Лекции по Теории Типов 5 семестр

Ilya Yaroshevskiy

13 мая 2023 г.

Оглавление

Лекция 0

1.1 λ-исчисление

Определение.
- $\lambda x.A$ — абстракция
- (AB) — применение (аппликация)
- x — переменная (атом)

Примечание. λ съедает все.
Пример.

$$\lambda x.(\lambda y.(P))$$

Определение.

1. $T \coloneqq \lambda x.\lambda y.x$ — истинна
2. $F \coloneqq \lambda x.\lambda y.y$ — ложь
3. $Not \coloneqq \lambda p.p\ F\ P$
   Пример.

$$\begin{aligned} Not\ F &\to_\beta \\ ((\lambda x.\lambda y.y)\ F)\ P &\to_\beta \\ (\lambda y.y)\ T &\to_\beta \\ T & \end{aligned}$$

4. $And \coloneqq \lambda a.(\lambda b.((a\ b)\ F))$

Определение. Черчевский нумерал — способ кодировать числа:

- $\overline{0} \coloneqq \lambda f.\lambda x.x$
- $\overline{1} \coloneqq \lambda f.\lambda x.f\ x$
- $\overline{3} \coloneqq \lambda f.\lambda x.f\ (f\ (f\ x))$
- $\overline{n+1} \coloneqq \lambda f.\lambda x.f\ (\overline{n}\ f\ x)$
- $(+1) \coloneqq \lambda n.\lambda f.\lambda x.n\ f\ (f\ x)$

- $(+) \coloneqq \lambda a.\lambda b.b\ (+1)\ a$
- $(\cdot) \coloneqq \lambda a.\lambda b.a\ ((+)\ b)\ \overline{0}$

Определение.

$$\langle a, b\rangle \overset{*}{\Rightarrow} \langle b, b+1\rangle$$

$$\langle 0,0\rangle \overset{*}{\Rightarrow} \langle 0,1\rangle \overset{*}{\Rightarrow} \langle 1,2\rangle \overset{*}{\Rightarrow} \langle 2,3\rangle$$

- $Pair \coloneqq \lambda a.\lambda b.\underbrace{(\lambda p.p\ a\ b)}_{\langle a,b\rangle}$

1.1.1 Типы

Определение. $M \vdash A : \tau$

- M — контекст(функции с типами)

Пример. $(+) : int \to int \to int$

Свойство 1. $\to_\beta$ — *сохраняет значения*

Свойство 2.
- $Y \coloneqq \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$ — *Y-комбинатор*
- $\Omega \coloneqq (\lambda x.x\ x)\ (\lambda x.x\ x)$ — *Ω-комбинатор*

Примечание. Решим уравнение $f(A) = A$ — неподвижная точка. Y-комбинатор принимает функцию и возвращает неподвижную точку.

Примечание. Требуется:

- $(\supset)$ — импликация + аксиомы

Докажем, что доказуемо любое утверждение.

$$Y(\lambda b.A \supset b) =_\beta A \supset Y(\lambda b.A \supset b)$$

Пример. Чем интересен Y — он позволяет делать рекурсию. Предположим хотим посчитать факториал.

$$Fact:\ n! = \begin{cases} 1 & ,n = 0 \\ n \cdot (n-1)! & n > 0 \end{cases}$$

$$Fact = \lambda n.If\ (IsZero\ n)\ \overline{1}\ (Fact\ (n-1)) \cdot n$$

- $If\ x\ a\ b =_\beta x\ a\ b$
- $IsZero$ — ДЗ

$$Y\ (\lambda f.\lambda n.If\ (IsZero\ n)\ \overline{1}\ (f\ (n-1)) \cdot n)$$

Доделать

Пример.

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

Y — находит решение даже самых странных уравнений. Запретим (Введем типы).

Определение. Правила добропорядочных выражений:

1.

$$\overline{\Gamma, x : \alpha \vdash x : \alpha}$$

Лекция 1

Определение.

- $\lambda x.A$ — абстракция
- $A\ B$ — применение (аппликация)
- x — переменная (атом)

Примечание.

1. $\lambda a.a$
2. $\lambda b.b$

Хотим чтоб не отличались

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.

- свободные вхождения
- свобода для подстановки
  $A[x := B]$. Никакое свободное вхождение переменных в B, не станет связным после подстановки
- замена свободных вхождений $A[x := B]$

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

Определение. **β-редекс**. Выражение вида $(\lambda x.A)\ B$

Определение. Выражения A и B находятся в отношении **β-редукции**: $A \to_\beta B$, если

1. $A \equiv P\ Q,\ B \equiv R\ S$ и одно из
   - $P \to_\beta R$ и $Q =_\alpha S$
   - $P =_\alpha R$ и $Q \to_\beta S$

2. $A \equiv \lambda x.P,\ B \equiv \lambda x.Q$ и $P \to_\beta Q$
3. $A \equiv (\lambda x.P)\ Q,\ B \equiv P[x := Q]$
   Q свободно для подстановки. Ценой переименования связных переменных можно подставить любое выражение.

Пример. Комбинатор I (Identity)

$$I = \lambda x.x$$

$$(I\ I)\ (I\ I)$$

Доделать

Пример.

- $I = \lambda x.x$
- $K = \lambda x.\lambda y.x$ — Константа
- $\Omega = \omega\ \omega$, где $\omega = \lambda x.x\ x$

$$\begin{aligned} k\ I\ \Omega = \\ ((\lambda x.\lambda y.x)\ I)\ \Omega \to_\beta \\ (\lambda y.I)\ \Omega \to_\beta I \end{aligned}$$

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

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

Существует d: $bRd,\ cRd$

Примечание. Не выполняется на натуральных числах

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

Теорема 2.0.1 (Черча-Россера)**.** β-редуцируемость обладает ромбовидным свойством

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 $(\to_\beta)$

Лемма 1. $(\rightrightarrows_\beta)$ — *обладает ромбовидными свойством*

Лемма 2. *Если R — обладает ромбовидным свойством, то R^* (транзитивное, рефлексивное замыкание) — обладает ромбовидным свойством*

Лемма 3. $(\rightrightarrows_\beta) \subseteq (\twoheadrightarrow_\beta)$

Доказательство Теоремы.

1. $(\rightrightarrows_\beta)^* \subseteq (\twoheadrightarrow_\beta)$ — из леммы
2. $(\twoheadrightarrow_\beta) \subseteq (\rightrightarrows_\beta)^*$
3. Раз $(\rightrightarrows_\beta)^*$ обладает ромбовидным свойством, то и $(\twoheadrightarrow_\beta)$ обладает ромбовидным свойством

□

Следствие 2.0.1.1. У λ-выражения существует не более 1 нормальной формы

Доказательство. Пусть A имеет две нормальные формы:

- $A \twoheadrightarrow_\beta B$
- $A \twoheadrightarrow_\beta C$

и $B \neq_\beta C$. Тогда есть D: $B \twoheadrightarrow_\beta D$ и $C \twoheadrightarrow_\beta D$. Противоречие □

Определение. Нормальный порядок редукции — редуцируем самый левый редекс

Теорема 2.0.2. Если нормальная форма существует, она может быть получена нормальным порядком редукции.

Пример. Перепишем `if (a > b) then X else Y` в λ-выражение:

$$(((a > b)\ X)\ Y)$$

$$\begin{gathered}(((T)\ X)\ Y) \to_\beta \\ (\lambda b.X)\ Y \to_\beta \\ X\end{gathered}$$

Это ленивое вычисление

2.1 Про Y-комбинатор

$$Y\ f =_\beta f\ (Y\ f) =_\beta f\ (f\ (Y\ f)) =_\beta \cdots =_\beta f^{(n)}\ (Y\ f)$$

$$Y\ \overbrace{(\lambda f.\lambda x.x)}^{\varphi} =_\beta \varphi\ (Y\ \varphi)$$

Задача 1. Дз 8.g

Решение.

$$(Y \lambda f.\lambda a.\lambda b.\lambda n.(IsZero\ \ n) a (f\ b\ (a+b)\ (n-1)))\ 1\ 1$$

```
fib a b n =
  if n = 0 then a
  else fib b (a + b) (n - 1)
```

Лекция 2

Определение. Пред λ-терм:

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

Примечание.

1. $\lambda a.a$
2. $\lambda b.b$

Хотим чтоб не отличались

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.

- свободные вхождения
- свобода для подстановки
  $A[x := B]$. Никакое свободное вхождение переменных в B, не станет связным после подстановки
- замена свободных вхождений $A[x := B]$

Пример. Парадокс

1. $\Phi_A \supset \Phi_A$
2. $\Phi_A \supset (\Phi_A \supset A)$
3. $\Phi_A \supset (\Phi_A \supset A) \supset (\Phi_A \supset A)$
4. $\Phi_A \supset A$
5. $(\Phi_A \supset A) \supset \Phi_A$
6. Φ_A

Определение. Типовые переменные:

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

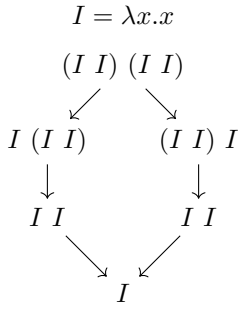
Тип: $\tau ::= (\tau \to \tau) \mid \alpha$
Следует:

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

Определение.

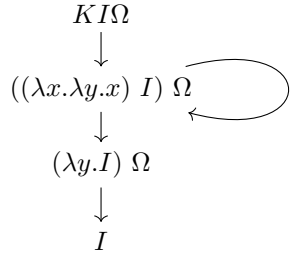
- $\Gamma \vdash A : \tau$ (ΓA^{τ}), где Γ — $\{x_1 : \tau_1,\ x_2 : \tau_2, \dots\}$
- правила:
  1. $$\frac{}{\Gamma, x_1 : \tau_1 \vdash x_1 : \tau_1} \quad x_1 \notin Fv(\Gamma) \quad \text{(Ax.)}$$
  2. $$\frac{\Gamma \vdash A : \sigma \to \tau \quad \Gamma \vdash B : \sigma}{\Gamma \vdash A\ B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

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



Примечание.

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



Пример.

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

Применимо только правило 2. Не имеет типа

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

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

Существует d: $bRd,\ cRd$

Примечание. Не выполняется на натуральных числах

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

Теорема 3.0.1 (Черча-Россера)**.** Если $\Gamma \vdash A : \tau$, то любое подвыражение имеет тип

Определение. Параллельная β**-редукция** $(\rightrightarrows_\beta)$ это $(\rightarrow_\beta)$+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 $(\rightarrow_\beta)$

Лемма 4. $(\rightrightarrows_\beta)$ — *обладает ромбовидными свойством*

Лемма 5. *Если R — обладает ромбовидным свойством, то R^* (транзитивное, рефлексивное замыкание) — обладает ромбовидным свойством*

Лемма 6. $(\rightrightarrows_\beta) \subseteq (\twoheadrightarrow_\beta)$

Доказательство Теоремы. Рассмотрим случай: если выражение типизируется, значит используется одно из правил:

1. $\Gamma \vdash A : \tau$ используется 1 правило $\frac{}{\Gamma, a:\tau \vdash a:\tau}$

Переход:
пусть любое выражение короче n символов обладает свойством. Покажем что этим свойством обладает выражение A длины

Доделать □

Теорема 3.0.2 (Subject reduction)**.** $\Gamma \vdash A : \sigma$ и $A \twoheadrightarrow_\beta B$
Тогда $\Gamma \vdash B : \sigma$

Доказательство. $A \rightarrow_\beta B$ случаи:

1. $\lambda x.A \rightarrow \lambda x.B$
2. $A\ B \rightarrow A'\ B'$
3. $(\lambda x.A)\ B \rightarrow A[x \coloneqq B]$

□

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

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

Теорема 3.0.3. Если конекст Γ и выражение P типизируется, то $\Gamma \vdash_4 P : \sigma$

Пример.

$$\vdash_C \lambda x.x : \alpha \to \alpha(\beta \to \beta)$$

$$\vdash_4 \lambda x.x : \sigma \to \sigma$$

Пример.

$$\lambda f.\lambda x.f\ (f\ x) : \begin{smallmatrix}(\alpha\to\alpha)\to(\alpha\to\alpha)\\(\beta\to\beta)\to(\beta\to\beta)\end{smallmatrix}$$

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

Лекция 3

4.1 Просто типизированное λ-исчисление

4.1.1 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

4.1.2 Исчисления

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

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

Пример.

$$\lambda f.\lambda x.f\ (f\ x) : (\alpha \to \alpha) \to \alpha \to \alpha$$

- $x : \alpha$
- $f : \alpha \to \alpha$

Доказательство:

$$\dfrac{\dfrac{\dfrac{}{f : \alpha \to \alpha \vdash f : \alpha \to \alpha} \quad \dfrac{\dfrac{}{\Gamma \vdash f : \alpha \to \alpha} \quad \dfrac{}{\Gamma \vdash x : \alpha}}{\Gamma \vdash f\ x : \alpha}}{f : \alpha \to \alpha, x : \alpha \vdash f\ f(f\ x) : \alpha}}{f : \alpha \to \alpha \vdash \lambda x.f\ (f\ x) : \alpha \to \alpha}}{\vdash \lambda f.\lambda x.\ f\ (f\ x) : (\alpha \to \alpha) \to \alpha \to \alpha}$$

Пример.

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

Применимо только правило 2. Не имеет типа

Теорема 4.1.1. Если $\Gamma \vdash A : \tau$, то любое подвыражение имеет тип

Доказательство. Рассмотрим случай: если выражение типизируется, значит используется одно из правил:

1. $\Gamma \vdash A : \tau$ используется 1 правило $\frac{}{\Gamma, a:\tau \vdash a:\tau}$

Переход:
пусть любое выражение короче n символов обладает свойством. Покажем что этим свойством обладает выражение A длины

Доделать □

Теорема 4.1.2 (Subject reduction)**.** $\Gamma \vdash A : \sigma$ и $A \twoheadrightarrow_\beta B$
Тогда $\Gamma \vdash B : \sigma$

Доказательство. $A \to_\beta B$ случаи:

1. $\lambda x.A \to \lambda x.B$
2. $A\ B \to A'\ B'$
3. $(\lambda x.A)\ B \to A[x \coloneqq B]$

□

Теорема 4.1.3. Если $\Gamma \vdash M : \sigma$ и:

- существуют N, P :

$$M \twoheadrightarrow_\beta N \quad \Gamma \vdash N : \sigma$$
$$M \twoheadrightarrow_\beta P \quad \Gamma \vdash P : \sigma$$

Тогда найдется такой S, что $\Gamma \vdash S : \sigma$ и $N \twoheadrightarrow_\beta S$ и $P \twoheadrightarrow_\beta S$

4.1.4 Исчисление по черчу

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

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

Теорема 4.1.4. 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.

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

Лекция 4

Примечание. Зачем $\lambda_{\to}$?

1. Логические выражения $\supset$
2. Запрещенные выражения
   - Y — явно нехорошо
   - $\Phi_A =_\beta \Phi_A \supset A$

Добавим очевидные аксиомы

- $A =_\beta B$, то $\vdash A \supset B$ и $\vdash B \supset A$
- $(A \supset A \supset B) \supset (A \supset B)$
- $A, A \supset B$, то B

Тогда при любом A, $\vdash A$

Примечание.

$$\Phi_A \equiv Y\ (\lambda x.x \supset A)$$

5.1 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

- ...

Обозначение. $|\Gamma|$ — оставляем только типы

Теорема 5.1.1 (об изоморфизме Карри-Ховарда)**.**

- $\Gamma \vdash_{\lambda_\to} A : \tau$, то $|\Gamma| \vdash_\to \tau$
- Если $\Delta \vdash_\to \tau$, то найдутся Γ, A: $|\Gamma| = \Delta$, $\Gamma \vdash A : \tau$

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

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

Теорема 5.1.2. 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 $\Gamma \vdash \tau$

Доказательство. Γ^* – множество формул, замкнутых по доказуемости, т.е. $\tau \in \Gamma^*$ т.и т.т.к $\Gamma^* \vdash \tau$ Доделать □

Примечание. $? \vdash ? : ?$. 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

Утверждение. *Выразимы только расширенные полиномы*

Лекция 5

6.1 Алгебраические термы

Определение.

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

Пример.

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

6.1.1 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:

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

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

6.1.2 Эквивалентность уравнений и систем

Определение. Две системы: $E_1 : \begin{cases} T_1 = P_1 \\ \vdots \\ T_\text{n} = P_\text{n} \end{cases}$, $E_2 : \begin{cases} {T_1}' = {P_1}' \\ \vdots \\ {T_\text{n}}' = {P_\text{n}}' \end{cases}$ называются **эквивалентными**, если любое решение системы E_1 подходит к E_2 и наоборот

Утверждение. *Для системы* $\begin{cases} T_1 = P_1 \\ \vdots \\ T_n = p_n \end{cases}$ *существует эквивалентное уравнение*

Доказательство. Выберем k — новый ф.с. n-местный интервал

$$T_1 \ \ldots \ T_n = h \ P_1 \ \ldots \ P_n$$

□

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

Определение. $U \circ T$:

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

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 $S(T) = S(P)$

Теорема 6.1.1. У уравнения в алгебраических термах $T = P$ всегда есть наиболее общее решение, если есть хоть какое-то

Определение. Несовместные система — система с уравнением вида:

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

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

Примечание. В Haskell и OCamll — «occurs check»

Определение. Система в **разрешенной форме** $\begin{cases} a_1 = T_1 \\ \vdots \\ a_n = T_n \end{cases}$, где

1. все a различны
2. T_i не содержит a_j

6.1.3 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.
$$\begin{cases} x = P \\ \vdots \\ T_2 = P_2 \\ \vdots \\ T_n = P_n \end{cases} \Longrightarrow \begin{cases} T_2[x := P] = P_2[x := P] \\ \vdots \\ T_n[x := P] = P_n[x := P] \\ \vdots \\ x = P \end{cases}$$

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

Теорема 6.1.2. Применяя шаги алгоритма унификации, за конечное время можно получить систему либо в разрешенной форме, либо несовместной

6.1.4 Вывод типов в $\lambda_{\to}$

Примечание. $(\to)$ — 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: $E_A \cup E_B \cup \{\tau_B \to \beta = \tau_A\}$
   Тип: β
3. $\lambda x.A$ — рекурсивный вызов $\langle E_A, \tau_A \rangle$
   Система: E_A
   Тип: $\alpha_x \to \tau_A$

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

Пример.
$$\overbrace{\lambda x.\ \underbrace{x}_{A}}^{B}$$

- $E_A = \varnothing$, $\tau_A = \alpha_x$
- $E_B = \varnothing$, $\tau_B = \alpha_x \to \alpha_x$

$$\begin{cases} \tau_A = \alpha \\ \tau_B = \alpha \to \alpha \end{cases}$$

— эта система в разрешенной форме

$$\vdash \lambda x.x : \alpha \to \alpha$$

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

Теорема 6.1.3. $\lambda_{\to}$ сильно нормализуемо

6.2 Исчисление предикатов 2 порядка

Хотим писать $\forall p.p \vee \neg p$

Определение. Логика 2 порядка

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

Утверждение. *Можно выразить:*

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

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

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

Пример.

- $id : \alpha \to \alpha$. $id \equiv \Lambda\alpha.\lambda x^{\alpha}.x$, $id\ (Int)\ 5$

6.3 Практика

6.3.1 ДЗ 3.6

$$A^{\eta \to \eta \to \eta}\ M^{\eta}\ N^{\eta}\ f^{\alpha \to \alpha}\ a : \alpha : \alpha$$

- $\vdash x : \eta, \alpha \to \alpha, \alpha$
- $P\ Q$?
- $(\lambda x^{\tau}.P^{\pi})^{\tau \to \pi}\ A^{\tau} : \pi$, $\pi \equiv \alpha$ Гиганский тип выражения A окажется в переменной x в P

6.3.2 Черчевский нумералы в F

$$\Lambda\alpha.\lambda f^{\alpha \to \alpha}.\lambda x^{\alpha}.f \ldots f(x) : \eta : \forall\alpha.(\alpha \to \alpha) \to \alpha \to \alpha$$
$$(+) = \Lambda\alpha.\lambda a^{\eta}.\lambda b^{\eta}.\lambda f^{\alpha \to \alpha}.\lambda x^{\alpha}.(a\ \alpha)\ f\ ((b\ \alpha)\ f\ x)$$

6.3.3 Правила

Примечание.

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

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

Лекция 6

7.1 Абстрактные типы данных

ООП = АТД + наследование

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

Пример. Стэк:

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

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

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

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

Определение.

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

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

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

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

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, $\tau = [\eta\ \text{List}]$, push = `cons`, pop = λ`(x:xs)` $\to$ `(x, xs)`, empty = `[]`

Пример.

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

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

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

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

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

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

Примечание. $\text{compute} \approx \lambda x^{\delta}.x_2\ (x_1\ \text{true})$

$$\text{case}\ E^{\alpha\vee\beta}\ A^{\alpha\to\beta}\ B^{\beta\to\rho}$$

Передадим типовой параметр в compute:

$$\text{compute} = \Lambda\alpha.\lambda x^{(\to\alpha)\&(\alpha\to)}.x_2\ (x_1\ \text{true})$$

Примечание.

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

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

7.2 Система F

Теорема 7.2.1. Система F сильно нормализуема

Теорема 7.2.2. Система F неразрешима

Лекция 7

8.1 Типовая система Хиндли-Милнера

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 $R(\tau_0 \to \tau_1, k+1)$

Пример.

$$R(\forall\alpha.\alpha, 5)$$
$$R(\alpha, 0) \implies R(\alpha, 5)$$
$$R(\forall\alpha.\alpha, 5)$$

Пример.

$$R(\alpha \to \alpha, 0)$$
$$R(\forall\alpha.\alpha \to \alpha, 1)$$
$$R(\forall\alpha.\alpha \to \alpha, 5)$$

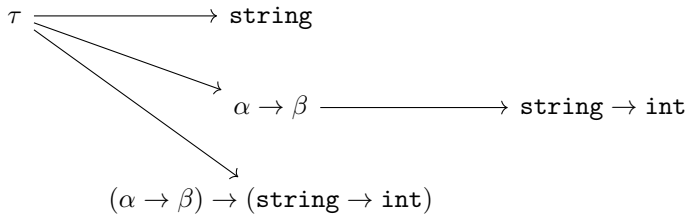
Пример.

$$\neg R((\forall\alpha.\alpha) \to \beta, 1), \text{т.к.} \neg R(\forall\alpha.\alpha, 0)$$
$$R((\forall\alpha.\alpha) \to \beta, 2)$$

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. 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.

Пример.

$$\forall\alpha.\alpha \to \alpha \sqsubseteq \forall\beta.(\beta \to \beta) \to (\beta \to \beta)$$

Определение.

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

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

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

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

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

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

Пример.

$$I \equiv \lambda x.x : \forall\alpha.\alpha \to \alpha$$

$(I\ 1, I\ \text{“}a\text{“}) : (\texttt{int}, \texttt{string})$ — В первом элементе $I : \texttt{int} \to \texttt{int}$, во втором $I : \texttt{string} \to \texttt{string}$

1. $\texttt{let}\ I = \lambda x.x\ \texttt{in}\ (I\ 1, I\ \text{“}a\text{“})$, где $I : \forall\alpha.\alpha \to \alpha$
2. $(\lambda i.(i\ 1, i\ \text{“}a\text{“}))\ (\lambda x.x)$

8.2 Алгоритм W

Утверждение. *Система типов XM* ***разрешима***

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

Пример.

$$\texttt{let}\ I = \lambda x.x\ \texttt{in}\ (I\ 1, I\ \text{“}a\text{“})$$

Применяя 4 пункт алгоритма:

$$I : \forall\alpha.\alpha \to \alpha \vdash (I\ 1, I\ \text{“}a\text{“})$$

8.3 Апгрейд

Добавим правило для Y:

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

Примечание.

```
1 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`:

```
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 : $\alpha \to f(\alpha)$

Лекция 8

9.1 λ-куб

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

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

Пример.

```
1 array[a..b] of int
```

`array[a..b] of` — функция на типах. Ее род: $* \to *$. Применяя ее к `int` получим `int[a..b]`.

Примечание.

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

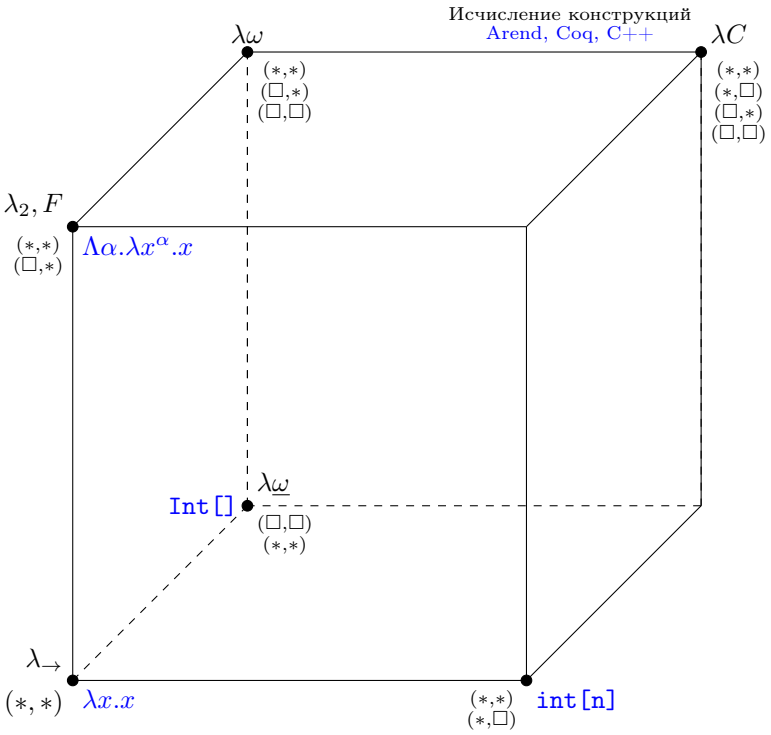
Определение. $S \subseteq C \times C$ — параметризует т.с. $(s_1, s_2) \in S$

- 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

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



Определение. *Сокращение*: $\varphi \to \pi$ заменим на $\Pi x : \varphi.\pi$ если $x \notin FV(\pi)$

Примечание. Пусть $x \notin FV(B)$

$$\frac{\Gamma \vdash A : * \quad \Gamma, x : A \vdash b : B \quad \Gamma \vdash B : *}{\Gamma \vdash (\lambda x : A.b) : A \to B}$$

Примечание.

```
printf("%d", "a") // нельзя
```

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

Пример.

```
template <int n, int m>
class matrix {
        // -- snip --
};
```

$$= \lambda n^{\texttt{int}}.\lambda m \Pi n^{\texttt{int}}.\Pi m^{\texttt{int}}.\exists \alpha.\alpha \& \alpha : \square$$

Пример.

```
data List a = Nil | Const a (List a)
```

- `List`: $* \to * : \square$
- `Nil`: $(a : *) \to \texttt{List}_a \equiv \Pi a^*.\texttt{List}_a : \square$
- `Cons`: $(a : *) \to (h : a : *) \to (t : \texttt{List}_a : *) \to (\texttt{List}_a : *) : \square$

$$\dfrac{\dfrac{}{\vdash * : \square} \quad \dfrac{\dfrac{}{\vdash * : \square} \quad \dfrac{}{\vdash * : \square}}{* : \square \vdash * : \square}}{\vdash * \to * : \square}\ {\scriptstyle \Pi x^*.* : \square}$$

Лекция 9

10.1 Равенство

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
```

10.2 Элиминаторы

Пример. Элиминатор для дизъюнкции: `case` $(f : L \to \Theta)$ $(g : R \to \Theta)$ $(v : L \vee R)$: Θ

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

Примечание. 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`.

10.2.1 transport

```
\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
```

Примечание. 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})
```

Примечание. Константный путь, соединяет a с a

```
\func idp {A : \Type} {a : A} : a = a => path (\lam _ => a)
```

10.2.2 Контруэнтность

```
\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
```

10.2.3 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
```

Лекция 10

11.1 Неравенство

Определение.

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

Зависимая пара:

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

Пример. Хотим доказать: $5 \leq 12$, тогда зависимая пара: `(5, idp) : \Sigma (x : Nat) (5 + x = 12)`

Сделаем GADT для неравенства

Определение.

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

Хотим написать фнукцию, которая будем перестраивать `less-or-equal` в тип с квантором существования (зависимая пара)

```
\func f1 {a b : Nat}  {p1 : less-or-equal a b} : less-or-equal'
  \elim a, b, p1
    | 0, b, base => (b, idp)
    | suc a, suc b, next (pr 1) =>
      \let (pb, ppr) => f1 pr1 \in (pb, pmap suc ppr)
```

где `less-or-equal'` — $\exists x. a + x = b$

Аренд умеет сам делать 1-2 перехода ао определению, например `suc (x + a) = suc b` = `x + suc a = suc b`

```
\func f2 {a b : Nat} (p1 : less-or-equal' a b) : less-or-equal a b
  | {0}, _ => base
```

```
  -- по идее это absurd (transport fs p ()),
  -- но здесь аренд с сотоянии сам найти противоречие
  | {suc a}, {0}, (xm p) => contradiction
  | {suc a}, {suc b}, (x, p) => next (f2 {a} {b} (x, pmap minus1 p))
```

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
```

11.2 Классы

Вспомним определение группы: это $\langle R, +, e : R, ^{-1} : R \to R \rangle$, такие что:

- $e + x = x$
- $x + e = x$
- $x + x^{-1} = e$
- $x^{-1} + x = e$

```
\class Preorder R
 --snip--

\instance OrdNat : Preorder Nat
  -- Это зависимый тип, который нужно доказать
  | <= (a b : Nat) => TruncP (\Sigma (r : Nat) (r + a = b))
  | <=-reflexive => inP ((0, idp))
  | <=-transitive {x} {y} {z} =>
    -- Было искушение написать \lam (t1 : x <= y) ...
    \lam (t1 : TruncP (\Sigma (r : Nat)) (r + x = y))
         (t2 : TruncP (\Sigma (r : Nat)) (r + y = z)) =>
         \case t1, t2 \with {
           inP (d1, p1), inP (d2, p2) => inP (d1 + d2, {?})
         }
```

Примечание. Про `x` можем доказывать что:

- `x : Type`

- `x : Prop`

$\forall a\ b\ :\ x.a = b$. `TruncP` делает свой аргумент `Prop`'ом, то есть теперь все доказательства равны между собой

Лекция 11

12.1 Язык

Примечание. 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`

```
\func Church => (x : \Type)  -> (x -> x) -> (x -> x)
\func add (m n : Church) => m Church inc n
```

Здесь ожидаем, что `inc : Church -> Church`

Примечание. Как это фиксить: `\Type : \Type` невозможно (парадокс Жирара)

12.1.1 Предикативность

Определение.

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

Пример. Черч на типах

```
\func Church => \Pi (x : \Type) -> (x -> x) -> (x -> x)
\func inc (n : Church) => \lam t f x => n t f (f x)
\func add (m : Church \levels (\suc\lp)\lh)
          (n : Church \levels \lp \lh) : Church \levels \lp \lh
          => m Church inc m
```

В add нужно применить `Church` к другим `Church`. `\lp` — внутренняя перменная, которая хранит текущий уровень предикативности. Мы говорим что первый аргумент имеет уровень предикативности на один больше, второй аргуент и результат имеют один уровень предикативности. Тогда `m Church` будет иметь тип `(Church -> Church) -> (C -> C)`, `Church` который внутри имеет уровень предикативности на один меньше

Примечание. `\Prop` — вселенная пропозиций "чистых утверждений"

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

Пример. Доказательство `Nat` — prop

Пример. `a : \Prop`, `b : \Prop`, то `(a, b) : \Prop`

Пример. `Either a b` — не prop, `inLeft a` $\neq$ `inRight b`

Пример. $\exists x.\varphi(x)$ — не prop

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

Примечание. Гомотопический уровень типа — $+1$ от уровня равенств на нем

Определение. Импредикативность — нет различий по пропозициональным уровням

Примечание. Все `\Prop` импредикативны

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`

Можем объявлять равенство между некоторыми вещами

```
\data Int'
  | pos' Nat
  | neg' Nat
```

Проблема: есть два 0. Можно сделать так

```
\data Int
  | pos Nat
  | neg (n : Nat) \elim n {
    | 0 => pos 0
  }
```

Примечание. 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

Примечание.

- `Set` деклалирует единственность равенства
- Prop деклалирует единственность элементов

Лекция 12

13.1 Парадокс Жирара

λ-куб не такой вырахительный как хотелось бы

Примечание. Топология: $\Omega \subseteq \mathcal{P}(X)$

- $\mathcal{P}(X)$ — множество всех функций типа `X -> *`
- $x \in \mathcal{P}(X)$, `x : X -> *`

Топология: `(X -> *) -> *` — про подмножество говорим, подходит ли оно

Примечание.

0:	Все значения (λ-терм)	
1:	Типы (утверждения)	$*$
2:	Рода	$* \to *$
3:	Сорта	$\square$
4:		$\triangle$

13.1.1 Обобщение λ-куба

Примечание. λHOL:

- $*, \square, \triangle$
- Правила: $(*,*), (\square,\square), (\square,*)$.

Если добавим $(\triangle, *)$, все останется хорошо. Там живет: $\Pi x^{\square}$.значения. Если добавим $(\triangle, \square)$, получим неконсистентность.

$$\overbrace{\underbrace{(*,*)\ (\square,\square)\ (\square,*)}_{\textbf{Система } U^-}\ (\triangle,\square)}^{\textbf{Система } U}\ (\triangle,*)$$

И U и U^- неконсистентны.

Примечание. $\mathcal{P}(X)$ — тип всех функций $X \to *$, тогда топология:

$$\underbrace{\underbrace{X \to *}_{\square} \to *}_{\square}$$

Заметим, что это на самом деле квантор всеобщности:

$$\forall \alpha^{\tau}.\varphi(\alpha) = A(\underset{*}{\tau}, \varphi)^{\square}_{\square}$$

$$S \ : \ A(\tau) \approx \overbrace{(\tau \to *)}^{\varphi} \to *$$

φ — обобщенная функция отображающая значения типа в утверждение, квантор всеобщности — штука, которая отображает такую фукнцию в утверждение: истина эта функция или ложна.

Примечание. 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

13.2 Парадок Бурали-Форте

1. **Утверждение.** *Не существует максиамльного ординала (множества всех ординалов)*

   Определение. Ординал — транзитивное, вполне упорядоченное множество

   S — множество всех ординалов, Тогда оно $\in S$?

2. Фундированное множество X — множество, где нет бесконечной цепочки $\in$

   $$\underbrace{X \ni x_1 \ni x_2 \ni \cdots \ni x_n}_{\text{конечное } n}$$

   Множество всех фундированных множеств — фундированное

3. Множество всех множеств

   $$\sigma : X \to \mathcal{P}X$$
   $$\tau : \mathcal{P}X \to X$$

   - σX — $\{a | a \in X\}$ — начальный отрезок до X
   - τX — ординал, соответствующий X

   $$\sigma\tau X = \{\tau\sigma\alpha | \alpha \in X\}$$
   $$\sigma\tau X = \{\beta | \beta < \tau X\} = \{\beta | \beta = \tau\sigma\alpha \text{ для } \alpha \in Z\}$$

Определение. Парадоксальный универсум

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

Если для всех $X \in \mathcal{P}U$

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

Определение. $y \in \sigma x$, то $y < x$

Примечание. $\tau\sigma y < \tau\sigma x$

X — **индуктивен**, если каждый $x : y < x$, то $y \in X$, тогда $x \in X$

Примечание. Трансфинитная индукция — если утверждение истнинно для всех ординалов меньше $x \implies$ истинно для ординалов x, то оно истинно везде

X — фундировано, если x принадлежит всем индуктивным множествам

Определение. $\Omega = \tau\{x | x$ — фундировано$\}$

Утверждение. Ω — *фундировано*

Утверждение. Ω — *не фундировано*

Примечание. Как это выразить в U?

- $\mathcal{P}X$: `X -> *`
- $U : \square$
- $\sigma : U \to \mathcal{P}U$
- $\tau : \mathcal{P}U \to U$
- $o : \forall S^{\mathcal{P}U}.\sigma(\tau(x)) = \lambda u^U.\exists x^U.(Sx)\&u = \tau(\sigma(x))$

Лекция 13

14.1 Теорема Диаконеску

Теорема 14.1.1. 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

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 $f(v) \in V$

Примечание. $f(u) \stackrel{?}{=} f(v)$

да

$$f(u) = 0 \implies f(v) = 0 \implies p$$
$$f(U) = 1 \implies p$$

нет

$$f(u) \neq f(v)$$
$$u \neq v \to \neg p$$

Пусть p истно, тогда $u = v$, но $u \neq v$

14.1.1 Аксиома выбора (попытка 1)

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)
```

Произошла перестановка кванторов

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

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

Здесь E — соответсвующее отношение равенства

Проблема: `not_empty` слишком сильный

14.1.2 `Logic.Classical`

1. Переформулируем аксиому выбора
   - `A : \Set`
   - `B : \Set`
   - $\underbrace{\Pi x : A.|Bx|}_{p} \to |\Pi x A.Bx|$

   Пусть $y : A$ и $p\ y = \text{Empty}$
   - Т.е. Bx не пусто для всех $x : A$
   - Это чистое существование
2. LEM

```
\lem (p : \Prop) : Dec p =>
  \case (f (in~ true)).1 \as x,
        (f (in~ false)).1 \as y \with {
    true, true => {?} -- yes
    false, false => {?} -- yes
    true, false => {?} -- no
    false, true => {?} -- no
  }
```

Лекция 14

15.1 Линейные типы

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

Примечание. 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:

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

Заметим что:

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

Хотим потсроить теорию без сжатия и ослабления:

- B, C — линейная

- B, C, K — афинная

Определение. Правила для конъюнкции

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \times B}$$

$$\frac{\Gamma \vdash A \times B \quad \Delta, A, B \vdash C}{\Gamma, \Delta \vdash C}$$

Пример.

$$\dfrac{\dfrac{\dfrac{}{A \vdash A} \quad \dfrac{\dfrac{}{A \to B \vdash A \to B} \quad \dfrac{}{A \vdash A}}{A \to B, A \vdash B}}{A, A \to B, A \vdash A \times B}}{\dfrac{A \to B, A, A \vdash A \times B}{A \to B, A \vdash A \times B}}$$

Примечание. Классические правила

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \times B}$$

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

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

15.1.1 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.

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

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

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

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

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

Соответствие с ИИВ

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

15.1.2 Уникальные типы

Примечание. Универсумы:

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

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.

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

Определение.

- Введеление перменных

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

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

- Абстракция

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

- Аппликация

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

Лекция 15

16.1 Подтипы

Примечание. Полиморфизм:

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

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

- Наследственный
  - ООП
  - Приведение

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

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

Примечание. $F \prec: G$

- $T = F \to H$
- $S = G \to H$

тогда $S \prec: T$

- $T' = H \to F$
- $S' = H \to G$

тогда $T' \prec: S'$

Определение. Вариантность

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

Пример. Рассмотрим списки $A \prec B$:

- `set` — ковариантен, $B[] \prec A[]$
- `get` — контрвариантен,$A[] \prec B[]$

Списко инвариантен

Определение. $A[x] = x \to H$, $A : * \to *$

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. $\frac{\Gamma \vdash t : S \quad S \prec: T}{\Gamma \vdash t : T}$

Примечание. Хотим делать ограничения на кванторы $\forall x.x \to x$

$$\frac{}{\Gamma \vdash \text{Top} : *}$$
$$\frac{X \prec: T, \Gamma \vdash T : K}{X \prec: T, \Gamma \vdash X : K}$$
$$\frac{\Gamma, X \prec: T_1 \vdash T_2 : *}{\Gamma \vdash \forall x \prec: T_1.T_2 : *}$$
$$\frac{\Gamma \vdash S : *}{\Gamma \vdash S \prec: \text{Top}}$$

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

Примечание. Соберем ООП:

- $(\prec:)$
- $\exists$ (экзистенциальные типы)
- `B = A + {x : S}` $\prec:$ `A = { ... }`

$$\frac{}{\{l_i : T_{i=\overline{1 \dots n+k}}\} \prec: \{l_i : T_{i=\overline{i \dots n}}\}}$$