Functional Programming.
Colloquium

DistSys

2021

Содержание

1 λ-термы. β-редукция.

Def: λ-термы – это выражения, строящиеся из переменных путём применения операций:

- Если x — переменная, то x — терм.
- Применение: u и v — термы, то (uv) — терм.
- λ-абстракция: u — терм, x — переменная, то $(\lambda x.u)$ — терм.

Def: β-редукция – $(\lambda x.u)v \to_\beta u[x := v]$, то есть подстановка v вместо свободных x в u. При корректной подстановке ранее свободные переменные не становятся связанными.

2 Свойство Чёрча–Россера (конфлюэнтность).

Th. Свойство Чёрча–Россера (конфлюэнтность). Если $u \twoheadrightarrow_\beta v_1$ и $u \twoheadrightarrow_\beta v_2$, тогда $\exists w$, что $v_1 \twoheadrightarrow_\beta w$ и $v_2 \twoheadrightarrow_\beta w$.

Cor. Из-за свойства Чёрча-Россера, если НФ существует, то она единственна.

3 Нормальная форма. Сильная и слабая нормализуемость. Примеры термов с разными свойствами нормализуемости.

Def: Нормальная форма – терм без β-редексов (нельзя далее редуцировать).

Cor. Из-за свойства Чёрча-Россера, если НФ существует, то она единственна.

Def: Сильно нормализуемые термы – термы, которые приводимы к нормальной форме при любой последовательности (пути) редукций.

Def: Слабо нормализуемые термы – термы, которые приводимы к нормальной форме при одной последовательности (пути) редукций, а при другой редукции не завершаются (путь бесконечен).

Ex. $\omega = \lambda x.(xx)$, $\Omega = \omega\omega$. Терм Ω сводится редукциями к себе, следовательно он не нормализуем.

$$\Omega = \omega\omega = (\lambda x.(xx))(\lambda x.(xx)) \to_\beta (\lambda x.(xx))(\lambda x.(xx)) = \omega\omega = \Omega$$

Ex. $(\lambda x.y)\Omega$ является слабо нормализуемым. Если редуцируем сначала Ω, то попадаем в бесконечный цикл, иначе сразу получаем НФ.

Ex. $(\lambda x.x)y$ является сильно нормализуемым

4 Нормальная стратегия редукций, теорема об успешности нормальной стратегии.

Def: Один редекс левее другого – когда λ первого редекса расположена левее (в записи терма) λ второго редекса.

Это означает, что либо первый редекс целиком расположен левее второго, либо второй редекс находится внутри первого.

Def: Нормальная стратегия редукций – всегда редуцируй самый левый редекс.

При этом это не обязательно самая левая λ - левее могут быть лямбды, не образующие β-редексов.

Th. Успешность нормальной стратегии. Если терм можно привести к нормальной форме, то нормальная стратегия добьётся этого.

5 Комбинаторы неподвижной точки, пример: комбинатор Y.

Для достижения полноты по Тьюрингу требуется рекурсия. Мы не хотим, чтобы функция определялась через саму себя, поэтому требуется комбинатор неподвижной точки.

Def: Неподвижная точка функции f – такой терм F, что для функции f выполнено $F =_\beta fF$.

Символ $=_\beta$ означает β-эквивалентность – $a =_\beta b$, если $\exists c : a \to_\beta c \land b \to_\beta c$

Def: Комбинатор неподвижной точки – функция высшего порядка, вычисляющая неподвижную точку другой функции.

Ex (Y-комбинатор). Основное свойство $\mathbf{Y}f =_\beta f(\mathbf{Y}f)$.

$$\mathbf{Y} = \lambda f.\left((\lambda x.f(xx))(\lambda x.f(xx))\right)$$

Imp! Функция с **Y** комбинатором не может быть сильно нормализуемой, поэтому важна нормальная стратегия из вопроса 4.

Ex. Факториал можно выразить следующим образом:

$$\text{Fact} = \mathbf{Y}\left(\lambda g.\lambda x.(\text{if Zero } x \text{ then } 1 \text{ else } (g(\text{Prev } x) \cdot x))\right),$$

где Zero – проверка на 0, а Prev – предыдущее число (тут используются нумералы Чёрча).

6 Слабая головная нормальная форма.

Def: Слабая головная нормальная форма (WHNF) – это терм вида:

1. $\lambda x.u$;
2. $xv_1v_2 \dots v_n$, где x — переменная, а $v_1 \dots v_n$ – термы.

Def: thunk – недоредуцированный редекс.

Imp! Всякая НФ является СГНФ, но не наоборот.

Imp! В Haskell понятие СГНФ отличается.

Imp! СГНФ не вычисляет то, что может не пригодиться (в отличие от НФ):

1. функцию, которая еще не применена (λ снаружи);
2. переменные обозначающие неизвестные функции.

7 Исчисление типизации $\lambda_\to$ по Карри.

Def: Стрельчатый тип – функциональный тип $(A \to B)$.

Типы собираются из переменных по типам $(r_1, r_2, \ldots)$ и констант по типам $(p_1, \ldots)$ только с помощью операции $\to$ (A, B — типы, то $A \to B$ — тип).

Ограничение только на применение: (vu) корректно $\leftrightarrow$ u имеет тип A, а v имеет тип $A \to B$.

λ-абстракция может применяться всегда, образуя стрельчатый тип.

Типы переменных указываются в контексте $\Gamma = x_1 : A_1, x_2 : A_2, \ldots$.

Def: Типизация по Карри – полиморфная система типов. Терм в контексте может иметь много различных типов.

Imp! Типизация термов со свободными переменными осуществляется в контексте.

Def: Утверждение о типизуемости – запись вида $\Gamma \vdash u : B$, которая обозначает, что B – допустимый тип для u в контексте Γ.

Эти утверждения могут доказываться.

Терм u **не типизируем в** Γ, если не доказуемо $\Gamma \vdash u : B$ ни для какого B.

Для доказательства типизируемости используются следующие правила исчисления типизации по Карри.

$$\frac{}{\Gamma, x : A \vdash x : A} Ax \text{ аксиома}$$

$$\frac{\Gamma, x : A \vdash u : B}{\Gamma \vdash (\lambda x.u) : (A \to B)} Abs$$

$$\frac{\Gamma \vdash u : (A \to B) \qquad \Gamma \vdash v : A}{\Gamma \vdash (uv) : B} App$$

То что ниже, наверное, можно и не рассказывать.

Th. О нормализуемости Любой типизируемый терм сильно нормализуем.

Cor. Комбинатор неподвижной точки $\mathbf{Y}$ в $\lambda_\to$ не типизируем. Это можно исправить, если ввести константу $\mathbb{Y}$ с полиморфным типом $(r \to r) \to r$ и σ-редукцию $\mathbb{Y}u \to_\sigma u(\mathbb{Y}u)$.

Однако возникнет проблема, что его потребуется включить в контекст, где r зафиксируется.

8 Контекст типизации. Понятие наиболее общего типа для терма u в контексте Γ. Примеры: комбинаторы B и K.

Def: Контекст – множество $x_i : A_i$, т.е. присвоений типов для свободных переменных.

Def: Bσ – применение подстановки $\sigma = [r_1 := A_1, \ldots]$ к типу B.

Тип B — более общий, а $B\sigma$ — более конкретный.

Def: B_0 – наиболее общий тип для u_0 в контексте Γ_0, если:

1. $\Gamma_o \vdash u_0 : B_0$;
2. $\Gamma_o \vdash u_0 : B \Rightarrow \exists \sigma : B = B_0\sigma$.

Th. Для любых Γ_0 и u_0 либо существует наиболее общий тип для u_0 контексте Γ_0, либо u_0 не типизуем в контексте Γ_0.

Cor. Задача выведения типов = задача поиска наиболее общего типа.

Def: Комбинатор – замкнутые (без свободных переменных) термы чистого λ-исчисления.

Комбинатор $B = \lambda fgx.f(gx)$. НОТ: $(B \to A) \to ((C \to B) \to (C \to A))$.

Комбинатор $K = \lambda x.\lambda y.x$. НОТ: $A \to (B \to A)$.

Подходят $\text{Char} \to (\text{Bool} \to \text{Char})$, $(\text{Int} \to \text{Bool}) \to (\text{Char} \to (\text{Int} \to \text{Bool}))$, ...

Проверьте. Я не совсем понял, что от нас хотят услышать про комбинаторы.

9 Безопасность типов при β-редукции.

Def: Свойство безопасности типов при β-редукции – если $\Gamma \vdash u : B$ и $u \twoheadrightarrow_\beta u'$, то $\Gamma \vdash u' : B$

Это означает, что в процессе вычислений можно не контролировать типы, достаточно (статической) проверки в начале.

Imp! В обратную сторону условие не гарантируется: если $u \twoheadrightarrow_\beta u'$ и $\Gamma \vdash u' : B$, не обязательно $\Gamma \vdash u : B$. Может оказаться, что u вообще не типизируем, либо у него меньше корректных типов, чем у u'.

Ex. Возможно всё-таки стоит привести пример когда обратное условие не выполняется. Если кто-то придумает такой пример - напишите его сюда пожалуйста.

Думаю, что подойдет $(\lambda x.y)\Omega$ про не типизируемость и $(\lambda ft.(\lambda xy.y)f(42)t))$ для меньшего числа корректных типов, т.к. после редукции внутреннеко редекса f может быть любым термом, а не только $Int->smth$

10 Исчисление типизации $\lambda 2$ (система F). Пример типизуемого терма, не имеющего наиболее общего типа.

Def: Система F – система типов поверх λ-исчисления.

Типы получаются из базовых конструкциями $A \to B$ и $\forall r.A$.

Правила типизации

$$\frac{}{\Gamma, x : A \vdash x : A} Ax; \quad \frac{\Gamma, x : A \vdash u : B}{\Gamma \vdash (\lambda x.u) : (A \to B)} Abs; \quad \frac{\Gamma \vdash u : (A \to B) \quad \Gamma \vdash v : A}{\Gamma \vdash (uv) : B} App;$$

$$\frac{\Gamma \vdash u : A}{\Gamma \vdash u : (\forall r.A)} Gen; \quad \frac{\Gamma \vdash u : (\forall r.B)}{\Gamma \vdash u : B[r := A]} Inst;$$

Imp! Позволяет применить функцию к себе $x : \forall r.(r \to r) \vdash (xx) : \forall r.(r \to r)$, но все еще все типизируемые термы обладают свойством сильной нормализуемости, а следовательно $\Omega = (\lambda x.(xx))(\lambda x.(xx))$ не типизируем.

Ex. Рассмотрим терм $\lambda x.(xx)$.

У него есть две несравнимых типизации: $(\forall r.(r \to r)) \to (\forall r.(r \to r))$ и $(\forall r.((r \to r) \to (r \to r))) \to (\forall r.((r \to r) \to (r \to r)))$. Например, $\lambda f.\lambda g.\lambda x.fgx$ имеет тип 2, но не тип 1. Функция $\lambda f.\lambda z.(\lambda xz.z)(f0)$ – наоборот. Следовательно у терма нет наиболее общего типа в системе F.

Задача типизации в системе F алгоритмически неразрешима.

11 Система типов Хиндли–Милнера. Let-полиморфизм.

В предыдущих пунктах $\lambda_{\to}$ не позволяла адекватно реализовать полиморфизм, а F не имела наиболее общего типа. Система типов Хиндли–Милнера является чем-то «между» двумя системами выше. По сути в F добавили ограничение + let in.

Def: Система типов Хиндли–Милнера – система типов поверх λ-исчисления.

Типы — безкванторный (мономорфные) типы $\lambda_{\to}$ и типы $\forall r_1.\forall r_2.\dots.\forall r_k.A$, где A – мономорфный тип. λ-абстракция разрешена только для мономорфных типов.

Также система Х-М вводит новый конструктор термов **let in**, например **let** x = v **in** u, и редукцию **let** x = v **in** u → u [x := v]. Без типов это равносильно $(\lambda x.u)v$ (в типах в let у x может быть полиморфный тип).

Правила типизации

$$\frac{}{\Gamma, x : A \vdash x : A} Ax; \quad \frac{\Gamma \vdash u : (A \to B) \quad \Gamma \vdash v : A}{\Gamma \vdash (uv) : B} App;$$

$$\frac{\Gamma, x : A \vdash u : B}{\Gamma \vdash (\lambda x.u) : (A \to B)} Abs \text{ – только для безкванторного типа } A;$$

$$\frac{\Gamma \vdash v : A \quad \Gamma, x : A \vdash u : B}{\Gamma \vdash (\textbf{let } \mathrm{x} = \mathrm{v} \textbf{ in } \mathrm{u}) : B} Let$$

$$\frac{\Gamma \vdash u : A}{\Gamma \vdash u : (\forall r.A)} Gen; \quad \frac{\Gamma \vdash u : (\forall r.B)}{\Gamma \vdash u : B[r := A]} Inst \text{ – если подстановка корректна } A;$$

Ex. $\lambda f.(ftrue, f0)$ не типизируется, т.к. f — мономорфный.

$letf = \lambda x.xin(ftrue, f0)$ типизируется.

Imp! В системе Х-М у типизируемого терма существует наиболее общий тип, а задача его нахождения разрешима.

Imp! Система типов Haskell является расширенной системой типов Хиндли–Милнера (Алг. типы, классы типов, рекурсивный let).

12 Полезные материалы.

Основная страница курса: `http://www.mi-ras.ru/~sk/lehre/fp_hse2021/`.

Немного конспектов за 2020 год: `http://www.mi-ras.ru/~sk/lehre/fp_hse2020/`

Теория за 2018 год: `http://www.mi-ras.ru/~sk/lehre/fp_hse2018/`

13 Раздел для разработчиков.

Def: Автомобиль – дом на колёсах.

Ex. Квадрат является примером прямоугольника.

Imp! Важный момент, который нельзя забывать!

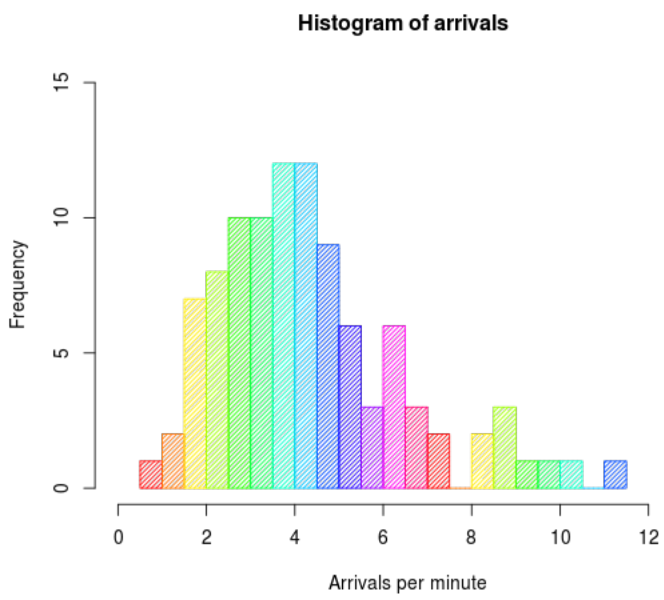
Th. О равенстве полов Пифагоровы штаны во все стороны летят.

Доказательство. Помахали руками, готово. **[:|||:]**

Cor. Следует из теоремы.

St. Утверждение (можно доказать).

Пример того, как вставить картинку:



Пример гистограммы.

Пример таблицы:

	$\dots$	Δ_i	$\dots$
H	$\dots$	n_i	$\dots$
O	$\dots$	$n\,p_i$	$\dots$