Вопросы к зачету

Ilya Yaroshevskiy

14 октября 2023 г.

Содержание

1 Бестиповое лямбда-исчисление

Определение. Пред λ-терм:

1. x — переменная
2. $L\ L$ — аппликация
3. $\lambda x.L$ — абстракция

2 Общие определения (альфа-эквивалентность, бета-редукция, бета-эквивалентность)

Определение. α-эквивалентность. $A =_\alpha B$

1. $A \equiv x,\ B \equiv x$ — одна и та же переменная
2. $A \equiv P\ Q,\ B \equiv R\ S \quad P =_\alpha R,\ Q =_\alpha S$
3. $A \equiv \lambda x.P,\ B \equiv \lambda y.Q$. Существует t — новая переменная, что $P[x := t] =_\alpha Q[y := t]$

Определение.

- $\Gamma \vdash A : \tau$ (ΓA^τ), где Γ — $\{x_1 : \tau_1,\ x_2 : \tau_2, \dots\}$
- правила:

1. $$\frac{}{\Gamma, x_1 : \tau_1 \vdash x_1 : \tau_1} \quad x_1 \notin Fv(\Gamma) \quad \text{(Ax.)}$$
2. $$\frac{\Gamma \vdash A : \sigma \to \tau \quad \Gamma \vdash B : \sigma}{\Gamma \vdash A\ B : \tau}$$
3. $$\frac{\Gamma, x : \tau A : \sigma}{\Gamma \vdash \lambda x.A : \tau \to \sigma}$$

Определение. β-эквивалентность ($=_\beta$) — транзитивное, рефлексивное, симметричное замыкание отношения β-редукции.

3 Параллельная бета-редукция

Определение. Параллельная β-редукция ($\rightrightarrows_\beta$) это ($\to_\beta$)+правило:

0. $A =_\alpha B$

1. $A \equiv P\ Q,\ B \equiv R\ S$ и $P \rightrightarrows_\beta R$ и $Q \rightrightarrows_\beta S$

2,3. аналогично ($\to_\beta$)

4 Теорема Чёрча-Россера

Определение. β**-редуцируемость** ($\twoheadrightarrow_\beta$) — рефлексивное, транзитивное замыкание ($\to_\beta$)

Определение. Будем говорить что отношение R обладает **ромбовидным** свойством, если для любых a, b, c

1. $aRb,\ aRc$
2. $b \neq c$

Существует d: $bRd,\ cRd$

Теорема 4.1 (Черча-Россера)**.** Если $\Gamma \vdash A : \tau$, то любое подвыражение имеет тип

5 Нормальный и аппликативный порядок редукций

Определение. Нормальный порядок редукции — редукция самого левого β-редекса

Определение. Аппликативный порядок редукции — редукция самого левого β-редекса из самых вложенных

6 Y-комбинатор

Определение. Y**-комбинатор** $\coloneqq \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$

7 Нетипизируемость Y-комбинатора

Теорема 7.1. Y-комбинатор не типизируется в просто типизированном по Карри λ-исчислении

8 Слабая и сильная нормализация

Определение. Терм называется **слабо-нормализуемым**, если существует последовательность β-редукция, приводящая его к нормальной форме

Определение. Терм — **сильно-нормализуем**, если не существует бесконечной последовательность β-редукций, не приводящая к нормальной форме

9 Парадокс Карри, парадокс при интерпретации бестипового лямбда-исчисления как логики

9.1 Парадокс Карри

Попробуем построить логику на основе λ-исчисления. Введем логический символ $\supset$. Будем требовать от этого исчисления наличия следующий аксиом:

1. $\vdash A \supset A$
2. $\vdash (A \supset (A \supset B)) \supset (A \supset B)$
3. $\vdash A =_\beta B$, тогда $A \supset B$

А также правила вывода MP:

$$\frac{\vdash A \supset B \quad \vdash A}{\vdash B}$$

Не вводя дополнительные правила вывода и схемы аксиом, покажем, что данная логика является противоречивой. Для чего введем следующие условные обозначения:

- $F_\alpha \equiv \lambda x.(x\ x) \supset \alpha$
- $\Phi_\alpha \equiv F_\alpha\ F_\alpha \equiv (\lambda x.(x\ x) \supset \alpha)\ (\lambda x.(x\ x) \supset \alpha)$

Редуцируя Φ_α получаем

$$\begin{aligned}\Phi_\alpha &=_\beta (\lambda x.(x\ x) \supset \alpha)\ (\lambda x.(x\ x) \supset \alpha)\\ &=_\beta (\lambda x.(x\ x) \supset \alpha)\ (\lambda x.(x\ x) \supset \alpha) \supset \alpha\\ &=_\beta \Phi_\alpha \supset \alpha\end{aligned}$$

Теперь докажем противоречивость введенной логики. Для этого докажем, что в ней выводимо любое утверждение:

1)	$\vdash \Phi_\alpha \supset \Phi_\alpha \supset \alpha$	$\vdash \Phi_\alpha =_\beta \Phi_\alpha \supset \alpha$
2)	$\vdash (\Phi_\alpha \supset \Phi_\alpha \supset \alpha) \supset (\Phi_\alpha \supset \alpha)$	$\vdash (A \supset (A \supset B)) \supset (A \supset B)$
3)	$\vdash \Phi_\alpha \supset \alpha$	MP $2, 1$
4)	$\vdash (\Phi_\alpha \supset \alpha) \supset \Phi_\alpha$	$\vdash \Phi_\alpha \supset \alpha =_\beta \Phi_\alpha$
5)	$\vdash \Phi_\alpha$	MP $4, 3$
6)	$\vdash \alpha$	MP $3, 5$

Таким образом, введенная логика оказывается противоречивой

10 Импликационный фрагмент ИИВ

Определение. Импликационный фрагмент ИИВ — оставляем правила $I_\to, E_\to, Ax$

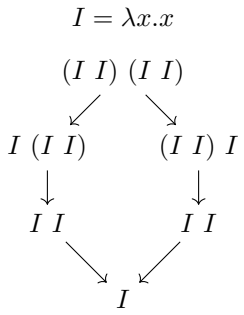
11 Теорема о замкнутости И.Ф. относительно доказуемости

Обозначение. $\Gamma \vdash_\to \tau$ — доказуемость в И.Ф. ИИВ

Теорема 11.1. Импликационный фрагмент ИИВ замкнут относительно доказуемости. Если $\Gamma \vdash \tau$ и τ содержит только $\alpha, \to$, то $\Gamma \vdash_\to \tau$. И если $\Gamma \vdash_\to \tau$, то $\Gamma \vdash \tau$

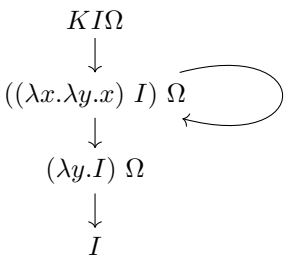
12 Комбинаторы, базис SKI, его аналог в логике

Пример. Комбинатор I (Identity). Доказательство того, что $(\to_\beta)$ не обладает ромбовидным свойством:



[org:

$$I = \lambda x.x \quad K = \lambda x.\lambda y.x \quad \Omega = \omega\ \omega \quad \omega = \lambda x.x\ x$$



]

Пример.

$$\Omega = (\lambda x.x\ x)\ (.x\ x)$$

Применимо только правило 2. Не имеет типа

Определение.

- $S \coloneqq \lambda x.\lambda y.\lambda z.x\ z\ (y\ z)$
- $K \coloneqq \lambda x.\lambda y.x$
- $I \coloneqq \lambda x.x$

Утверждение. *Любое λ-выражение без свободных перменных можно записать с помощью комбинаторов S и K, где:*

- $S = \lambda x.\lambda y.\lambda z.(x\ z)\ (y\ z) : (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma$
- $K = \lambda x.\lambda y.x : \alpha \to \beta \to \alpha$

Утверждение. *Комбинаторы S и K являются аксиомами ИИВ*

13 Просто типизированное лямбда-исчисление

Определение. Типовые переменные:

- α, β, γ — атомарные
- τ, σ — составные

Тип: $\tau ::= (\tau \to \tau) \mid \alpha$
Следует:

- 2 традиции:
  1. Исчисление по Черчу
  2. Исчисление по Карри

14 Исчисление по Чёрчу и по Карри

Определение. Исчисление по Карри:

- $\Gamma \vdash A : \tau$ ($\Gamma \vdash A^\tau$), где Γ — $\{x_1 : \tau_1,\ x_2 : \tau_2, \dots\}$
- правила:
  1. $$\frac{}{\Gamma, x_1 : \tau_1 \vdash x_1 : \tau_1} \quad x_1 \notin Fv(\Gamma) \quad \text{(Ax.)}$$
  2. $$\frac{\Gamma \vdash A : \sigma \to \tau \quad \Gamma \vdash B : \sigma}{\Gamma \vdash A\ B : \tau}$$
  3. $$\frac{\Gamma, x : \tau \vdash A : \sigma}{\Gamma \vdash \lambda x.A : \tau \to \sigma}$$

Определение. Исчисление по Чёрчу: $\Gamma \vdash_{\text{Ч}} A : \tau$
Язык:

- x — переменная
- $A\ B$ — аппликация
- $\lambda x^\tau.P$ — абстракция

[org: Отличие только в том, что в исчисление по Черчу типовые аннотации обязательны, это позволяет доказать следующую теорему.
]

Теорема 14.1. Если конекст Γ и выражение P типизируется, то $\Gamma \vdash_{\text{Ч}} P : \sigma$

Пример.

$$\vdash_C \lambda x.x : \substack{\alpha \to \alpha \\ \beta \to \beta}$$

$$\vdash_{\text{Ч}} \lambda x.x : \sigma \to \sigma$$

Пример.

$$\lambda f.\lambda x.f\ (f\ x) : \substack{(\alpha \to \alpha) \to (\alpha \to \alpha) \\ (\beta \to \beta) \to (\beta \to \beta)}$$

$$\lambda f^{\alpha \to \alpha}.\lambda x.^{\alpha}.f\ (f\ x) : (\alpha \to \alpha) \to (\alpha \to \alpha)$$

15 Изоморфизм Карри-Ховарда

Определение.

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \quad \frac{}{\Gamma, \tau \vdash \tau}$$

$$\frac{\Gamma \vdash A : \sigma \to \tau \quad \Gamma \vdash B : \sigma}{\Gamma \vdash A\ B : \tau} \quad \frac{\Gamma \vdash \sigma \to \tau \quad \Gamma \vdash \sigma}{\Gamma \vdash \tau}$$

$$\frac{\Gamma, x : \sigma \vdash A : \tau}{\Gamma \vdash \lambda x.A : \sigma \to \tau} \quad \frac{\Gamma, \sigma \vdash \tau}{\Gamma \vdash \sigma \to \tau}$$

Если $\Gamma \vdash A : \tau$, то

- A — эквиваленто доказательству τ
- τ — выражения в ИИВ
- Обитаемость типа — доказуемость τ
- Тип λ-абстракции — импилкация
- ...

16 Конъюнкция, дизъюнкция, ложь и соответствующие им конструкции в лямбда-исчислении

Определение. Введем тип $\bot$:

$$\frac{\Gamma \vdash A : \bot}{\Gamma \vdash A : \tau}$$

И остальные конструкции и правила для них:

- $\varphi \& \psi$
  - $\text{mkPair} \coloneqq \lambda x.\lambda y.\lambda f.f\ x\ y$, тогда $\langle x, y\rangle = \text{mkPair}\ x\ y$
  - $\pi_1 \coloneqq \lambda p.p\ (\lambda a.\lambda b.a)$
  - $\pi_2 \coloneqq \lambda p.p\ (\lambda a.\lambda b.b)$

$$\frac{\Gamma \vdash A : \varphi \quad \Gamma \vdash B : \psi}{\Gamma \vdash \langle A, B\rangle : \varphi \& \psi} \quad \frac{\Gamma \vdash \langle A, B\rangle : \varphi \& \psi}{\Gamma \vdash \pi_1\ \langle A, B\rangle : \varphi} \quad \frac{\Gamma \vdash \langle A, B\rangle : \varphi \& \psi}{\Gamma \vdash \pi_2\ \langle A, B\rangle : \psi}$$

- $\varphi \vee \psi$
  - $\text{in}_L \coloneqq \lambda x.\lambda f.\lambda g.f\ x$
  - $\text{in}_R \coloneqq \lambda x.\lambda f.\lambda g.g\ x$
  - $\text{case} \coloneqq \lambda a.\lambda f.\lambda g.a\ f\ g$

$$\frac{\Gamma \vdash A : \varphi}{\Gamma \vdash \text{in}_L\ A : \varphi \vee \psi} \quad \frac{\Gamma \vdash A : \psi}{\Gamma \vdash \text{in}_R\ A : \varphi \vee \psi} \quad \frac{\Gamma \vdash L : \varphi \vee \psi \quad \Gamma \vdash f : \varphi \to \tau \quad \Gamma \vdash g : \psi \to \tau}{\Gamma \vdash \text{case}\ L\ f\ g : \tau}$$

17 Чёрчевские нумералы

Определение. **λ-трем**. $\Lambda/_{=_\alpha}$ — множество λ-термов

18 Теорема о выразительной силе просто типизированного лямбда-исчисления (формулировка)

Теорема 18.1. Просто типизируемое λ-исчисление сильно нормализуемо. Поэтому любое выражение редуцирется за конечное число шагов.

19 Алгебраические термы

Определение.

$$T ::= a | (f\ T_1 \dots T_n)$$

Пример.

$$f_1\ (f_2\ a\ b)\ c$$

19.1 Подстановка переменных

Определение. Подстановка:

- $S_0 : V \to T$ — тождественно почти всюду (кроме конечного количества)
- $T_1 = T_2$ — **уравнение**
  Решение: такая подстановка S, что $S(T_1) \equiv S(T_2)$

Пример.

$$f\ a\ (g\ c) = f\ (g\ d)\ b$$

Положим:

- $S_0(a) = g\ d$
- $S_0(b) = g\ c$

$$S_0(f\ a\ (g\ c)) = f\ (g\ d)\ (g\ c) = S_0(f\ (g\ d)\ b)$$

20 Задача унификации в алгебраических термах

Задача приведения системы уравнений в алгебраических термах к разрешенному или несовместному виду.

21 Алгоритм унификации

Определение. Несовместные система — система с уравнением вида:

$$f\ T_1 \dots T_k = g\ P_1 \dots P_n$$

, где $k \neq n$, либо $f \neq g$ либо $x = \dots\ x\ \dots$.

21.1 Алгоритм унификации

Рассмотрим систему $\begin{cases} T_1 = P_1 \\ \vdots \\ T_n = P_n \end{cases}$:

1. $x = x$ — отбрасываем
2. $T = x$, где $T \not\equiv x \implies x = T$
3. $$\begin{cases} x = P \\ \vdots \\ T_2 = P_2 \\ \vdots \\ T_n = P_n \end{cases} \implies \begin{cases} T_2[x := P] = P_2[x := P] \\ \vdots \\ T_n[x := P] = P_n[x := P] \\ \vdots \\ x = P \end{cases}$$

4. $f\ T_1\ \ldots\ T_n = f\ P_1\ \ldots\ P_n \implies \begin{cases} T_1 = P_1 \\ \vdots \\ T_n = Pn \end{cases}$

Теорема 21.1. Применяя шаги алгоритма унификации, за конечное время можно получить систему либо в разрешенной форме, либо несовместной

22 Наиболее общее решение задачи унификации

Определение. $U \circ T$:

$$(U \circ T)(P) = U(T(P))$$

Определение. Определим порядок на подстановках: $S \leq T$, если S — частный случай T: существует подстановка U, что $S = U \circ T$

Определение. Наиболее общим решением $T = P$ назовем подстановку S, что для любой S_1: $S_1(T) \equiv S_1(P)$ выполнено $S_1 \leq S$ и $S(T) = S(P)$

23 Задачи проверки типа, реконструкции (вывода) типа, обитаемости типа в просто типизированном лямбда-исчислении, их аналоги в ИИВ

[org: $? \vdash ? : ?$. Три задачи

1. Обитаемость типа: $\Gamma \vdash ? : \tau$
   По изоморфизму К-Х и теореме(о замкнутости И.Ф. относительно доказуемости) эквивалентно $\Gamma \vdash \tau$
2. Вывод типа(реконструкция): $\Gamma \vdash A : ?$
3. Проверка типов: $\Gamma \vdash A : \tau$
   Вывод типа и проверка что одно и то же

1 — полнота; 2,3 — разрешимость
]

24 Алгоритм нахождения типа в просто типизированном лямбда-исчислении

[org: $(\rightarrow)$ — двуместный функциональный символ
Индукция по структуре λ-выражения

1. x — введем тип α_x.
   Система: $\varnothing$
   Тип: α_x
2. $A\ B$ — рекурсивный вызов, $\langle E_A, \tau_A \rangle, \langle E_B, \tau_B \rangle$
   Система: $E_A \cup E_B \cup \{\tau_B \rightarrow \beta = \tau_A\}$
   Тип: β
3. $\lambda x.A$ — рекурсивный вызов $\langle E_A, \tau_A \rangle$
   Система: E_A
   Тип: $\alpha_x \rightarrow \tau_A$

Разрешение системы: унификация (перепишем $\alpha \rightarrow \beta$ в алгебраических термах — $\rightarrow\ \alpha\ \beta$)
]

25 Наиболее общий тип, наиболее общая пара

Определение. Пусть α и β — два типа в $\lambda_{\to}$. Тогда α — **частный случай** β, если существует подстановка U, что $U(\beta) = \alpha$.

Определение. τ — **наиболее общий тип** в некотором семействе типов T, если $\forall x \in T$, x — частный случай τ.

26 Логика второго порядка

Определение. Логика 2 порядка

$$\Phi_{\Pi} ::= p \mid \Phi_{\Pi} \vee \Phi_{\Pi} \mid \Phi_{\Pi} \& \Phi_{\Pi} \mid \Phi_{\Pi} \to \Phi_{\Pi} \mid \forall p.\Phi_{\Pi} \mid \exists p.\Phi_{\Pi} \mid \ \bot$$

27 Выразимость связок через импликацию и квантор всеобщности в интуиционистской логике 2-го порядка (конъюнкция, дизъюнкция, ложь, отрицание, квантор существования)

Утверждение. *Можно выразить:*

- $a \& b \coloneqq \forall p.(a \to b \to p) \to p$
- $a \vee b \coloneqq \forall p.(a \to p) \to (b \to p) \to p$
- $\bot \coloneqq \forall p.p$
- $\exists p.A \coloneqq \forall x.(\forall p.A \to x) \to x$, *($\exists p.A \approx \neg\forall p.\neg A$)*

28 TODO Простая модель для логики второго порядка

29 Система F

Определение. L_2 – лямбда исчисление 2 порядка (Система F)

$$L_2 ::= x \mid \lambda x^{\alpha}.A \mid P\ Q \mid \Lambda\alpha.A \mid P\ \tau$$

[org:]

$$\frac{\Gamma \vdash A : \varphi}{\Gamma \vdash \Lambda\alpha.A : \forall\alpha.\varphi}$$

$$\frac{\Gamma \vdash A : \forall\alpha.\varphi}{\Gamma \vdash A\ \pi : \varphi[\alpha \coloneqq \pi]}$$

]

30 TODO Изоморфизм Карри-Ховарда для системы F: квантор всеобщности, упорядоченные пары, алгебраические типы

Может это?

31 Экзистенциальные типы

Определение. $\exists\alpha.\ \underbrace{\varphi}_{\text{интерфейс}}$

[org: Правила?

$$\frac{\Gamma \vdash \varphi[\alpha := \theta]}{\Gamma \vdash \exists\alpha.\varphi}$$

$$\frac{\Gamma \vdash \exists\alpha.\varphi \quad \Gamma, \varphi \vdash \psi}{\Gamma \vdash \psi}$$

]

Пример. Стэк:

$$\texttt{push} : \alpha \to \alpha\ \texttt{stack} \to \alpha\ \texttt{stack}$$

$$\texttt{pop} : \alpha\ \texttt{stack} \to (\alpha \cdot \alpha\ \texttt{stack})$$

$$\texttt{empty} : \alpha\ \texttt{stack}$$

$$\exists\alpha.(\eta \to \alpha \to \alpha)\&(\alpha \to \alpha\&\eta)\&\alpha$$

32 Конструкции `pack` и `abstype`

[org:

$$\textbf{pack}\ \tau, M\ \textbf{to}\ \exists\alpha.\sigma \equiv \Lambda\beta.\lambda x^{\forall\alpha.\sigma\to\beta}.x\ \tau\ M$$

$$\textbf{abstype}\ \tau\ \textbf{with}\ x : \sigma\ \textbf{is}\ M\ \textbf{in}\ N : \rho \equiv M\ \rho\ (\Lambda\tau.\lambda x^{\sigma}.N)$$

]

Пример.

$$\delta \equiv \underbrace{(\eta \to \alpha \to \alpha)}_{\text{push}}\ \&\ \underbrace{(\alpha \to (\eta\&\alpha))}_{\text{pop}}\ \&\ \underbrace{\alpha}_{\text{empty}}$$

abstype τ **with** $x : \delta$
is (**pack** τ, (push, pop, empty) **to** $\exists\alpha.\delta$)
in $\pi_l\ (x_2\ (x_1\ 5\ x_3))$
Где, например, $\tau = [\eta$ List], push = `cons`, pop = λ`(x:xs)` $\to$ `(x, xs)`, empty = `[]`

Пример.

$$\texttt{set} : \text{bool} \to \alpha$$

$$\texttt{isTrue} : \alpha \to \text{bool}$$

$$\delta \equiv (\text{bool} \to \alpha)\&(\alpha \to \text{bool})$$

$$\xi = (\text{bool} \to \text{bool})\&(\text{bool} \to \text{bool})$$

$$\Gamma \vdash \underbrace{\delta[\alpha := \text{bool}]}_{\xi}$$

abstype bool **with** $x : \delta$
is (**pack** bool, $\underbrace{(\lambda x^{\text{bool}}.x, \lambda x^{\alpha}.x)}_{\xi}$ **to** $\exists\alpha.\delta$)
in $x_2\ (x_1\ \text{true}) \twoheadrightarrow_\beta \text{true}$

33 TODO Абстрактные типы данных

Определение.

$$\frac{\Gamma \vdash \varphi[x \coloneqq \Theta]}{\Gamma \vdash \exists x.\varphi}$$

$$\frac{\Gamma, \psi \vdash \varphi \quad \Gamma \vdash \exists x.\psi}{\Gamma \vdash \varphi} \quad x \notin FV(\Gamma)$$

$$\frac{\Gamma \vdash M : \delta[\alpha \coloneqq \tau]}{\Gamma \vdash \textbf{pack } \tau, M \textbf{ to } \exists\alpha.\delta : \exists\alpha.\delta}$$

$$\frac{\Gamma, x : \delta \vdash M : \rho \quad \Gamma \vdash S : \exists\alpha.\delta}{\Gamma \vdash \textbf{abstype } \alpha \textbf{ with } x : \delta \textbf{ is } S \textbf{ in } M : \rho} \quad x \notin FV(\Gamma)$$

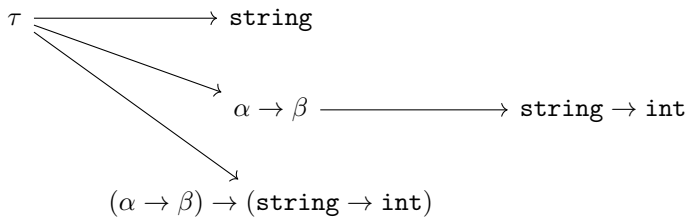
34 Ранг типа

Определение. Ранг типа. Пусть σ — тип без кванторов. $R \subseteq$ тип $\times \mathbb{N}$:

1. $R(\sigma, 0)$
2. Если $R(\tau, k)$, то $R(\forall\overline{\alpha}.\tau, \max(k, 1))$
3. Если $R(\tau_0, k)$ и $R(\tau_1, k+1)$, то $R(\tau_0 \to \tau_1, k+1)$

35 Частный случай типа

Определение. Отношение **“Быть частным случаем“** (специализация)



$\sigma_1 \sqsubseteq \sigma_2$ (σ_2 — частный случай σ_1), если:

- $\sigma_1 \equiv \forall\alpha_1 \ldots . \forall\alpha_n.\tau_1$
- $\sigma_2 \equiv \forall\beta_1 \ldots . \forall\beta_k.\tau_1[\alpha_1 \coloneqq \Theta_1, \ldots, \alpha_n \coloneqq \Theta_n]$

Новые переменные $\beta_1, \ldots, \beta_n$ не входят свободно в σ_1.

36 Типы и типовые схемы

См. Типовая система Хиндли-Милнера

37 Типовая система Хиндли-Милнера

Определение. Типовая система Хиндли-Милнера:
Рассмотрим λ-исчисление второго порядка по Карри. Типы:

1. типы(без кванторов) $\tau = \alpha | (\tau \to \tau)$
2. типовые схемы $\sigma = \forall\alpha.\sigma | \tau$

Определение.

$$\frac{}{\Gamma, x : \sigma \vdash x : \sigma} \quad x \notin FV(\Gamma)$$

$$\frac{\Gamma \vdash A : \tau \to \tau' \quad \Gamma \vdash B : \tau}{\Gamma \vdash A\ B : \tau'}$$

$$\frac{\Gamma, x : \tau \vdash A : \tau'}{\Gamma \vdash \lambda x.A : \tau \to \tau'}$$

$$\frac{\Gamma \vdash A : \sigma \quad \Gamma, x : \sigma \vdash B : \tau}{\Gamma \vdash \texttt{let}\ x = A\ \texttt{in}\ B : \tau}$$

$$\frac{\Gamma \vdash A : \sigma'}{\Gamma \vdash A : \sigma} \quad \sigma' \sqsubseteq \sigma$$

$$\frac{\Gamma \vdash A : \sigma}{\Gamma \vdash A : \forall\alpha.\sigma} \quad \alpha \notin FV(\Gamma)$$

38 Алгоритм W

Определение. Хотим решить $? \vdash A :?$, при чем найти наиболее общий тип. $\mathcal{V}$ — вызов алгоритма унификации

$$W(\Gamma, E) \Rightarrow (\tau, S)$$

1. $E \equiv x$, $x \in \Gamma$, $x : \sigma_x$
   Новые переменные $\beta_1, \ldots, \beta_n$, $\sigma_x = \forall\alpha_1 \ldots \alpha_n.\tau$

   $$(\forall\beta_1 \ldots \forall\beta_n.\tau, \varnothing)$$

2. $E \equiv \lambda x.P$

   $$W(\Gamma, E) = (S_1(\gamma \to \tau_P), S_1)$$
   $$W(\Gamma \cup \{x : \gamma\}, P) = (\tau_P, S_1)$$

3. $E \equiv P\ Q$

   $$W(\Gamma, E) = (S_3\gamma, S_3 \circ S_2 \circ S_1)$$
   $$W(\Gamma, P) = (\tau_P, S_1)$$
   $$W(S_1\Gamma, Q) = (\tau_Q, S_2)$$
   $$\mathcal{V}(S_2\tau_P, \tau_Q \to \gamma) = S_3$$

4. $E \equiv \texttt{let}\ x = P\ \texttt{in}\ Q$

   $$W(\Gamma, E) = (\tau_Q, S_2 \circ S_1)$$
   $$W(\Gamma, P) = (\tau_P, S_1)$$
   $$W(S_1\Gamma \cup \{x : \forall.\tau_f\}, Q) = (\tau_Q, S_2)$$

   , где $\forall.\tau_f$ — кванторы по всем свободным перменным из τ_f

39 Типизация Y-комбинатора

Определение. $Y : \forall\alpha.(\alpha \to \alpha) \to \alpha$

40 Экви- и изорекурсивные типы, μ-оператор, `roll` и `unroll`

[org:

```haskell
data IntList = Nil | Cons Int IntList
```

Есть два способа разрешения рекурсивных типов:

1. **Эквирекурсивный**:
   Введем оператор аналогичный Y-комбинатору, только на типах: $\mu\alpha.f(\alpha) = f(\mu\alpha.f(\alpha))$
   Пример: выведем тип $\lambda x.x\ x$:
   Пусть $\tau = \mu\alpha.\alpha \to \beta$. Если мы раскроем τ один раз, то получим $\tau = \tau \to \beta$. Если раскроем еще раз, то получим $\tau = (\tau \to \beta) \to \beta$.

   $$\dfrac{\dfrac{\dfrac{}{x:\tau \vdash x:\tau}}{x:\tau \vdash x:\tau \to \beta} \quad \dfrac{}{x:\tau \vdash x:\tau}}{\dfrac{x:\tau \vdash x\ x:\beta}{\vdash \lambda x.x\ x:\tau \to \beta}}$$

   Также можем доказать $Y \equiv \lambda f.(\lambda x.f\ (x\ x))\ (\lambda x.f\ (x\ x))$.

2. **Изорекурсивный**:
   Будем считать, что $\mu\alpha.f(\alpha)$ изоморфно $\grave{(}\mu\alpha.f(\alpha))$. Такой подход используется в `C`:

   ```c
   struct list {
     list* x;
     int a;
   };
   ```

   Можем использовать так: `x->x->x->a`. Заметим, что мы неявно использовали разыменование: `*` : `list*` $\to$ `list`. В изорекурсивных типах введены специальные операции для работы с этими типами:

   - roll : `list*` $\to$ `list`
   - unroll : `list` $\to$ `list*`

   В более общем виде (введение в типовую систему):

   - roll : $f(\alpha) \to \alpha$
   - unroll : $\alpha \to f(\alpha)$

]

41 TODO Примеры конструкций и операторов в языках программирования

42 Обобщённые типовые системы

Определение. Обобщенные типовые системы

$$\mathcal{F} = x \mid \mathcal{F}\ \mathcal{F} \mid \lambda x:\mathcal{F}.\mathcal{F} \mid \Pi x:\mathcal{F}.\mathcal{F} \mid c$$

$$c = * \mid \square$$

Обозначение. Обозначим за s множество $(*, \square)$

Обозначение. $x:y:z \implies x:y$ и $y:z$

Определение.

- Аксиома

$$\frac{}{\vdash * : \square}$$

- Начальное правило

$$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A} \quad x \notin FV(\Gamma)$$

- Применение

$$\frac{\Gamma \vdash \varphi : (\Pi x : A.B) : s \quad \Gamma \vdash a : A}{\Gamma \vdash (\varphi\ a) : (B[x \coloneqq A])}$$

- Conversion (преобразование)

$$\frac{\Gamma \vdash A : B \quad \Gamma \vdash B' : s \quad B =_\beta B'}{\Gamma \vdash A : B'}$$

- Ослабление

$$\frac{\Gamma \vdash B : C \quad \Gamma \vdash A : s}{\Gamma, x : A \vdash B : C} \quad x \notin FV(\Gamma)$$

43 Типы, рода, сорта

[org:

$$\underset{\substack{\text{знач.}\\ value}}{7} : \underset{\substack{\text{тип}\\ type}}{int} : \underset{\substack{\text{род}\\ kind}}{*} : \underset{\substack{\text{сорт}\\ sort}}{\square}$$

]

44 Лямбда-куб

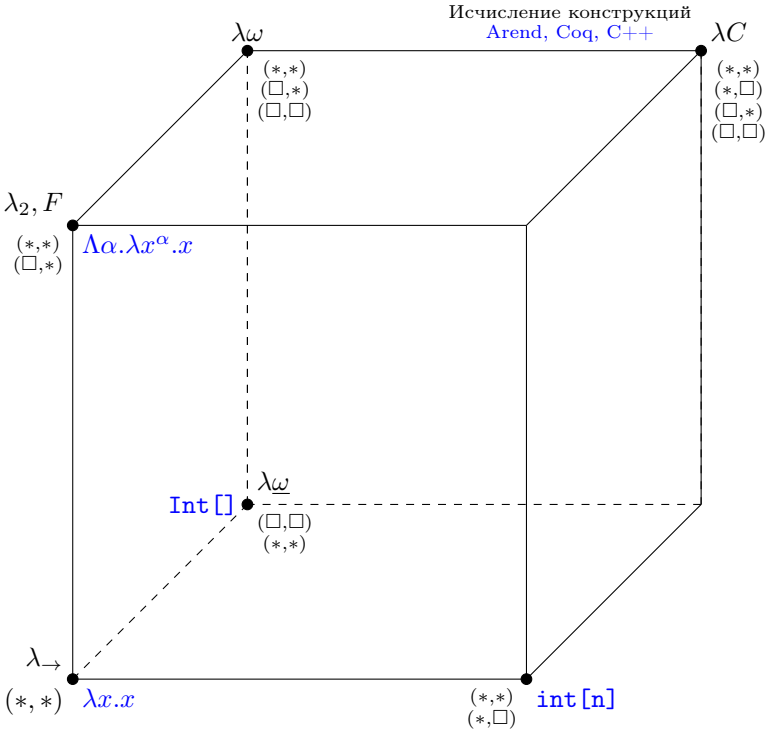
Определение. $S \subseteq C \times C$ — параметризует т.с. $(s_1, s_2) \in S$

- П-правило

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash (\Pi x : A.B) : s_2}$$

- λ-правило

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash b : B \quad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash (\lambda x : A.b) : (\Pi x : A.B)}$$



45 Краткая характеристика вершин лямбда-куба

- $(*, *)$ — позволяет записывать термы, которые зависят от термов
- $(\square, *)$ — позволяет записывать термы, которые зависят от типов
- $(*, \square)$ — позволяет записывать типы, которые зависят от термов
- $(\square, \square)$ — позволяет записывать типы, которые зависят от типов

46 Σ и Π типы

- $\Pi x : \alpha.P(x)$ — эту запись можно читать как (в каком-то смысле в интуиционистском понимании): "У меня есть метод для конструирования объекта типа $P(x)$, использующий любой предоставленный x типа α". Если же смотреть на эту запись с точки зрения классической логики, то ее можно понимать как бесконечную конъюнкцию $P(x_1)\&P(x_2)\&\ldots$. Данная конъюнкция соответствует декартовому произведению, отсюда и название Π-типа (иногда в англоязычной литературе можно встретить *dependent function type*).
- $\Sigma x : \alpha.P(x)$ — Аналогично предыдущему пункту рассмотрим значение с интуиционистской точки зрения: "У меня есть объект x типа α, но больше ничего про него не знаю кроме того, что он обладает свойством $P(x)$". Это как раз в стиле интуиционизма, что нам приходится знать и объект x и его свойство $P(x)$. Это можно представить как пару, а пара — бинарное произведение. С точки же зрения классической логики, мы можем принимать эту формулу как бесконечную дизъюнкцию $P(x_1) \vee P(x_2) \vee \ldots$, которая соответствует алгебраическим типам данных. (иногда в англоязычной литературе можно встретить *dependent sum*).

47 TODO Зависимые типы

48 Функция `printf`

[org:

```
1 printf("%d", "a") // нельзя
```

`printf` : $(x : \texttt{string}) \to F(x)$, пишем так: $\Pi x : \texttt{string}.F(x)$

]

49 Интенсиональное и экстенсиональное равенства, достоинства и недостатки подходов

Определение.

- **Интенсиональное** равенство основано на сравнении объектов по внутренней структуре
- **Экстенсиональное** равенство предполагает, что объекты неразличимы внешне

[org:

- Интенсиональное равенство

  \+ разрешимо

  − слабо

  Успешно завершит сравнение $0''$ и $0''$

- Экстенсиональное равенство

  − неразрешимо

  \+ сильно

  Необходимо предоставить доказательство для сравнения $0''$ и $0''$

]

50 Равенство как путь в топологическом пространстве

Определение. a, b из типа A, тогда $a \overset{A}{=} b$ (равны в типе A), если сущесвует путь $a \rightsquigarrow b$.

- I — интервальный тип [`left`;`right`]
- Сущесвует непрерывная функция $f : I \to A$
  - f `left` $=_\beta a$
  - f `right` $=_\beta b$

```
\data Path
    | path (f: I -> A) : f left = f right
```

51 TODO Язык Аренд

52 Интервальный тип, магия и `coe`

Определение. Элиминатор для равенства (или для интервального типа I):

```
\func coe (P : I -> \Type) (a : P left) (i : I) : P i \elim i
    | left => a
```

Это не настоящее определение, внутри происходит магия

[org: Что здесь написано: матчимся только по `left`. Все объекты вдоль пути равны, значит наверное можем всегда выдавать левую часть. Если `a : P left` (a выполнено в $P\ left$) и `a = a'`, тогда `a' : P right`.
]

53 Стандартные функции: `transport`, `pmap`

```
\func transport {A : \Type}  (B : A -> \Type) {a a' : A} (p : a=a') (b : B a) : B a'
    => coe (\lam i => B (p @ i)) b right
```

[org: *Что здесь написано*:

- Есть тип в котором мы живем A
- Нечто что нам выдаст утверждение B
- `p: a=a'`
  - `p @ left` — `a`
  - `p @ right` — `a'`
- Лямбда идет в первый аргумент `coe`, получаем путь из `B a` в `B a'`
- У нас есть `B a`, попросим конец пути `B a'`

]

```
\func inv (A : \Type) {a a' : A} (p : a = a') : a' = a
    => transport (\lam x => x=a) p (idp {A} {a})
```

```
\func pmap (A B : \Type) (f : A -> B) (a a' : A) {p : a = a'} : f a = f a'
    => transport (\lam x => f a = f x) p idp
```

54 TODO Функциональная экстенсиональность, её доказуемость в Аренде

Определение. Функциональная экстенсиональность — функции, значения которых равны для всех равных значений их аргументов, равны между собой

55 TODO Σ и Π типы в языке Аренд

56 Индуктивные типы, задание отношения «меньше» через индуктивные типы и через Σ-тип

Определение.

$$a \leq b \Leftrightarrow \exists x. a + x = b$$

Зависимая пара:

- Значение $(x, a + x = b)$
- Тип `\Sigma (x : Nat) (a + x = b)`

Определение.

```
\data less-or-equal (a b : Nat)  \with
  | zero, _ => base
  | suc a', suc b' => next (p : less-or-equal a' b')
```

57 TODO Неравенство

57.1 Неравенство

$0 \neq 1$

```
\data Nat
    | zero
    | suc (k : Nat)
```

Докажем что `zero` $\neq$ `suc zero`

- `Not (zero = suc zero)`
- `Not (A : \Type) : A -> Empty`

```
\func proof_ne (a : Nat) : \Type \elim a
    | zero => 0 = 0
    | succ x => Empty
\func zne (x : 0 = 1) : Empty => transport proof_ne {0} {1} x idp
```

58 TODO Доказательство неравенств в Аренде

59 Rewrite

Определение.

```
\func plus-assoc {a b c : Nat}  : (a + b) + c = a + (b + c) \elim c
  | 0 => idp
  | suc c =>
```

```
    -- Можем так:
    -- pmap suc plus-assoc
    -- Но можем попробовать сделать замену
    rewrite plus-assoc idp
    -- На самом деле внутри происходит transport, rewrite угадавает эту лямбду
    -- transport (\lam x => (a + b) + suc c = suc x) plus-assoc idp
```

60 Каков тип типа: необходимость увеличения выразительной силы языка

[org: Мы находимся где-то в λC

```
\func id (x : \Type) : \Type = x
```

К какой части λ-куба относится `id: * -> *`

```
\func idid => id id --- нельзя
```

Но такое мы можем написать:

```
\func id2 (x : \Type) : (x -> x) => \lam a => a
\func idid2 => id2 (\Type -> \Type) (id2 \Type)
```

Получается что `\Type -> \Type : \Type`
]

61 TODO Типы, универсумы, пропы, множества

Определение. `X : \Type` — `\Prop`, если все элементы x равны

Определение. `\Set` — тип, в котором равентсво — `\Prop`

[org: Гомотопический уровень типа — +1 от уровня равенств на нем
]

62 Импредикативность

Определение. Импредикативность — нет различий по пропозициональным уровням

[org: Все `\Prop` импредикативны
]

63 TODO Иерархия универсумов, предикативный и гомотопический уровни

Определение.

- `\Type n`
- `\Type 0` — базовые типы
- `\Type 1` — все, включая `\Type 0`
- `\Type (k + 1)` — все, включая `\Type k`

64 Пропозициональное обрезание

Определение. Пропозициональное обрезание $\|x\|$ `: \Prop`

- `|| Either a b || => a || b`
- `|| \Sigma (x : N) (T(x)) ||` — $\exists x^N\ T(x)$

В аренде это `TruncP : \Type -> \Prop`, `inP`

65 TODO Фактор-множества в Аренде

[org:] Можем писать, говоря что все элементы дататайпа равны между собой

```
\truncated \data Quotient
  (A : \Type) (R : A -> A -> \Type) : \Set
  | inR A
  | eq (a a' : A) (r : R a a') (i : I)
    \elim i {
      | left => inR a
      | right => inR a'
    }
```

Положили `A` в коробочку и рядом положили равенство

]

66 TODO Конструкция `\using \level`

67 TODO Аксиома выбора и эквивалентные утверждения

Определение. Аксиома выбора. S — семейство непустых множеств, то есть $f : S \to \bigcup S$, что $f(x) \in x$. В частности, есть $f : \{u, v\} \to B$, что $f(v) \in u$ и $f(v) \in V$

68 Конструктивная аксиома выбора и её доказуемость

Определение.

- `A B : \Set`, `A` — индексы (S), `B` — множества ($\bigcup S$)
- `Q : A -> B -> \Prop` — отношение быть подмножеством: `Q a b` — b принадлжеит a

```
\func Choice (A B : \Set)
             (Q : A -> B -> \Prop)
             (not_empty : \Pi (x : A) -> \Sigma (y : B) (Q x y)) :
  \Sigma (f : \Pi (x : A) -> B) (\Pi (x : a) -> Q x (f x)) =>
    (\lam (x : A) => not_empty x.1, \lam (x : A) => not_empty x.2)
```

69 Сетоиды

Определение. Сетоид — $S/_{\approx}$, $\approx$ — отношение эквивалентности

```
<S, E, E-trans, E-refl, E-sym>
E : S -> S -> \Prop
```

Здесь E — соответсвующее отношение равенства

70 TODO Аксиома выбора как перестановка кванторов и пропозиционального обрезания

71 Теорема Диаконеску

Теорема 71.1. ИИП + ZF + Аксиома выбора $\implies$ Исключенное третье
Рассмотрим P. $B = \{0, 1\}$

$$U = \{x \in B | x = 0 \vee P\}$$

$$V = \{x \in B | x = 1 \vee P\}$$

Можем заметить что и U и V непустые

72 TODO Парадокс Бурали-Форте

см. лек. 12

73 TODO Парадоксальные универсумы, идея доказательства парадокса Бурали-Форте при существовании парадоксального универсума

Определение. Парадоксальный универсум

- $\sigma : U \to \mathcal{P}U$
- $\tau : \mathcal{P}U \to U$

Если для всех $X \in \mathcal{P}U$

$$\sigma\tau X = \{\tau\sigma x | x \in X\}$$

74 TODO Общая идея построения парадокса Жирара в системе U

[org: В системе U можем написать что-то вроде Y-комбинатора — $F \equiv \lambda$-выражение, которое не заканчивается, но $\vdash F : \varphi$, φ - любой. Значит любой тип обитаем
]

75 Линейная логика

Определение. Структурные правила:

- Обмен

$$\frac{\Gamma, \Delta \vdash A}{\Delta, \Gamma \vdash A}$$

- Сжатие

$$\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}$$

- Ослабление

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}$$

75.1 Линейная логика

Определение. Линейная логика Связки:

$$\alpha \coloneqq x \mid \alpha \multimap \beta \mid \alpha \otimes \beta \mid \alpha \oplus \beta \mid !\alpha$$

Контексты:

$$\underbrace{\langle A \rangle \quad \vdash \ldots}_{\text{линейные}} \qquad \underbrace{[B] \quad \vdash \ldots}_{\text{обычные}}$$

Определение.

$$\frac{\Gamma, \langle A \rangle \vdash B}{\Gamma \vdash A \multimap B} \qquad \frac{\Gamma \vdash A \multimap B \quad \Delta \vdash A}{\Gamma, \Delta \vdash B}$$

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \qquad \frac{\Gamma \vdash A \otimes B \quad \Delta, \langle A \rangle, \langle B \rangle \vdash C}{\Gamma, \Delta \vdash C}$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \& B} \qquad \frac{\Gamma \vdash A \& B}{\Gamma \vdash A} \qquad \frac{\Gamma \vdash A \& B}{\Gamma \vdash B}$$

$$\frac{\Gamma, [A], [A] \vdash B}{\Gamma, [A] \vdash B} \qquad \frac{}{\langle A \rangle \vdash A} \qquad \frac{}{[A] \vdash A}$$

$$\frac{[\Gamma] \vdash A}{[\Gamma] \vdash !A} \qquad \frac{\Gamma \vdash !A \quad \Delta, [A] \vdash B}{\Gamma, \Delta \vdash B}$$

Соответствие с ИИВ

$$\begin{array}{lll} A \to B & !A \multimap B & \\ A \times B & A \& B & !A \otimes !B \\ A + B & !A \oplus !B & \end{array}$$

76 TODO Уникальные типы

Определение. λ-исчисление

$$e \coloneqq x^{\odot} \mid x^{\otimes} \mid \lambda x.e \mid e\ e \mid \tau_k \mid C_k \mid \tau_{(k' \to k)}\ \tau_{k'}$$

Пример.

$$\begin{gathered} dup\ ::\ (t^{\times} \to (t^{\times}, t^{\times}))^{u} \\ dup\ x = (x^{\otimes}, x^{\otimes}) \end{gathered}$$

Определение.

- Введеление перменных

$$\frac{}{\Gamma, x : t^{u} \vdash x^{\odot} : t^{u}|_{x:u}}\mathrm{Var}^{\odot}$$

$$\frac{}{\Gamma, x : t^{\times} \vdash x^{\otimes} : t^{\times}|_{x:\times}}\mathrm{Var}^{\otimes}$$

- Абстракция

$$\frac{\Gamma, x : a \vdash e : b|\mathrm{fv} \quad \mathrm{fv}' = ?_{\times}\mathrm{fv}}{\Gamma \vdash \lambda x.e : a \xrightarrow{\vee \mathrm{fv}'} b|_{\mathrm{fv}'}}$$

- Аппликация

$$\frac{\Gamma \vdash e_1 : a \xrightarrow{u} b|_{\mathrm{fv}_1} \quad \Gamma \vdash e_2 : a|_{\mathrm{fv}_2}}{\Gamma \vdash e_1\ e_2 : b|_{\mathrm{fv}_1 \cup \mathrm{fv}_2}}$$

[org: Универсумы:

- $\mathcal{T}$ — базовых типов (`int`, `bool` и т.п.)
- U — уникальность
  - $\cdot, \times$ — уникальный, не уникальный
  - $\wedge, \vee, \neg$ на U
- $Attr : \mathcal{T} -> U -> *$

]

77 Комбинаторный базис $BCKW$

[org: Вспомним комбинаторы:

- $B = \lambda x.\lambda y.\lambda z.x\ z\ y$
- $C = \lambda x.\lambda y.\lambda z.x\ (y\ z)$
- $K = \lambda x.\lambda y.x$
- $W = \lambda x.\lambda y.x\ y\ y$

Ослабление:

$$\begin{array}{ll} \alpha & \\ \alpha \to \beta \to \alpha & \text{(Схема 1)} \\ \beta \to \alpha & \text{(M.P.)} \end{array}$$

Заметим что:

- Обмен — B
- Сжатие — W
- Ослабление — K

]

78 Полиморфизм (параметрический и наследственный)

[org: Полиморфизм:

- Параметрический
  - Параметричекий полифморфизм как X-M, F
    $\forall x.x \to x$
  - Перегрузка Ad-Hoc

```
print("a")
print(7)
```

- Наследственный
  - ООП
  - Приведение

]

79 Отношение подтипизации

Определение. $F \prec: G$ (F подтип G):

- $f : F$, $g : G$
- f годится везде, где годится g

80 Ко- и контравариантность.

Определение. Вариантность

- Ковариантность: $a \prec_F b$, $f(a) \prec_G f(b)$
- Контрвариантность: $a \prec_F b$, $g(a) \succ_H g(b)$

81 Система $F_{\prec:}$ Ядерное и полное правила

Определение. $F_{\prec:}$

1. $\frac{}{S \prec: S}$ — рефлексивность
2. $\frac{}{S \prec: \text{Top}}$, где Top — константа(тип)
3. $\frac{S \prec: U \quad U \prec: T}{S \prec: T}$ — тразитивность
4. $\frac{T_1 \prec: S_1 \quad S_2 \prec: T_2}{S_1 \to S_2 \prec: T_1 \to T_2}$
5. $\frac{\Gamma \vdash t : S \quad S \prec: T}{\Gamma \vdash t : T}$

[org: Хотим делать ограничения на кванторы $\forall x.x \to x$

$$\frac{}{\Gamma \vdash \text{Top} : *}$$

$$\frac{X \prec: T, \Gamma \vdash T : K}{X \prec: T, \Gamma \vdash X : K}$$

$$\frac{\Gamma, X \prec: T_1 \vdash T_2 : *}{\Gamma \vdash \forall x \prec: T_1.T_2 : *}$$

$$\frac{\Gamma \vdash S : *}{\Gamma \vdash S \prec: \text{Top}}$$

] **[org:** Есть 2 варианта исчисления:

- Ядерное (проще работать)

$$\frac{\Gamma \vdash U_1 : K_1 \quad \Gamma, X \prec: U_1 \vdash S_2 \prec: T_2}{\Gamma \vdash \forall x \prec: U_1.S_2 \prec: \forall x \prec: U_1.T_2}$$

- Полное

$$\frac{\Gamma \vdash U_2 \prec: U_1 \quad \Gamma, X \prec: U_1 \vdash S_2 \prec: T_2}{\Gamma \vdash \forall x \prec: U_1.S_2 \prec: \forall x \prec: U_2.T_2}$$

]