

Функциональное программирование

Лекция 2. Рекурсия и редукция

Денис Николаевич Москвин

ИТМО, магистратура JB SE

14.09.2020

- 1 Теорема о неподвижной точке
- 2 Редексы и нормальная форма
- 3 Теорема Чёрча-Россера
- 4 Стратегии редукции

- 1 Теорема о неподвижной точке
- 2 Редексы и нормальная форма
- 3 Теорема Чёрча-Россера
- 4 Стратегии редукции

Схема β -преобразования

$$(\lambda n. M) N = [n \mapsto N] M$$

даёт возможность решать простейшие уравнения на термы.

Пример

Найти F , такой что $\forall M, N, L \quad \lambda \vdash F M N L = M L (N L)$.

$$\begin{aligned} F M N L &= M L (N L) \\ F M N L &= (\lambda l. M l (N l)) L \\ F M N &= \lambda l. M l (N l) \\ F M &= \lambda n. \lambda l. M l (n l) \\ F &= \lambda m n l. m l (n l) \end{aligned}$$

А если уравнение рекурсивное, например, $F M = M F$?

Схема β -преобразования

$$(\lambda n. M) N = [n \mapsto N] M$$

даёт возможность решать простейшие уравнения на термы.

Пример

Найти F , такой что $\forall M, N, L \lambda \vdash F M N L = M L (N L)$.

$$\begin{aligned} F M N L &= M L (N L) \\ F M N L &= (\lambda l. M l (N l)) L \\ F M N &= \lambda l. M l (N l) \\ F M &= \lambda n. \lambda l. M l (n l) \\ F &= \lambda m n l. m l (n l) \end{aligned}$$

А если уравнение рекурсивное, например, $F M = M F$?

Оказывается, имеется универсальный способ решения!

Теоремы о неподвижной точке

Теорема

Для любого λ -терма F существует неподвижная точка:

$$\forall F \in \Lambda \quad \exists X \in \Lambda \quad \lambda \vdash FX = X$$

Доказательство

Введем $W \equiv \lambda x. F(x x)$ и $X \equiv WW$. Тогда

$$X \equiv WW \equiv (\lambda x. F(x x)) W = F(WW) \equiv FX \quad \blacksquare$$

Теорема

Существует комбинатор неподвижной точки Y , такой что

$$\forall F \quad F(YF) = YF.$$

Доказательство

Введём $Y \equiv \lambda f. (\lambda x. f(x x))(\lambda x. f(x x))$. Имеем $YF \equiv$

$$(\lambda x. F(x x))(\lambda x. F(x x)) = F(\underbrace{((\lambda x. F(x x))(\lambda x. F(x x)))}_{YF}) \equiv F(YF) \quad \blacksquare$$

Y-комбинатор позволяет ввести рекурсию в λ -исчисление.

Пример

Факториал рекурсивно:

$$\text{fac} = \lambda n. \text{if} (\text{iszro } n) 1 (\text{mult } n (\text{fac} (\text{pred } n)))$$

Переписываем в виде

$$\text{fac} = \underbrace{(\lambda f n. \text{if} (\text{iszro } n) 1 (\text{mult } n (f (\text{pred } n))))}_{\text{fac}' } \text{fac}$$

Отсюда видно, что fac — неподвижная точка для вспомогательной функции fac' :

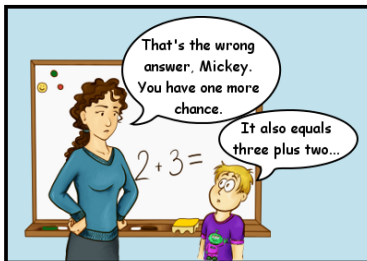
$$\text{fac} = Y \text{fac}'$$

Как работает $\text{fac} \equiv Y \text{ fac}'$?

Пример

```
fac 3 = (Y fac') 3
      = fac' (Y fac') 3
      = if (iszro 3) 1 (mult 3 ((Y fac') (pred 3)))
      = mult 3 ((Y fac') 2)
      = mult 3 (fac' (Y fac') 2)
      = mult 3 (mult 2 ((Y fac') 1))
      = mult 3 (mult 2 (mult 1 ((Y fac') 0)))
      = mult 3 (mult 2 (mult 1 1))
      = 6
```


- 1 Теорема о неподвижной точке
- 2 Редексы и нормальная форма
- 3 Теорема Чёрча-Россера
- 4 Стратегии редукции



Sad fact: many math teachers do not know the difference between equality and reduction.

- Мы строили λ -исчисление как теорию о равенстве термов.
- Эффективный способ доказывать равенство — сократить все редексы:

Докажем, что $K I = I I K_*$

$$K I \equiv (\lambda x y. x) (\lambda z. z) = \lambda y z. z$$

$$I I K_* \equiv (\lambda x. x) I K_* = I K_* \equiv (\lambda x. x) (\lambda y z. z) = \lambda y z. z$$

- А как доказывать неравенство? Например, $K \neq K_*$?

- Процесс вычисления по схеме β -преобразования

$$(\lambda x. M) N =_{\beta} [x \mapsto N] M$$

носит односторонний характер: термы при конверсиях «упрощаются».

- Для исследования подобного вычислительного аспекта вводят *отношение редукции*:
 - $K I \rightarrow_{\beta} K_*$ — редуцируется за один шаг;
 - $I I K_* \rightarrow_{\beta} K_*$ — редуцируется;
 - $K I =_{\beta} I I K_*$ — конвертируемо (равно).
- **Может ли шаг редукции увеличить число редексов в выражении?**

Определение

Бинарное отношение \mathcal{R} над Λ называют **совместимым** (с операциями λ -исчисления), если для любых $M, N, Z \in \Lambda$:

$$\begin{aligned} M \mathcal{R} N &\Rightarrow (ZM) \mathcal{R} (ZN), \\ &\quad (MZ) \mathcal{R} (NZ), \\ &\quad (\lambda x. M) \mathcal{R} (\lambda x. N). \end{aligned}$$

Определения

Совместимое отношение эквивалентности называют отношением **конгруэнтности** над Λ .

Совместимое, рефлексивное и транзитивное отношение называют отношением **редукции** над Λ .

Определение

Бинарное отношение β -редукции за один шаг \rightarrow_β над Λ :

$$(\lambda x. M) N \rightarrow_\beta [x \mapsto N] M$$

$$M \rightarrow_\beta N \Rightarrow ZM \rightarrow_\beta ZN$$

$$M \rightarrow_\beta N \Rightarrow MZ \rightarrow_\beta NZ$$

$$M \rightarrow_\beta N \Rightarrow \lambda x. M \rightarrow_\beta \lambda x. N$$

По определению \rightarrow_β это β -правило плюс совместимость.

Пример: редуцируем терм I ($K I$)

$$(\lambda x. x) ((\lambda y z. y) (\lambda p. p)) \rightarrow_\beta (\lambda y z. y) (\lambda p. p) \rightarrow_\beta \lambda z p. p$$

$$(\lambda x. x) ((\lambda y z. y) (\lambda p. p)) \rightarrow_\beta (\lambda x. x) (\lambda z p. p) \rightarrow_\beta \lambda z p. p$$

Определение

Бинарное отношение β -*редукции* \rightarrow_β над Λ (индуктивно):

$$\begin{array}{lcl} & M \rightarrow_\beta M & \text{(refl)} \\ M \rightarrow_\beta N \Rightarrow & M \rightarrow_\beta N & \text{(ini)} \\ M \rightarrow_\beta N, N \rightarrow_\beta L \Rightarrow & M \rightarrow_\beta L & \text{(trans)} \end{array}$$

Отношение \rightarrow_β является **транзитивным рефлексивным** замыканием \rightarrow_β и, следовательно, отношением редукции.

Примеры

$$\begin{array}{lcl} (\lambda x. x) ((\lambda y z. y) (\lambda p. p)) & \rightarrow_\beta & (\lambda x. x) ((\lambda y z. y) (\lambda p. p)) \\ (\lambda x. x) ((\lambda y z. y) (\lambda p. p)) & \rightarrow_\beta & (\lambda y z. y) (\lambda p. p) \\ (\lambda x. x) ((\lambda y z. y) (\lambda p. p)) & \rightarrow_\beta & \lambda z p. p \end{array}$$

Определение

Бинарное отношение $=_{\beta}$ над Λ (индуктивно):

$$\begin{aligned}M \rightarrow_{\beta} N &\Rightarrow M =_{\beta} N && \text{(ini)} \\M =_{\beta} N &\Rightarrow N =_{\beta} M && \text{(sym)} \\M =_{\beta} N, N =_{\beta} L &\Rightarrow M =_{\beta} L && \text{(trans)}\end{aligned}$$

Отношение $=_{\beta}$ является отношением конгруэнтности.

Утверждение

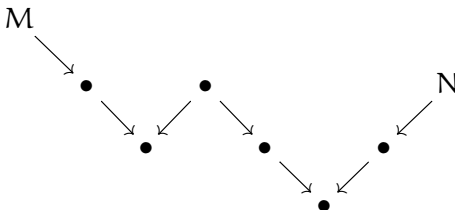
$$M =_{\beta} N \Leftrightarrow \lambda \vdash M = N.$$

Доказательство

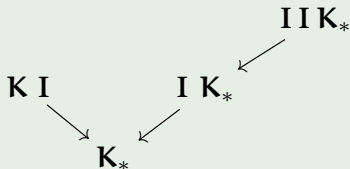
Индукция по определениям. ■

Отношение конвертируемости $=_{\beta}$ (интуитивно)

Интуитивно: два терма M и N связаны отношением $=_{\beta}$, если есть связывающая их цепочка \rightarrow_{β} -стрелок:



Пример. $K I =_{\beta} I I K_*$



Определение

λ -терм M **находится** в β -нормальной форме (β -NF), если в нем нет подтермов, являющихся β -редексами.

Определение

λ -терм M **имеет** β -нормальную форму, если для некоторого N выполняется $M =_{\beta} N$ и N находится в β -NF.

Примеры

- Терм $\lambda x y. x (\lambda z. z x) y$ находится в β -нормальной форме.
- Терм $(\lambda x. x x) y$ не находится в β -нормальной форме, но имеет в качестве β -nf терм $y y$.

Нормальная форма (2)

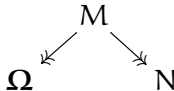
Утверждение

Не все термы имеют β -нормальную форму.

Пример

$$\begin{aligned}\Omega &\equiv \omega \omega \\ &\equiv (\lambda x. x x) (\lambda x. x x) \\ &\rightarrow_{\beta} (\lambda x. x x) (\lambda x. x x) \\ &\rightarrow_{\beta} \dots\end{aligned}$$

Это пока не доказательство! Может быть существует терм N в β -NF, такой что $\Omega =_{\beta} N$, например, так



Бывают термы, «удлинняющиеся» при редукции.

Пример

$$\begin{aligned}\Omega_3 &\equiv \omega_3 \omega_3 \\ &\equiv (\lambda x. x x x) (\lambda x. x x x) \\ &\rightarrow_\beta (\lambda x. x x x) (\lambda x. x x x) (\lambda x. x x x) \\ &\rightarrow_\beta (\lambda x. x x x) (\lambda x. x x x) (\lambda x. x x x) (\lambda x. x x x) \\ &\rightarrow_\beta \dots\end{aligned}$$

С какой скоростью будет расти $\Omega_4 \equiv \omega_4 \omega_4$?

Не все последовательности редукций приводят к β -NF.

Пример

$$\begin{aligned} \mathbf{KI}\Omega &\equiv \mathbf{KI}((\lambda x. x x) (\lambda x. x x)) \\ &\rightarrow_{\beta} \mathbf{KI}((\lambda x. x x) (\lambda x. x x)) \\ &\rightarrow_{\beta} \dots \end{aligned}$$

$$\begin{aligned} \mathbf{KI}\Omega &\equiv (\lambda x y. x) \mathbf{I}\Omega \\ &\rightarrow_{\beta} (\lambda y. \mathbf{I}) \Omega \\ &\rightarrow_{\beta} \mathbf{I} \end{aligned}$$

Редукционные графы (1)

Определение

Редукционный граф терма $M \in \Lambda$ (обозначаемый $G_\beta(M)$) — это ориентированный мультиграф с вершинами в $\{N \mid M \twoheadrightarrow_\beta N\}$ и дугами \rightarrow_β .

$$G_\beta(\mathbf{I}(\mathbf{I}x)) = \bullet \begin{array}{c} \curvearrowright \\ \longrightarrow \\ \curvearrowleft \end{array} \bullet \longrightarrow \bullet \quad G_\beta(\mathbf{\Omega}) = \bullet \begin{array}{c} \curvearrowright \\ \longrightarrow \\ \curvearrowleft \end{array} \bullet$$

$$G_\beta((\lambda x. \mathbf{I}) \mathbf{\Omega}) = \bullet \begin{array}{c} \curvearrowright \\ \longrightarrow \\ \curvearrowleft \end{array} \bullet \quad G_\beta(\mathbf{KI} \mathbf{\Omega}) = \bullet \begin{array}{c} \curvearrowright \\ \longrightarrow \\ \curvearrowleft \end{array} \bullet \longrightarrow \bullet \begin{array}{c} \curvearrowright \\ \longrightarrow \\ \curvearrowleft \end{array} \bullet \longrightarrow \bullet$$

$$G_\beta(\mathbf{\Omega}_3) = ??? \quad G_\beta((\lambda x. \mathbf{I}) \mathbf{\Omega}_3) = ???$$

Редукционные графы (2)

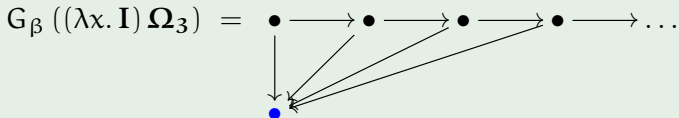
Не все редукционные графы конечны.

Пример

$$G_{\beta}(\Omega_3) = \bullet \longrightarrow \bullet \longrightarrow \bullet \longrightarrow \bullet \longrightarrow \dots$$

Не все бесконечные редукционные графы не имеют нормальной формы.

Пример

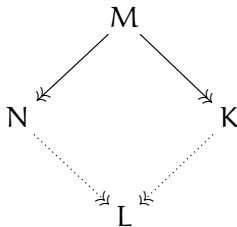


- 1 Теорема о неподвижной точке
- 2 Редексы и нормальная форма
- 3 Теорема Чёрча-Россера**
- 4 Стратегии редукции

Теорема [Чёрч-Россер]

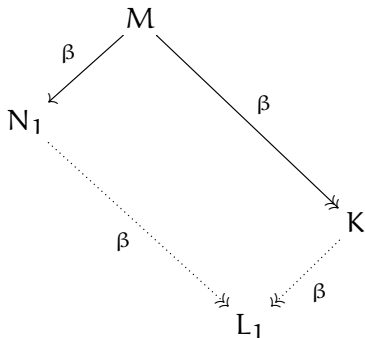
Если $M \rightarrow_{\beta} N$, $M \rightarrow_{\beta} K$, то существует L , такой что $N \rightarrow_{\beta} L$ и $K \rightarrow_{\beta} L$.

- Иначе говоря, β -редукция обладает *свойством ромба*:



- Иногда используют термин *конфлюентность*.

Доказываем лемму полосы (Strip lemma):



А затем из полосок составляем «ромб».

Термы *подкрашенного* лямбда-исчисления, множество которых мы будем обозначать Λ , определяются индуктивно:

$$\begin{aligned}x \in V &\Rightarrow x \in \Lambda, \\M, N \in \Lambda &\Rightarrow (MN) \in \Lambda, \\M \in \Lambda, x \in V &\Rightarrow (\lambda x. M) \in \Lambda, \\M, N \in \Lambda, x \in V &\Rightarrow (\lambda x. M) N \in \Lambda.\end{aligned}$$

То есть в редексах (и только в них) некоторые лямбды могут быть покрашены.

Подкрашенные редукции (одношаговые и многошаговые) определяются стандартным образом на базе правил сокращения:

$$\begin{aligned}(\lambda x. M) N &\rightarrow_{\beta} [x \mapsto N] M, \\(\lambda x. M) N &\rightarrow_{\beta} [x \mapsto N] M.\end{aligned}$$

То есть вычисления игнорируют раскраску, но не стирают ее.

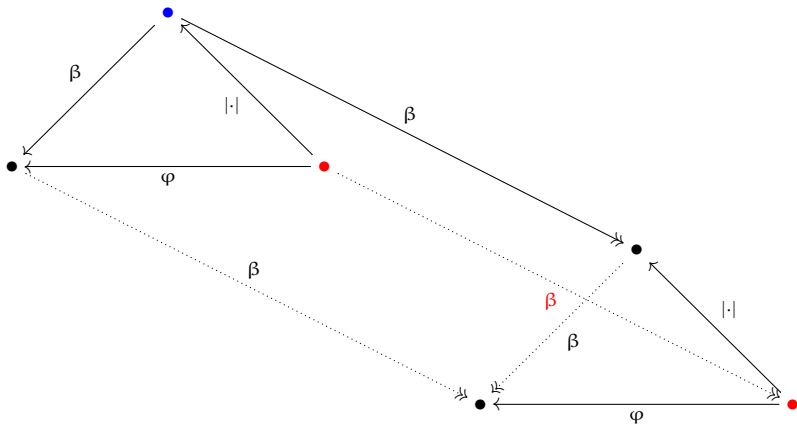
- Вводится операция *стирания подкраски*. Если $M \in \Lambda$, то $|M| \in \Lambda$ получается заменой в M всего красного на черное.
- Задается операция $\varphi : \Lambda \rightarrow \Lambda$, заключающаяся в сокращении всех подкрашенных редексов изнутри наружу:

$$\begin{aligned}\varphi(x) &= x, \\ \varphi((\lambda x. M) N) &= [x \mapsto \varphi(N)] \varphi(M), \\ \varphi(M N) &= \varphi(M) \varphi(N), \\ \varphi(\lambda x. M) &= \lambda x. \varphi(M).\end{aligned}$$

- Общая идея доказательства: пока жив, покрашенный редекс остается редексом.
- Он может размножиться, исчезнуть (по внешним причинам или сократившись), M и N могут измениться, но редекс не может «развалиться».

Доказательство леммы полосы

Три леммы: четырехугольники с красным и нижний треугольник.



Лемма полосы — грань без красного. ■

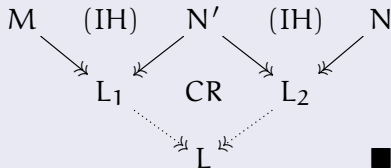
Следствия теоремы Чёрча-Россера (1)

Теорема о существовании общего редукта

Если $M =_{\beta} N$, то существует L , такой что, $M \rightarrow_{\beta} L$ и $N \rightarrow_{\beta} L$.

Доказательство (индукция по генерации $=_{\beta}$)

- $M =_{\beta} N$, поскольку $M \rightarrow_{\beta} N$. Возьмем $L \equiv N$.
- $M =_{\beta} N$, поскольку $N =_{\beta} M$. По гипотезе индукции имеется общий β -редукт L_1 для N, M . Возьмем $L \equiv L_1$.
- $M =_{\beta} N$, поскольку $M =_{\beta} N'$, $N' =_{\beta} N$. Тогда



Следствия теоремы Чёрча-Россера (2)

Теорема [Редуцируемость к NF]

Если M имеет N в качестве β -NF, то $M \rightarrow_{\beta} N$.

Теперь мы можем доказать отсутствие NF у Ω . Иначе выполнялось бы

$$\Omega \rightarrow_{\beta} N, \quad N \text{ является } \beta\text{-NF.}$$

Но Ω редуцируется лишь к себе и не является β -NF.

Теорема [Единственность NF]

λ -терм имеет не более одной β -NF.

Теперь мы можем доказывать «неравенства», например

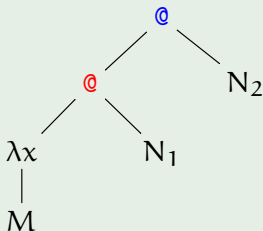
$$\lambda \not\approx \text{tru} = \text{fls}.$$

Иначе было бы $\text{tru} =_{\beta} \text{fls}$, но это две разные NF, что противоречит единственности.

- 1 Теорема о неподвижной точке
- 2 Редексы и нормальная форма
- 3 Теорема Чёрча-Россера
- 4 Стратегии редукции**

- Как мы можем редуцировать терм?
 - Переменная: v — редукция завершена.
 - Абстракция: $\lambda x. M$ — редуцируем M .
 - Аппликация: $M N$. Все варианты отсюда.
- Разбираем аппликацию до не-аппликации (обычно влево, то есть в M):
 - $(\dots ((v N_1) N_2) \dots N_k)$ — редуцируем отдельно все N_i (обычно слева направо).
 - $(\dots (((\lambda x. M) N_1) N_2) \dots N_k)$. Все варианты отсюда.
- **Нормальная стратегия:** сокращаем редекс $(\lambda x. M) N_1$.
- **Аппликативная стратегия:** редуцируем отдельно все N_i (обычно слева направо) до нормальной формы N'_i , затем сокращаем редекс $(\lambda x. M) N'_1$.
- Иногда аппликативной называют стратегию, когда только N_1 редуцируется до нормальной формы.

Пример: дерево для терма $((\lambda x. M) N_1) N_2$



- Узлы $@$ задают аппликацию, узлы λ — абстракцию.
- Узлы $@$ могут задавать редекс ($@$) или нет ($@$).
- В первом случае при поиске редекса — кандидата на сокращение есть три варианта (нашли, влево, вправо), во втором — два (влево, вправо).

Теорема

Лямбда-терм может иметь одну из двух форм:

$$\begin{aligned}\lambda \vec{x}. y \vec{N} &\equiv \lambda x_1 \dots x_n. y N_1 \dots N_k, \quad n \geq 0, k \geq 0 \\ \lambda \vec{x}. (\lambda z. M) \vec{N} &\equiv \lambda x_1 \dots x_n. (\lambda z. M) N_1 \dots N_k, \quad n \geq 0, k > 0\end{aligned}$$

Определение

Первая форма называется *головной нормальной формой* (HNF). Переменная y называется *головной переменной*, а редекс $(\lambda z. M) N_1$ — *головным редексом*.

Переменная y может совпадать с одной из x_i .

Определение

Слабая головная нормальная форма (WHNF) — это HNF или лямбда-абстракция, то есть *не редекс на верхнем уровне*.

Нормальная vs аппликативная стратегии

$$KI\Omega \equiv (\lambda x y. x) I \Omega \rightarrow_{\beta} I$$

$$KI\Omega \equiv KI((\lambda x. x x) (\lambda x. x x)) \rightarrow_{\beta} \dots$$

Теорема о нормализации

Если терм M имеет нормальную форму, то последовательное сокращение самого левого внешнего редекса приводит к этой нормальной форме.

- То есть **нормальная стратегия** гарантированно нормализует нормализуемое.
- Можем доказывать отсутствие NF. Например, $K\Omega I$.

Недостаток нормальной стратегии — возможная неэффективность, достоинство — не считает ничего «лишнего».

Нормальная vs аппликативная стратегии

- Пусть N — «большой» терм

$$(\lambda x. F x (G x) x) N \rightarrow_{\beta} F N (G N) N$$

В процессе дальнейших редукций редексы в N придётся сокращать три раза.

- Зато в

$$(\lambda x y. y) N \rightarrow_{\beta} \lambda y. y$$

нормальная стратегия не вычисляет N ни разу.

- Аппликативная стратегия в обоих примерах вычислит N один раз.

- Аппликативная стратегия похожа на стратегию вычислений («энергичную», eager) большинства языков программирования. Сначала вычисляются аргументы, затем происходит применение функции.
- Нормальная стратегия похожа на способ вычисления в «ленивых» (lazy) языках (Haskell, Clean).
- Для решения проблем с эффективностью в «ленивых» языках используют *механизм разделения* (через вычисления в контекстах или через редукцию на графах).

- Нет необходимости всегда доводить редукцию до NF. На практике часто ограничиваются WHNF.
- Это позволяет избежать захвата переменной при редукции *замкнутого* терма. (почему?)
- При наличии констант (в расширенных системах) понятие WHNF (и HNF) дополняют частично применёнными константными функциями, например

and true

поскольку его можно записать в η -эквивалентном WHNF-виде

$\lambda x. \text{and true } x$

- В Haskell к WHNF относят и конструктор данных, применённый полностью или частично.

- **Механизм вызова** — термин, применяемый при исследовании высокоуровневых языков программирования.
- В функциональных языках:
 - «вызов по значению» — аппликативный порядок редукций до WHNF;
 - «вызов по имени» — нормальный порядок редукций до WHNF;
 - «вызов по необходимости» — «вызов по имени» плюс разделение.