Краткое введение в лямбда-исчисление

Белкин Дмитрий, студент группы 4362
Бертыш Вадим, студент группы 4374

23 декабря 2015

LaTeXdfadfsafgfgaserhgsetrvffdv

Введение.

VODAVODAVODA $\to_\beta$ $\twoheadrightarrow_\beta$

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

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

При этом:

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

Или же, используя БНФ:

$$\begin{aligned} var &::= v | var' \\ \Lambda &::= var | (var\ var) | (\lambda var.\Lambda) \end{aligned}$$

В дальнейшем условимся, что:

- $x, y, z, \cdots$ – произвольные переменные.
- $M, N, L, \cdots$ – произвольные лямбда термы.
- Скобки верхнего уровня опускаются.
- Аппликация правоассоциативна, т.е.

$$F\ M_1\ M_2\ \ldots\ M_n \equiv (\ldots((F\ M_1)\ M_2)\ \ldots\ M_n)$$

- Допустима абстракция сразу по нескольким переменным

$$\lambda x_1\ x_2\ \ldots\ x_n.M \equiv \lambda x_1.(\lambda x_2.\ \ldots\ (\lambda x_n.M))$$

Свободные и связанные переменные

Множеством свободных переменных терма N называется $FV(N)$, индуктивно определяемое по следующим правилам:

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

Мы будем называть переменную связанной, если она не принадлежит множеству свободных. 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°. Существует раздел математики, тесно связанный с λ-исчислением - комбинаторная логика. Она изучает комбинаторы и вычисления построенные на них. В комбинаторной логике вводится несколько стандартных комбинаторов:

$$Sxyz = xz(yz) \qquad\qquad Kxy = x$$

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 носит название α-эквивалентность. Так же напишем некоторые аксиомы для этого отношения:

$$\begin{aligned} M &=_\alpha M \\ M =_\alpha N &\Rightarrow N =_\alpha M \\ M =_\alpha N, N =_\alpha L &\Rightarrow M =_\alpha L \\ M =_\alpha M' &\Rightarrow M\ Z =_\alpha M'\ Z \\ M =_\alpha M' &\Rightarrow Z\ M =_\alpha Z\ M' \\ M =_\alpha M' &\Rightarrow \lambda x.M =_\alpha \lambda x.M' \end{aligned}$$

Если $M =_\alpha N$, иногда также пишут $\lambda \models M =_\alpha N$

Обобщая определения выше, M и N равны (альфа-эквивалентны), если можно получить один из другого, путем замены имен связанных переменных. Любые два равных терма в одинаковом контексте так же будут равны.

Сам процесс замены имени носит название α-конверсия и определяется следующим образом:

$$\lambda x.M \to_\alpha \lambda y.(M[x := y]) \text{ если } y \notin FV(M) \qquad (\alpha)$$

Подстановки

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

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

Условимся, что подстановка всегда выполняется корректно, то есть, заменяемая переменная никогда не является связанной ни в каких внутренних термах. Этой ситуации всегда можно избежать заменой имен во внутреннем терме. Например для следующей подстановки предварительно произведем замену имени во внутренней лямбды

$$\lambda a.\lambda b.a\ b[a := b] = \lambda a.(\lambda b.a\ b[b := b'])[a := b] = \lambda b.\lambda b'.b\ b'$$

Теперь мы готовы описать λ-исчисление как формальную теорию.

β-редукция

Основная схема λ-исчисления

$$\forall M, N \in \Lambda : (\lambda x.M)\ N = M[x := N] \tag{β}$$

Лемма 1.

$$(\lambda x_1\ x_2\ \dots\ x_n.M)\ X_1\ X_2\ \dots\ X_n \equiv M[x_1 := X_1][x_2 := X_2]\dots[x_n := X_n]$$

Доказательство. Пусть $M' = \lambda x_2 \dots x_n.M$. По аксиоме (β) мы имеем

$$(\lambda x_1.M')\ X_1\ X_2\ \dots\ X_n \equiv M'[x_1 := X_1]X_2\ \dots\ X_n$$

Дальнейшее равенство получаем индукцией по связанным переменным. □

Однократное применение аксиомы (β) будем называть β-редукцией и обозанчать как $(\to_\beta)$. Место, где можно применить аксиому (β) называется редексом. Применение аксиомы (β) ноль или более раз обозначим как $\twoheadrightarrow_\beta$ Также введем отношение β-эквивалентности стандартным образом, как транзитивное замыкание отношения $\twoheadrightarrow_\beta$ и обозначим $=_\beta$ Любое вычисление представляет собой некоторое колличество шагов β-редукции.

Вычисления заканчиваются, когда в терме не остается редексов, будем говорить, что такие термы находятся в нормалной форме.

Стоит заметить, что β-редукция, хотя и называется редукцией, не всегда сокращает терм, и тем более, не всегда делает терм проще. Как пример плохого, можно привести следующий терм:

$$\omega = (\lambda x.xx)(\lambda x.xx)$$

Такая конструкция за один шаг редуцируется в себя. Очевидно, что у ω никакая цепочка преобразований не приведет к нормальной форме, тогда возникает вопрос, для каких термов существует нормальная форма, зависит ли она от порядка применения (β) и любой ли порядок ведет к нормальной форме. На некоротые из этих вопросов отвечает теорема Черча-Россера:

На данном этапе мы можем относительно свободно строить различные термы, однако особый подход требуется для описания рекурсивный функций, о чем сейчас и пойдет речь. Для бестипового лямбда-исчисления справедлива следующая теорема:

Теорема о неподвижной точке.

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

Более того, существует комбинатор, находящий X

$Y = \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$

(ii) $\forall F.(Y\ F = F\ (Y\ F))$

Доказательство.
Определим $W = \lambda x.F(x\ x)$ и $X = W\ W$ тогда

$$\begin{aligned} X \equiv W\ W \equiv (\lambda x.F(x\ x))W \rightarrow_\beta \\ F\ (W\ W) \equiv FX \end{aligned}$$

Аналогично

$$\begin{aligned} &Y\ F \rightarrow_\beta \\ &(\lambda x.F\ (x\ x))(\lambda f.(\lambda x.F\ (x\ x))) \rightarrow_\beta \\ &F\ ((\lambda f.(\lambda x.F\ (x\ x)))(\lambda f.(\lambda x.F\ (x\ x)))) \equiv F\ (Y\ F) \end{aligned}$$

□

Поговорим о лямбда-исчислении со стороны программирования. Для начала построим булеву алгебру на лямбда исчислении.

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

$$\begin{aligned} True = \lambda ab.a \\ False = \lambda ab.b \end{aligned}$$

Определим также термы and, or и not, представляющие соответсвующие опперации

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

Доходчивый читатель сам удостоверится, что такие определения удовлетворяют аксиомам булевой алгебры. Мы можем проверить различные свойства булевой алгебры, к примеру проверим инволюцию отрицания

$$not\ not\ x \to_\beta not\ \lambda a'b'.xb'a' \to_\beta \lambda a''b''.(\lambda a'b'.ab'a')b''a'' \twoheadrightarrow_\beta \lambda a''b''.xa''b'' \equiv x$$

Стоит также упомянуть условную конструкцию if

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

Следующий пункт - натуральные числа Определим натуральные числа следующим образом: Такое представление чисел назыается нумералами Чёрча Можно показать, что для них выполняются аксиомы Пеано.

Определим также основные арифметические опперации.