Краткое введение в лямбда исчисление

Д. Белкин В. Бертыш

16 января 2016 г.

Формальное определение

Множество лямбда-термов строится из бесконечного множества переменных $\mathcal{V}$ использованием аппликации и абстракции:

$$\mathcal{V} = \{v, v', v'', v''' \dots\}$$

$$
\begin{array}{rcll}
x \in \mathcal{V} & \implies & x \in \Lambda & \text{(Переменная является } \lambda\text{-термом)} \\
M, N \in \mathcal{V} & \implies & M\ N \in \Lambda & \text{(Аппликация является } \lambda\text{-термом)} \\
M \in \mathcal{V}, v \in \mathcal{V} & \implies & \lambda v.M \in \Lambda & \text{(Абстракция является } \lambda\text{-термом)}
\end{array}
$$

Свободные и связанные переменные

Множество свободных переменных терма N обозначается $FV(N)$

$$
\begin{aligned}
FV(x) &\equiv \{x\} \\
FV(M\ N) &\equiv FV(M) \cup FV(N) \\
FV(\lambda x.N) &\equiv FV(N) \setminus \{x\}
\end{aligned}
$$

Множество связанных переменных терма N принято обозначать как $BV(N)$. Заметим, что переменная связана, если она образует абстракцию. Мы будем называть терм M закрытым (или комбинатором), если $FV(M) \equiv \varnothing$. Множество комбинаторов обозначим как Λ°.

Подстановки

Результат подстановки N вместо всех свободных вхождений x в M обозначим $M[x := N]$ и определим как:

$$
\begin{aligned}
x[x := N] &\equiv N \\
y[x := N] &\equiv y \text{ (если } x \neq y) \\
(M_1\ M_2)[x := N] &\equiv ((M_1[x := N])\ (M_2[x := N])) \\
(\lambda y.M)[x := N] &\equiv (\lambda y.M[x := N])
\end{aligned}
$$

α-конверсия

Введем на Λ отношение эквивалентности, задаваемое следующим образом:

$$\forall P\ P =_\alpha P$$
$$\lambda x.P =_\alpha \lambda y.P[x := y], \text{ если } y \notin FV(P)$$

Это отношение назвается α-эквивалентностью.
Если $M =_\alpha N$, иногда также пишут $\lambda \models M =_\alpha N$

β-редукция

Основная схема λ-исчисления

$$\forall M, N \in \Lambda : (\lambda x.M)\ N = M[x := N] \qquad (\beta)$$

Однократное применение аксиомы (β) будем обозначать как $(\to_\beta)$. Место, где можно применить аксиому (β) будем называть редексом. Терм, в котором нет редексов будем называть нормальной формой выражения. Применение аксиомы (β) ноль или более раз (транзитивное замыкание $\to_\beta$) будем называть β-редукцией и обозначим как $\twoheadrightarrow_\beta$.
Также введем отношение β-эквивалентности, которое обозначим как $=_\beta$ и определим как

$$\forall M, N \in \Lambda : M \twoheadrightarrow_\beta N \implies M =_\beta N$$

η-редукция

Дополним наше λ-исчисление еще одной аксиомой

$$\lambda x.M\ x \equiv M \text{ если } x \notin FV(M) \qquad (\eta)$$

Аналогично β-редукции вводятся однократное применение аксиомы η ($\to_\eta$), её транзитивное замыкание – η-редукция ($\twoheadrightarrow_\eta$) и отношение η-эквивалентности ($=_\eta$).

Теорема о неподвижной точке

$(i)\ \ \forall F\ \exists X\ (F\ X = X)$

Более того, существует комбинатор, находящий X

$$\boldsymbol{Y} = \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$$

$(ii)\ \forall F\ \ \boldsymbol{Y}\ F = F\ (\boldsymbol{Y}\ F)$

Теорема Чёрча-Россера

$$\forall M, N_1, N_2 : M \twoheadrightarrow_\beta N_1, M \twoheadrightarrow_\beta N_2 \implies \exists N : N_1 \twoheadrightarrow_\beta N, N_2 \twoheadrightarrow_\beta N$$

Иначе говоря, для β-редукции выполняется свойство ромба

$$\begin{tikzcd}
 & M \arrow[dl, "\beta"'] \arrow[dr, "\beta"] & \\
N_2 \arrow[dr, "\beta"] & & N_1 \arrow[dl, "\beta"'] \\
 & N &
\end{tikzcd}$$

Логические значения

Пусть **True** и **False** некие лямбда термы, при этом **True** $\neq_\beta$ **False**. Один из возможных способов сделать это следующий

$$\textbf{True} = \lambda ab.a$$
$$\textbf{False} = \lambda ab.b$$

Логические операции

Определим термы **and**, **or** и **not**, представляющие соответствующие логические операции

$$\begin{aligned}
\textbf{and} &= \lambda t_1 t_2 ab.(t_1\ (t_2\ a\ b)\ b) \\
\textbf{or} &= \lambda t_1 t_2 ab.(t_1\ a\ (t_2\ a\ b)) \\
\textbf{not} &= \lambda fab.f\ b\ a
\end{aligned}$$

Ветвление

Условную конструкцию if можно определить следующим образом

$$\textbf{if} = \lambda t.t$$
$$\textbf{if}\ \textbf{True}\ a\ b \twoheadrightarrow_\beta a$$
$$\textbf{if}\ \textbf{False}\ a\ b \twoheadrightarrow_\beta b$$

Натуральные числа

Определим натуральные числа следующим образом:

$$\begin{aligned}
&\overline{\mathbf{0}} = \lambda fx.x = \mathbf{const}\\
&\overline{\mathbf{1}} = \lambda fx.f\ x = \mathbf{id}\\
&\dots\\
&\overline{\boldsymbol{n}} = \lambda f.\lambda x.f^n x\\
&\text{Где } f^n(x) = \underbrace{f(f(f(\dots f(}_{n \text{ раз}}x))))
\end{aligned}$$

Такое представление чисел называется нумералами Чёрча.

Простые арифметические операции

Определим инкремент следующим образом:

$$\mathbf{succ}\ \overline{\boldsymbol{n}} \equiv \overline{\boldsymbol{n} + \mathbf{1}}$$
$$\mathbf{succ} = \lambda nfx.f\ (n\ f\ x)$$

Сложение, умножение и возведение в степень определяются так:

$$\mathbf{add} = \lambda abfx.(b\ f\ (a\ f)\ x)$$
$$\mathbf{mul} = \lambda abf.a\ (b\ f)$$
$$\mathbf{pow} = \lambda ab.b\ a$$

Сложные арифметические операции

Вычитание – сложно. Нужно для индуктивно построенного $\overline{\boldsymbol{n}}$ найти $\overline{\boldsymbol{n}-\mathbf{1}}$.
Решение – пары.

$$\begin{aligned}\textbf{mkPair} =& \lambda abf.f\ a\ b\\ \textbf{fst} =& \lambda p.p\ \textbf{True}\\ \textbf{snd} =& \lambda p.p\ \textbf{False}\end{aligned}$$

Здесь **mkPair** создает пару, а **fst** и **snd** получают первый и второй элементы пары соответственно.
Далее определим “инкремент” пары

$$\textbf{succP} = \lambda p.\textbf{mkPair}\ (\textbf{snd}\ p)\ (\textbf{succ}\ (\textbf{fst}\ p))$$

Вычитание

Теперь можно определить декремент

$$\mathbf{prev} = \lambda n.\mathbf{fst}\ (n\ \mathbf{succP}\ (\mathbf{mkPair}\ \overline{0}\ \overline{0}))$$

и вычитание

$$\mathbf{sub} = \lambda ab.b\ \mathbf{prev}\ a$$

Рекурсивные функции

Попробуем написать терм, вычисляющий факториал. Для начала нужно определять, что $\overline{\boldsymbol{n}} = \overline{\mathbf{0}}$. Напишем терм, выполняющий эту проверку:

$$\mathbf{isZero} = \lambda n.n\ (\lambda x.\mathbf{False})\ \mathbf{True}$$

Определение факториала

Следующим шагом, уже в практически естественном виде мы можем определить факториал как

$$\mathbf{fact} = \lambda n.\mathbf{if}\ (\mathbf{isZero}\ n)\ \overline{\mathbf{1}}\ (\mathbf{mul}\ \overline{\boldsymbol{n}}\ (\mathbf{fact}\ (\mathbf{prev}\ \overline{\boldsymbol{n}})))$$

Но неверно, потому что зациклились. Выделим всё из рекурсивного вызова.

$$\mathbf{T} = \lambda fn.\mathbf{if}\ (\mathbf{isZero}\ n)\ \overline{\mathbf{1}}\ (\mathbf{mul}\ \overline{\boldsymbol{n}}\ (f\ (\mathbf{prev}\ \overline{\boldsymbol{n}})))$$

Тогда факториал определяется как

$$\mathbf{fact} = \mathbf{T}\ \mathbf{fact}$$

Избавимся от зацикливания при помощи $\mathbf{Y}$

$$\mathbf{fact} = \mathbf{Y}\ \mathbf{T}$$

Presto.