于 U_2，因为 A_2 可能依赖 x_1，所以需要进行替换，因此 U_2 需要满足条件 $U_2 : A_2\,[x_1 := U_1]$。因此 $U_3 : A_3\,[x_1 := U_1, x_2 := U_2]$，需要注意的是，因为 x_1 到 x_n 都是 $\mathcal{D}$ 中上下文的变量，因此 x_i 并不出现在 $\mathcal{D}$ 之外，所以替换时同时替换还是顺序替换并不影响替换的结果。

以此类推，对于 U_i，需要满足条件 $U_i : A_i\,[x_1 := U_1, ..., x_{i-1} := U_{i-1}]$。因为 $x_i, ..., x_n$ 不会出现在 A_i 中，所以对于每个表达式 U_i 所需条件的通用格式为：

$U_i : A_i\,[x_1 := U_1, ..., x_n := U_n]$

套用之前的缩写形式，使用 $\left[\overline{x} := \overline{U}\right]$ 作为替换 $[x_1 := U_1, ..., x_n := U_n]$ 的缩写。因此对于每个 U_i，应有 $U_i : A_i\left[\overline{x} := \overline{U}\right]$。再次使用上划线，以 $\Delta; \Gamma \vdash \overline{U} : \overline{V}$ 作为列表 $\Delta; \Gamma \vdash U_1 : V_1, ..., \Delta; \Gamma \vdash U_n : V_n$ 的缩写，此时规则的 **premisses** 被缩写为 $\Delta; \Gamma \vdash \overline{U} : \overline{A\left[\overline{x} := \overline{U}\right]}$。

而因为 $a(\overline{x}) : N$，因此 $a(\overline{U}) : N\left[\overline{x} := \overline{U}\right]$，因此可以得到实例化规则（的一部分，因为不包括无参数的定义实例化）：

定义 4.1（用于实例化的规则，*1*）
令 a 为一个没有参数列表的常量，令 $\mathcal{D} \in \Delta$，其中 $\mathcal{D} \equiv \overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$，则：

$$\textit{(inst-pos)}\ \frac{\Delta; \Gamma \vdash \overline{U} : \overline{A\left[\overline{x} := \overline{U}\right]}}{\Delta; \Gamma \vdash a(\overline{U}) : N\left[\overline{x} := \overline{U}\right]}$$

其中 pos 意为 positive，正数。

对于没有参数列表的定义，**premisses** 会为空，此时无法保证 **conclusion** 中 $\Delta; \Gamma$ 的良构与否（前文中规定由 **premisses** 保证）。故此时添加一条简单的 **premiss**，以保证 $\Delta; \Gamma$ 是良构的：

定义 4.2（用于实例化的推导规则，*2*）
令 a 为无参数列表的常量，令 $\mathcal{D} \in \Delta$，其中 $\mathcal{D} \equiv \emptyset \triangleright a() := M : N$，则：

$$\textit{(inst-zero)}\ \frac{\Delta; \Gamma \vdash * : \square}{\Delta; \Gamma \vdash a() : N}$$

而将规则 (inst-pos) 和 (inst-zero) 结合便得到了规则 (inst) 以覆盖参数列表空和非空时的情况：

定义 4.3（用于实例化的推导规则）
令 a 为常量，令 $\mathcal{D} \in \Delta$，其中 $\mathcal{D} \equiv \overline{x} : \overline{A} \triangleright a(\overline{x}) := M : N$，则：

$$\textit{(inst)}\ \frac{\Delta; \Gamma \vdash * : \square \quad \Delta; \Gamma \vdash \overline{U} : \overline{A\left[\overline{x} := \overline{U}\right]}}{\Delta; \Gamma \vdash a(\overline{U}) : N\left[\overline{x} := \overline{U}\right]}$$