Типы в языках программирования
Лекция 10. Зависимые типы: система λP

Денис Николаевич Москвин

СПбАУ РАН

17.05.2018

План лекции

1. Зависимые типы
2. Система λP: формализм
3. Минимальная логика предикатов в λP

План лекции

Зависимые типы: вектор

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

$$
\begin{aligned}
[\,] &: \texttt{Vect}\ 0 \\
[\texttt{fls}] &: \texttt{Vect}\ 1 \\
[\texttt{tru}, \texttt{fls}] &: \texttt{Vect}\ 2 \\
[\texttt{fls}, \texttt{tru}, \texttt{fls}] &: \texttt{Vect}\ 3 \\
&\ldots
\end{aligned}
$$

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

Зависимые типы: кайнды

- При этом могут возникнуть бессмысленные конструкции:

$$\texttt{Vect tru}, \quad \texttt{Vect (pair 3 5)}$$

- Для контроля за ними расширим понятие кайнда

$$\texttt{Vect} : \texttt{Nat} \longrightarrow *$$

- Правило типизации

$$\frac{\Gamma \vdash \varphi : \sigma \longrightarrow \kappa \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash \varphi\, N : \kappa}$$

позволяет осуществлять контроль соответствия:

$$\begin{aligned} \texttt{Vect 3} &: * \\ \texttt{Vect tru} &: \text{ошибка} \end{aligned}$$

Зависимые типы: тип, зависящий от терма

- Стандартные конструкторы вектора — их бесконечно много?

$$
\begin{aligned}
\texttt{vnil} &: \texttt{Vect}\ 0 \\
\texttt{vcons0} &: \sigma \to \texttt{Vect}\ 0 \to \texttt{Vect}\ 1 \\
\texttt{vcons1} &: \sigma \to \texttt{Vect}\ 1 \to \texttt{Vect}\ 2 \\
\texttt{vcons2} &: \sigma \to \texttt{Vect}\ 2 \to \texttt{Vect}\ 3
\end{aligned}
$$

$$
\begin{aligned}
[\,] &\equiv \texttt{vnil} \\
[\texttt{fls}] &\equiv \texttt{vcons0}\ \texttt{fls}\ \texttt{vnil} \\
[\texttt{tru}, \texttt{fls}] &\equiv \texttt{vcons1}\ \texttt{tru}\ (\texttt{vcons0}\ \texttt{fls}\ \texttt{vnil}) \\
[\texttt{fls}, \texttt{tru}, \texttt{fls}] &\equiv \texttt{vcons2}\ \texttt{fls}\ (\texttt{vcons1}\ \texttt{tru}\ (\texttt{vcons0}\ \texttt{fls}\ \texttt{vnil}))
\end{aligned}
$$

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

$$\texttt{vcons} : \Pi \texttt{n:Nat}.\ \sigma \to \texttt{Vect}\ \texttt{n} \to \texttt{Vect}\ (\texttt{succ}\ \texttt{n})$$

Зависимые типы: тип зависимого произведения

- 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 σ в терм типа $\tau[x := N]$.
- Π-тип обобщает обычный инфиксный конструктор типа функции $\to : * \to * \to *$.
- Мы можем объявить второй «синтаксическим сахаром» для первого:

$$\sigma \to \tau \equiv \Pi\_{:}\sigma.\,\tau$$

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

Введение зависимого произведения

Правило введения Π-типа:

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

Например,

$$\begin{array}{lcl} \mathtt{twice} & : & \Pi n^{\mathtt{Nat}}.\, \sigma \to \mathtt{Vect}\ n \to \mathtt{Vect}\ (\mathtt{succ}\ (\mathtt{succ}\ n)) \\ \mathtt{twice} & \equiv & \lambda n^{\mathtt{Nat}}\, e^{\sigma}\, v^{\mathtt{Vect}\ n}.\, \mathtt{vcons}\ (\mathtt{succ}\ n)\ e\ (\mathtt{vcons}\ n\ e\ v) \end{array}$$

$$[\mathtt{fls}, \mathtt{tru}, \mathtt{fls}] : \mathtt{Vect}\ 3$$

$$\mathtt{twice}\ 3\ \mathtt{tru}\ [\mathtt{fls}, \mathtt{tru}, \mathtt{fls}] : \mathtt{Vect}\ 5$$

$$\mathtt{twice}\ 3\ \mathtt{tru}\ [\mathtt{fls}, \mathtt{tru}, \mathtt{fls}] \to_{\beta} [\mathtt{tru}, \mathtt{tru}, \mathtt{fls}, \mathtt{tru}, \mathtt{fls}]$$

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

Полное определение вектора в $\lambda P2$

$$
\begin{array}{rcl}
\mathtt{Vect} & : & \mathtt{Nat} \to * \\
\mathtt{Vect} & \equiv & \lambda m^{\mathtt{Nat}}.\, \Pi \varphi^{\mathtt{Nat}\to *}.\, \varphi\, 0 \to (\Pi n^{\mathtt{Nat}}.\, \sigma \to \varphi\, n \to \varphi\, (\mathtt{succ}\ n)) \to \varphi\, m \\
\mathtt{List} & \equiv & \Pi \alpha^{*}.\, \alpha \to (\sigma \to \alpha \to \alpha) \to \alpha
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{vnil} & : & \mathtt{Vect}\ 0 \\
\mathtt{vnil} & \equiv & \lambda \varphi^{\mathtt{Nat}\to *}\, z^{\varphi\, 0}\, c^{\Pi n^{\mathtt{Nat}}.\, \sigma \to \varphi\, n \to \varphi\, (\mathtt{succ}\ n)}.\, z \\
\mathtt{nil} & \equiv & \lambda \alpha^{*}.\, z^{\alpha}\, c^{\sigma \to \alpha \to \alpha}.\, z
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{vcons} & : & \Pi n^{\mathtt{Nat}}.\, \sigma \to \mathtt{Vect}\ n \to \mathtt{Vect}\ (\mathtt{succ}\ n) \\
\mathtt{vcons} & \equiv & \lambda n^{\mathtt{Nat}}\, e^{\sigma}\, v^{\mathtt{Vect}\ n}\, \varphi^{\mathtt{Nat}\to *}\, z^{\varphi\, 0}\, c^{\Pi n^{\mathtt{Nat}}.\, \sigma \to \varphi\, n \to \varphi\, (\mathtt{succ}\ n)}.\, c\, n\, e\, (v\, \varphi\, z\, c) \\
\mathtt{cons} & \equiv & \lambda\, e^{\sigma}\, v^{\mathtt{List}}\, \alpha^{*}\, z^{\alpha}\, c^{\sigma \to \alpha \to \alpha}.\, c\ \ e\, (v\, \alpha\, z\, c)
\end{array}
$$

Вектор в $\lambda P\omega$

$$
\begin{aligned}
\texttt{Vec} &: \texttt{Nat} \to * \to * \\
\texttt{Vec} &\equiv \lambda m^{\texttt{Nat}}.\, \lambda \sigma^{*}.\, \Pi \varphi^{\texttt{Nat} \to *}.\, \varphi\, 0 \to (\Pi n^{\texttt{Nat}}.\, \sigma \to \varphi\, n \to \varphi\, (\texttt{succ}\ n)) \to \varphi\, m \\
\texttt{vnil} &: \texttt{Vec}\ 0\ \sigma \\
\texttt{vcons} &: \Pi n{:}\texttt{Nat}.\, \sigma \to \texttt{Vec}\ n\ \sigma \to \texttt{Vec}\ (\texttt{succ}\ n)\ \sigma
\end{aligned}
$$

Примеры функций:

$$
\begin{aligned}
\texttt{replicate} &: \Pi n{:}\texttt{Nat}.\, \sigma \to \texttt{Vec}\ n\ \sigma \\
\texttt{head} &: \Pi n{:}\texttt{Nat}.\, \texttt{Vec}\ (\texttt{succ}\ n)\ \sigma \to \sigma \\
\texttt{concat} &: \Pi m{:}\texttt{Nat}.\, \Pi n{:}\texttt{Nat}.\, \texttt{Vec}\ m\ \sigma \to \texttt{Vec}\ n\ \sigma \to \texttt{Vec}\ (m + n)\ \sigma \\
\texttt{zip} &: \Pi n{:}\texttt{Nat}.\, \texttt{Vec}\ n\ \sigma \to \texttt{Vec}\ n\ \tau \to \texttt{Vec}\ n\ \langle \sigma, \tau \rangle \\
\texttt{zip}' &: \Pi m\, n{:}\texttt{Nat}.\, \texttt{Vec}\ m\ \sigma \to \texttt{Vec}\ n\ \tau \to \texttt{Vec}\ \min(m, n)\ \langle \sigma, \tau \rangle
\end{aligned}
$$

План лекции

Типы и термы в λP

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

Контексты для λP

- *Высказывание* в λP имеет вид $M : A$, где $M, A \in \mathcal{T}$.
- *Контекст* — это конечное, линейно упорядоченное множество высказываний, с различными переменными в качестве субъекта.
- $\langle\rangle$ обозначает пустой контекст.
- Если $\Gamma = \langle x_1 : A_1, \ldots, x_n : A_n \rangle$, то $\Gamma, y : B = \langle x_1 : A_1, \ldots, x_n : A_n, y : B \rangle$.
- (Полностью идентично $\lambda\underline{\omega}$.)

Правила типизации для λP (1)

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

(аксиома) $$\langle\rangle \vdash * : \Box$$

(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...

Правила типизации для λP (2)

...начало на предыдущем слайде

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

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

Правило формирования: тип

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

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

$$\frac{\Gamma \vdash \sigma{:}* \qquad \Gamma, x{:}\sigma \vdash \tau{:}*}{\Gamma \vdash (\Pi x{:}\sigma.\, \tau) : *}$$

Если тип τ не зависит от переменной x, то стрелочную нотацию удобно восстановить как удобный синтаксический сахар:

$$\sigma \to \tau \;\equiv\; \Pi \_ {:}\sigma.\, \tau \;\equiv\; \Pi x{:}\sigma.\, \tau$$

Например,

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

Ниже мы увидим примеры с зависимыми типами.

Правило формирования: кайнд

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

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

$$\frac{\Gamma \vdash \sigma{:}* \qquad \Gamma, x{:}\sigma \vdash \kappa{:}\Box}{\Gamma \vdash (\Pi x{:}\sigma.\, \kappa) : \Box}$$

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

$$\sigma \to \kappa \;\equiv\; \Pi\_{:}\sigma.\, \kappa \;\equiv\; \Pi x{:}\sigma.\, \kappa$$

Например,

$$\alpha{:}* \;\vdash\; \alpha \to * : \Box$$

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

Правило применения: терм

$$\frac{\Gamma \vdash M : (\Pi x : A.\, B) \qquad \Gamma \vdash N : A}{\Gamma \vdash M\, N : B[x := N]} \quad \text{(правило применения)}$$

При $(\Pi x : A.\, B) : *$ имеем удаление Π-типа:

$$\frac{\Gamma \vdash M : (\Pi x : \sigma.\, \tau) \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash M\, N : \tau[x := N]}$$

Правило применения: тип

$$\frac{\Gamma \vdash M : (\Pi x : A.\, B) \qquad \Gamma \vdash N : A}{\Gamma \vdash M\, N : B[x := N]} \quad \text{(правило применения)}$$

При $(\Pi x : A.\, B) : \Box$, имеем удаление Π-кайнда:

$$\frac{\Gamma \vdash \varphi : (\Pi x : \sigma.\, \kappa) \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash \varphi\, N : \kappa[x := N]}$$

Например,

$$\alpha : *,\ \varphi : \alpha \to *,\ a : \alpha \ \vdash\ \varphi\, a : *$$
$$\alpha : *,\ \varphi : \alpha \to *,\ a : \alpha \ \vdash\ \varphi\, a \to * : \Box$$
$$\alpha : *,\ \varphi : \alpha \to * \ \vdash\ \Pi a : \alpha.\, \varphi\, a \to * : \Box$$

$$\alpha : *,\ \varphi : \alpha \to *,\ \psi : \Pi a : \alpha.\, \varphi\, a \to *,\ x : \alpha \ \vdash\ \psi\, x : \varphi\, x \to *$$
$$\alpha : *,\ \varphi : \alpha \to *,\ \psi : \Pi a : \alpha.\, \varphi\, a \to *,\ x : \alpha,\ y : \varphi\, x \ \vdash\ \psi\, x\, y : *$$

Правило абстракции: терм

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

При $s = *$ (и, следовательно, $B{:}*$) возникает правило введения Π-типа:

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

Например,

$$\begin{aligned} \alpha{:}*,\ \varphi{:}\alpha\to *,\ a{:}\alpha,\ x{:}\varphi\, a \ &\vdash\ x : \varphi\, a \\ \alpha{:}*,\ \varphi{:}\alpha\to *,\ a{:}\alpha \ &\vdash\ \lambda x^{\varphi\, a}.\, x : \varphi\, a \to \varphi\, a \\ \alpha{:}*,\ \varphi{:}\alpha\to * \ &\vdash\ \lambda a^{\alpha}.\, \lambda x^{\varphi\, a}.\, x : \Pi a{:}\alpha.\, \varphi\, a \to \varphi\, a \end{aligned}$$

Правило абстракции: тип

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

При $s = \Box$ (и, следовательно, $B{:}\Box$) возникает правило введения Π-кайнда

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

Примеры:

$$\alpha{:}*,\ \psi{:}\alpha \to \alpha \to *,\ a{:}\alpha \ \vdash\ \psi\, a : \alpha \to *$$

$$\alpha{:}*,\ \psi{:}\alpha \to \alpha \to *,\ a{:}\alpha \ \vdash\ \psi\, a\, a : *$$

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

Правило преобразования

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

Алгоритмически неудобно, в реализации будет встроено в правило применения

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

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

$$\varphi : \mathtt{Nat} \to *, y : \varphi(4+1) \vdash (\lambda x^{\varphi(2+3)} . x)\, y : ?$$

План лекции

Логика предикатов

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

$$S : *$$

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

- Высказывания тоже кодируются как типы:

$$A : *$$

Например, $\alpha \to \alpha : *$ — тавтология $\alpha \Rightarrow \alpha$ с доказательством $\lambda x^{\alpha}.\, x$.

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

$$P : S \to *$$

Минимальная логика предикатов

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

$$\Gamma \equiv A{:}*,\ a{:}A,\ f{:}A \to A,\ P{:}A \to *,\ Q{:}A \to *,\ R{:}A \to A \to *$$

- **Связки**: импликация представляется как $\to$, а квантор $\forall$ как Π:

$$\begin{aligned} \forall x \in A.\, P\, x &\sim \Pi x{:}A.\, P\, x \\ \forall x \in A.\, R\, x\, x \Rightarrow P\, x &\sim \Pi x{:}A.\, R\, x\, x \to P\, x \end{aligned}$$

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

Прямое представление логики в λP

- $$A : * \ \vdash \ A \to * : \Box$$

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

- $$A : *, \ P : A \to *, \ a : A \ \vdash \ P\, a : *$$

  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:

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

Прямое представление логики в λP

- $$A{:}*,\ R{:}A \to A \to *\ \vdash\ \Pi a{:}A.\, R\, a\, a : *$$

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

- $$A{:}*,\ P{:}A \to *,\ Q{:}A \to *\ \vdash\ \Pi a{:}A.\, P\, a \to Q\, a : *$$

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

Прямое представление логики в λP

- $$A{:}*,\ P{:}A \to * \ \vdash\ \Pi a{:}A.\, P\, a \to P\, a : *$$

  Это высказывание утверждает рефлексивность вложения.
- Доказательство рефлексивности вложения:

$$A{:}*,\ P{:}A \to * \ \vdash\ \lambda a{:}A.\, \lambda x{:}P\, a.\, x : \Pi a{:}A.\, P\, a \to P\, a : *$$

- Дерево вывода:

$$\frac{\dfrac{A{:}*,\ P{:}A \to *,\ a{:}A,\ x{:}P\, a \ \vdash\ x : P\, a}{A{:}*,\ P{:}A \to *,\ a{:}A \ \vdash\ \lambda x{:}P\, a.\, x : P\, a \to P\, a}}{A{:}*,\ P{:}A \to * \ \vdash\ \lambda a{:}A.\, \lambda x{:}P\, a.\, x : \Pi a{:}A.\, P\, a \to P\, a}$$

Прямое представление логики в λP

- В контексте

$$\Gamma \equiv A{:}*,\ P{:}A \to *,\ Q{:}*$$

имеем

$$\Gamma \vdash (\Pi a{:}A.\, P\, a \to Q) \to (\Pi a{:}A.\, P\, a) \to Q : *$$

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

- 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.

Прямое представление логики в $\lambda P2$

- Напомним, что $\bot \equiv \Pi A{:}*.\, A \equiv \forall A.\, A$.
- Пусть $\Gamma \equiv A{:}*,\ R{:}A \to A \to *$, тогда

$$\Gamma \ \vdash \ (\Pi a{:}A.\, \Pi b{:}A.\, R\, a\, b \to R\, b\, a \to \bot) \to (\Pi a{:}A.\, R\, a\, a \to \bot) : *$$

- Это высказывание утверждает, что асимметричное бинарное отношение иррефлексивно

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

- Докажите это утверждение, приведя терм такого типа.

Прямое представление логики в $\lambda P\underline{\omega}$

- В λP можно диагонализовать имеющийся бинарный предикат

$$A:*,\ P:A\to A\to *,\ a:A \ \vdash\ P\,a\,a:*:\Box$$
$$A:*,\ P:A\to A\to * \ \vdash\ \lambda a:A.\,P\,a\,a:A\to *:\Box$$

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

$$A:* \vdash \lambda P:A\to A\to *.\,\lambda a:A.\,P\,a\,a:$$
$$(A\to A\to *)\to (A\to *):\Box$$

и по домену

$$\vdash \lambda A:*.\,\lambda P:A\to A\to *.\,\lambda a:A.\,P\,a\,a:$$
$$\Pi A:*.\,\Pi P:A\to A\to *.\,\Pi a:A.\,*:\Box$$

Упражнения

- Переведите на язык λP 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