

# Типы в языках программирования

## Лекция 10. Зависимые типы: система λР

Денис Николаевич Москвин

ИТМО, корпоративная магистратура JetBrains  
Разработка ПО / Software Engineering

12.05.2021

- 1 Зависимые типы
- 2 Система  $\lambda P$ : формализм
- 3 Минимальная логика предикатов в  $\lambda P$

- 1 Зависимые типы
- 2 Система  $\lambda P$ : формализм
- 3 Минимальная логика предикатов в  $\lambda P$

- Вектор — список, в типе которого есть информация о его длине:

```
    [] : Vect 0
  [fls] : Vect 1
 [tru,fls] : Vect 2
[fls,tru,fls] : Vect 3
    ...
```

- Это приводит нас к идее *семейства типов* — типа, индексированного значениями другого типа.
- Конструктор `Vect` — оператор, принимающий значение и возвращающий тип.

- При этом могут возникнуть **бесмысленные конструкции**:

`Vect tru`, `Vect (pair 3 5)`

- Для контроля за ними расширим понятие **кайнда**

`Vect : Nat → *`

- Правило типизации

$$\frac{\Gamma \vdash \varphi : \sigma \rightarrow \kappa \quad \Gamma \vdash N : \sigma}{\Gamma \vdash \varphi N : \kappa}$$

позволяет осуществлять контроль соответствия:

`Vect 3` : \*

`Vect tru` : **ошибка**

- Стандартные конструкторы вектора — их бесконечно много?

$\text{vnil} : \text{Vect } 0$

$\text{vcons0} : \sigma \rightarrow \text{Vect } 0 \rightarrow \text{Vect } 1$

$\text{vcons1} : \sigma \rightarrow \text{Vect } 1 \rightarrow \text{Vect } 2$

$\text{vcons2} : \sigma \rightarrow \text{Vect } 2 \rightarrow \text{Vect } 3$

$[\ ] \equiv \text{vnil}$

$[\text{fls}] \equiv \text{vcons0 fls vnil}$

$[\text{tru}, \text{fls}] \equiv \text{vcons1 tru (vcons0 fls vnil)}$

$[\text{fls}, \text{tru}, \text{fls}] \equiv \text{vcons2 fls (vcons1 tru (vcons0 fls vnil))}$

- Обобщим понятие типа, введя *тип, зависящий от терма*:

$\text{vcons} : \prod n^{\text{Nat}}. \sigma \rightarrow \text{Vect } n \rightarrow \text{Vect } (\text{succ } n)$

- Конструкция

$$\Pi x^\sigma. \tau[x]$$

называется *типом зависимого произведения*.

(Отметим аналогию с  $\forall \alpha^*. \tau[\alpha]$  из системы  $\lambda 2$ .)

- $\Pi x^\sigma. \tau$  — тип функции, отображающей терм  $N$  типа  $\sigma$  в терм типа  $[x \mapsto N]\tau$ .
- $\Pi$ -тип обобщает обычный инфиксный конструктор типа функции  $\rightarrow : * \rightarrow * \rightarrow *$ .
- Мы можем объявить второй синтаксическим сахаром для первого:

$$\sigma \rightarrow \tau \equiv \Pi \_{}^\sigma. \tau$$

где подчеркивание означает, что мы не интересуемся значением аргумента при конструировании типа и, следовательно, не будем его использовать в  $\tau$ .

Правило введения  $\Pi$ -типа:

$$\frac{\Gamma, x:\sigma \vdash M:\tau \quad \Gamma \vdash \sigma : *}{\Gamma \vdash \lambda x^\sigma. M : \Pi x^\sigma. \tau}$$

Например,

```
twice  :   $\Pi n^{\text{Nat}}. \sigma \rightarrow \text{Vect } n \rightarrow \text{Vect } (\text{succ } (\text{succ } n))$   
twice   $\equiv \lambda n^{\text{Nat}} e^\sigma v^{\text{Vect } n}. v\text{cons } (\text{succ } n) e (v\text{cons } n e v)$ 
```

```
[fls, tru, fls] : Vect 3
```

```
twice 3 tru [fls, tru, fls] : Vect 5
```

```
twice 3 tru [fls, tru, fls]  $\rightarrow_\beta$  [tru, tru, fls, tru, fls]
```

В любой разумной прикладной реализации параметр  $n$  передается и контролируется неявно. Достаточно присутствия в типе.



# Полное определение вектора в $\lambda P2$

`Vect` :  $\text{Nat} \rightarrow *$

`Vect`  $\equiv$   $\lambda m^{\text{Nat}}. \Pi \varphi^{\text{Nat} \rightarrow *}. \varphi 0 \rightarrow (\Pi n^{\text{Nat}}. \sigma \rightarrow \varphi n \rightarrow \varphi (\text{succ } n)) \rightarrow \varphi m$

`List`  $\equiv$   $\Pi \alpha^*. \alpha \rightarrow (\sigma \rightarrow \alpha \rightarrow \alpha) \rightarrow \alpha$

`vnil` : `Vect` 0

`vnil`  $\equiv$   $\lambda \varphi^{\text{Nat} \rightarrow *}. z^{\varphi 0} c^{\Pi n^{\text{Nat}}. \sigma \rightarrow \varphi n \rightarrow \varphi (\text{succ } n)}. z$

`nil`  $\equiv$   $\lambda \alpha^*. z^{\alpha} c^{\sigma \rightarrow \alpha \rightarrow \alpha}. z$

`vcons` :  $\Pi n^{\text{Nat}}. \sigma \rightarrow \text{Vect } n \rightarrow \text{Vect } (\text{succ } n)$

`vcons`  $\equiv$   $\lambda n^{\text{Nat}}. e^{\sigma} v^{\text{Vect } n} \varphi^{\text{Nat} \rightarrow *} z^{\varphi 0} c^{\Pi n^{\text{Nat}}. \sigma \rightarrow \varphi n \rightarrow \varphi (\text{succ } n)}. c n e (v \varphi z c)$

`cons`  $\equiv$   $\lambda e^{\sigma} v^{\text{List}} \alpha^* z^{\alpha} c^{\sigma \rightarrow \alpha \rightarrow \alpha}. c e (v \alpha z c)$

$Vec$  :  $* \rightarrow Nat \rightarrow *$

$Vec \equiv \lambda \sigma^* m^{Nat}. \Pi \varphi^{Nat \rightarrow *}. \varphi 0 \rightarrow (\Pi n^{Nat}. \sigma \rightarrow \varphi n \rightarrow \varphi (succ\ n)) \rightarrow \varphi m$

$vnil$  :  $Vec\ \sigma\ 0$

$vcons$  :  $\Pi n^{Nat}. \sigma \rightarrow Vec\ \sigma\ n \rightarrow Vec\ \sigma\ (succ\ n)$

Примеры функций:

$replicate$  :  $\Pi n^{Nat}. \sigma \rightarrow Vec\ \sigma\ n$

$head$  :  $\Pi n^{Nat}. Vec\ \sigma\ (succ\ n) \rightarrow \sigma$

$concat$  :  $\Pi m^{Nat}. \Pi n^{Nat}. Vec\ \sigma\ m \rightarrow Vec\ \sigma\ n \rightarrow Vec\ \sigma\ (m + n)$

$zip$  :  $\Pi n^{Nat}. Vec\ \sigma\ n \rightarrow Vec\ \tau\ n \rightarrow Vec\ \langle \sigma, \tau \rangle\ n$

$zip'$  :  $\Pi m\ n^{Nat}. Vec\ \sigma\ m \rightarrow Vec\ \tau\ n \rightarrow Vec\ \langle \sigma, \tau \rangle\ \min(m, n)$

Можно ввести *тип зависимой суммы*

$$\Sigma x^\sigma. \tau[x]$$

Это тип пары из терма  $N:\sigma$  и терма  $M:[x \mapsto N]\tau$ .

Правило введения  $\Sigma$ -типа:

$$\frac{\Gamma \vdash N:\sigma \quad \Gamma, x:\sigma \vdash M:[x \mapsto N]\tau \quad \Gamma \vdash \sigma : *}{\Gamma \vdash \langle N, M \rangle : \Sigma x^\sigma. \tau}$$

Если  $\tau$  не содержит  $x$ , то это обычный тип (независимой) пары

$$\sigma \times \tau \equiv \Sigma \_^\sigma. \tau$$

Пример

$$\langle 2, [\text{tru}, \text{fls}] \rangle : \Sigma k^{\text{Nat}}. \text{Vec Bool } k$$

- 1 Зависимые типы
- 2 Система  $\lambda P$ : формализм
- 3 Минимальная логика предикатов в  $\lambda P$

- Мы не различаем заранее типы и термы.
- Множество *псевдовыражений* определяется индуктивно:

$$\mathcal{T} = V \mid C \mid \mathcal{T}\mathcal{T} \mid \lambda V^{\mathcal{T}}. \mathcal{T} \mid \Pi V^{\mathcal{T}}. \mathcal{T},$$

где  $V$  — бесконечный набор переменных, а  $C = \{*, \square\}$ .

- Константы  $*$  и  $\square$  называют *сортами*.
- В дальнейшем переменная  $s$  пробегает множество  $\{*, \square\}$ .

- *Высказывание* в  $\lambda P$  имеет вид  $M : A$  (или  $M^A$ ), где  $M, A \in \mathcal{T}$ .
- *Контекст* — это конечное, линейно упорядоченное множество высказываний с различными переменными в качестве субъекта.
- $\langle \rangle$  обозначает пустой контекст.
- Контексты можно расширять:  
если  $\Gamma = \langle x_1^{A_1}, \dots, x_n^{A_n} \rangle$ , то  $\Gamma, y^B = \langle x_1^{A_1}, \dots, x_n^{A_n}, y^B \rangle$ .
- Все это полностью идентично  $\lambda\omega$ .

# Правила типизации для $\lambda P$ (1)

Утверждение  $\Gamma \vdash_{\lambda P} M : A$  задается правилами:

(аксиома)

$$\langle \rangle \vdash * : \square$$

(начальное правило)

$$\frac{\Gamma \vdash A : s}{\Gamma, x^A \vdash x : A}, x \notin \Gamma$$

(правило ослабления)

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

(формирование типа/кайнда)

$$\frac{\Gamma \vdash A : * \quad \Gamma, x^A \vdash B : s}{\Gamma \vdash \Pi x^A. B : s}$$

продолжение на следующем слайде...

...начало на предыдущем слайде

(правило применения) 
$$\frac{\Gamma \vdash M : \Pi x^A. B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : [x \mapsto N]B}$$

(правило абстракции) 
$$\frac{\Gamma, x^A \vdash M : B \quad \Gamma \vdash \Pi x^A. B : s}{\Gamma \vdash \lambda x^A. M : \Pi x^A. B}$$

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



$$\frac{\Gamma \vdash A : * \quad \Gamma, x^A \vdash B : s}{\Gamma \vdash \Pi x^A. B : s} \quad (\text{формирование типа/кайнда})$$

Положим в правиле  $s = *$ . Тогда оно примет вид

$$\frac{\Gamma \vdash \sigma : * \quad \Gamma, x^\sigma \vdash \tau : *}{\Gamma \vdash \Pi x^\sigma. \tau : *}$$

Если тип  $\tau$  не зависит от переменной  $x$ , то стрелочную нотацию можно восстановить как удобный синтаксический сахар:

$$\sigma \rightarrow \tau \equiv \Pi \_^\sigma. \tau \equiv \Pi x^\sigma. \tau$$

Например,

$$\alpha^*, \beta^* \vdash \Pi x^\alpha. \beta : * \quad (\text{или } \alpha \rightarrow \beta : *)$$

Ниже мы увидим примеры с зависимыми типами.

$$\frac{\Gamma \vdash A : * \quad \Gamma, x^A \vdash B : s}{\Gamma \vdash \Pi x^A. B : s} \quad (\text{формирование типа/кайнда})$$

Положим в правиле  $s = \square$ . Тогда оно примет вид

$$\frac{\Gamma \vdash \sigma : * \quad \Gamma, x^\sigma \vdash k : \square}{\Gamma \vdash \Pi x^\sigma. k : \square}$$

Если кайнд  $k$  не зависит от переменной  $x$ , то пользуются стрелочной нотацией

$$\sigma \rightarrow k \equiv \Pi \_^\sigma. k \equiv \Pi x^\sigma. k$$

Например,

$$\alpha^* \vdash \alpha \rightarrow * : \square$$

То есть  $\alpha \rightarrow *$  — допустимый кайнд семейства типов.

$$\frac{\Gamma \vdash M : \Pi x^A. B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : [x \mapsto N]B} \quad (\text{правило применения})$$

При  $\Pi x^A. B : *$  имеем удаление  $\Pi$ -типа:

$$\frac{\Gamma \vdash M : \Pi x^\sigma. \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : [x \mapsto N]\tau}$$

$$\frac{\Gamma \vdash M : \Pi x^A. B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : [x \mapsto N]B} \quad (\text{правило применения})$$

При  $\Pi x^A. B : \square$ , имеем удаление  $\Pi$ -квантора:

$$\frac{\Gamma \vdash \varphi : \Pi x^\sigma. \kappa \quad \Gamma \vdash N : \sigma}{\Gamma \vdash \varphi N : [x \mapsto N]\kappa}$$

$$\alpha^*, \varphi^{\alpha \rightarrow *}, a^\alpha \vdash \varphi a : *$$

$$\alpha^*, \varphi^{\alpha \rightarrow *}, a^\alpha \vdash \varphi a \rightarrow * : \square$$

$$\alpha^*, \varphi^{\alpha \rightarrow *} \vdash \Pi a^\alpha. \varphi a \rightarrow * : \square$$

$$\alpha^*, \varphi^{\alpha \rightarrow *}, \psi^{\Pi a^\alpha. \varphi a \rightarrow *}, x^\alpha \vdash \psi x : \varphi x \rightarrow *$$

$$\alpha^*, \varphi^{\alpha \rightarrow *}, \psi^{\Pi a^\alpha. \varphi a \rightarrow *}, x^\alpha, y^{\varphi x} \vdash \psi x y : *$$

Функция  $\psi$  выступает как хранилище для зависимой пары.

# Правило абстракции: терм

$$\frac{\Gamma, x^A \vdash M : B \quad \Gamma \vdash \Pi x^A. B : s}{\Gamma \vdash \lambda x^A. M : \Pi x^A. B} \quad (\text{правило абстракции})$$

При  $s = *$  (и, следовательно,  $B : *$ ) возникает правило введения  $\Pi$ -типа:

$$\frac{\Gamma, x^\sigma \vdash M : \tau \quad \Gamma \vdash \sigma : *}{\Gamma \vdash \lambda x^\sigma. M : \Pi x^\sigma. \tau}$$

Например,

$$\begin{array}{l} \alpha^*, \varphi^{\alpha \rightarrow *}, a^\alpha, x^{\varphi a} \vdash x : \varphi a \\ \alpha^*, \varphi^{\alpha \rightarrow *}, a^\alpha \vdash \lambda x^{\varphi a}. x : \varphi a \rightarrow \varphi a \\ \alpha^*, \varphi^{\alpha \rightarrow *} \vdash \lambda a^\alpha. \lambda x^{\varphi a}. x : \Pi a^\alpha. \varphi a \rightarrow \varphi a \end{array}$$

# Правило абстракции: тип

$$\frac{\Gamma, x^A \vdash M : B \quad \Gamma \vdash \Pi x^A. B : s}{\Gamma \vdash \lambda x^A. M : \Pi x^A. B} \quad (\text{правило абстракции})$$

При  $s = \square$  (и, следовательно,  $B : \square$ ) возникает правило введения  $\Pi$ -кайнда

$$\frac{\Gamma, x^\sigma \vdash \tau : \kappa \quad \Gamma \vdash \sigma : *}{\Gamma \vdash \lambda x^\sigma. \tau : \Pi x^\sigma. \kappa}$$

Примеры:

$$\alpha^*, \psi^{\alpha \rightarrow \alpha \rightarrow *}, a^\alpha \vdash \psi a : \alpha \rightarrow *$$

$$\alpha^*, \psi^{\alpha \rightarrow \alpha \rightarrow *}, a^\alpha \vdash \psi a a : *$$

$$\alpha^*, \psi^{\alpha \rightarrow \alpha \rightarrow *} \vdash \lambda a^\alpha. \psi a a : \alpha \rightarrow *$$

$$\frac{\Gamma \vdash A : B \quad \Gamma \vdash B' : s \quad B =_{\beta} B'}{\Gamma \vdash A : B'} \quad (\text{правило преобразования})$$

Алгоритмически неудобно, в реализации будет встроено в правило применения

$$\frac{\Gamma \vdash M : \Pi x^A. B \quad \Gamma \vdash N : A' \quad A =_{\beta} A'}{\Gamma \vdash M N : [x \mapsto N]B}$$

Вопрос в том, насколько типизация должна форсировать вычисления. **Какой тип вернуть**

$$\begin{array}{l} \varphi^{\text{Nat} \rightarrow *}, y^{\varphi(4+1)} \vdash (\lambda x^{\varphi(2+3)}. x) y : ? \\ m^{\text{Nat}}, n^{\text{Nat}}, \varphi^{\text{Nat} \rightarrow *}, y^{\varphi(m+n)} \vdash (\lambda x^{\varphi(n+m)}. x) y : ? \end{array}$$

- 1 Зависимые типы
- 2 Система  $\lambda P$ : формализм
- 3 Минимальная логика предикатов в  $\lambda P$



- Три основных компоненты логики предикатов: множества, предикаты над множествами и высказывания.
- Множества кодируются как типы:

$$S : *$$

Например,  $\text{Nat} : *$  — множество натуральных чисел, элементы которого — числа (Черча).

- Высказывания тоже кодируются как типы:

$$A : *$$

Например,  $a \rightarrow a : *$  — тавтология  $a \Rightarrow a$  с доказательством  $\lambda x^a. x$ .

- Предикаты кодируются как функции из множества  $S$  в множество высказываний:

$$P : S \rightarrow *$$

- В  $\lambda P$  можно вложить минимальную логику предикатов первого порядка.
- Домены и формулы логики представляются типами, сигнатура (функциональные и предикатные символы) задается в контексте:

$$\Gamma \equiv A^*, a^A, f^{A \rightarrow A}, p^{A \rightarrow *}, q^{A \rightarrow *}, r^{A \rightarrow A \rightarrow *}$$

- **Связки:** импликация представляется как  $\rightarrow$ , а квантор  $\forall$  как  $\Pi$ :

$$\begin{aligned}\forall x \in A. P x &\sim \Pi x^A. P x \\ \forall x \in A. R x x \Rightarrow P x &\sim \Pi x^A. R x x \rightarrow P x\end{aligned}$$

- **Правила** введения и удаления — это лямбда-абстракция и аппликация.



$$A^* \vdash A \rightarrow * : \square$$

Если  $A$  это тип, рассматриваемый как множество, то  $A \rightarrow *$  представляет собой кайнд предикатов над  $A$ .



$$A^*, P^{A \rightarrow *}, a^A \vdash P a : *$$

Если  $a \in A$ , и  $P$  — предикат над  $A$ , то  $P a$  — это тип, рассматриваемый как высказывание (истинное, если тип населен, и ложное в противном случае).

- Различие между множествами и предикатами можно ввести явно:

$$A^{*^s}, P^{A \rightarrow *^p}, a^A \vdash P a : *^p$$



$$A^*, R^{A \rightarrow A \rightarrow *} \vdash \Pi a^A. R a a : *$$

Если  $R$  бинарный предикат над  $A$ , то  $\forall a \in A. R a a$  — высказывание.



$$A^*, P^{A \rightarrow *}, Q^{A \rightarrow *} \vdash \Pi a^A. P a \rightarrow Q a : *$$

Это высказывание утверждает, что предикат  $P$ , рассматриваемый как множество, вложен в предикат  $Q$ .



$$A^*, P^{A \rightarrow *}, \vdash \text{Па}^A. P a \rightarrow P a : *$$

Это высказывание утверждает рефлексивность вложения.

- Доказательство рефлексивности вложения:

$$A^*, P^{A \rightarrow *}, \vdash \lambda a^A. \lambda x^{P a}. x : \text{Па}^A. P a \rightarrow P a : *$$

- Дерево вывода:

$$\frac{\frac{A^*, P^{A \rightarrow *}, a^A, x^{P a} \vdash x : P a}{A^*, P^{A \rightarrow *}, a^A \vdash \lambda x^{P a}. x : P a \rightarrow P a}}{A^*, P^{A \rightarrow *}, \vdash \lambda a^A. \lambda x^{P a}. x : \text{Па}^A. P a \rightarrow P a}$$

- В контексте

$$\Gamma \equiv A^*, P^{A \rightarrow *}, Q^*$$

имеем

$$\Gamma \vdash (\Pi a^A. P a \rightarrow Q) \rightarrow (\Pi a^A. P a) \rightarrow Q : *$$

$$\begin{aligned} \Gamma, a_0^A \vdash & \lambda f^{\Pi a^A. P a \rightarrow Q}. \lambda g^{\Pi a^A. P a}. f a_0 (g a_0) \\ & : \Pi f^{\Pi a^A. P a \rightarrow Q}. \Pi g^{\Pi a^A. P a}. Q \equiv \\ & (\Pi a^A. P a \rightarrow Q) \rightarrow (\Pi a^A. P a) \rightarrow Q \end{aligned}$$

- Это высказывание доказывает, что

$$(\forall a \in A. P a \Rightarrow Q) \Rightarrow (\forall a \in A. P a) \Rightarrow Q$$

является истинным для *непустого* множества  $A$ .

- Напомним, что  $\perp \equiv \Pi A^*. A \equiv \forall A. A$ .
- Пусть  $\Gamma \equiv A^*, R^{A \rightarrow A \rightarrow *}$ , тогда

$$\Gamma \vdash (\Pi a^A. \Pi b^A. R a b \rightarrow R b a \rightarrow \perp) \rightarrow (\Pi a^A. R a a \rightarrow \perp) : *$$

- Это высказывание утверждает, что асимметричное бинарное отношение иррефлексивно

$$(\forall a, b \in A. R a b \Rightarrow \neg R b a) \Rightarrow \forall a \in A. \neg R a a$$

- Докажите это утверждение, приведя терм такого типа.

- В  $\lambda P$  можно диагонализировать имеющийся бинарный предикат

$$A^*, R^{A \rightarrow A \rightarrow *}, a^A \vdash R a a : * : \square$$

$$A^*, R^{A \rightarrow A \rightarrow *} \vdash \lambda a^A. R a a : A \rightarrow * : \square$$

- В  $\lambda P_{\omega}$  можно задать оператор диагонализации, абстрагируясь по предикату

$$A^* \vdash \lambda R^{A \rightarrow A \rightarrow *}. \lambda a^A. R a a : \\ (A \rightarrow A \rightarrow *) \rightarrow (A \rightarrow *) : \square$$

и по домену

$$\vdash \lambda A^* R^{A \rightarrow A \rightarrow *} a^A. P a a : \\ \Pi A^* R^{A \rightarrow A \rightarrow *} a^A. * : \square$$



- Переведите на язык  $\lambda P$  и докажите следующие тавтологии логики предикатов

$$(\forall a \in A. Q a) \Rightarrow \forall b \in A. P b \Rightarrow Q b$$

$$(\forall a \in A. P a \Rightarrow Q a) \Rightarrow (\forall b \in A. P b) \Rightarrow \forall c \in A. Q c$$

- Nederplet, Geuvers, TTFP, задачи 5.10-5.12