# Курс: Функциональное программирование

# Лекция 11. Алгоритм вывода типов

**Денис Николаевич Москвин**

09.12.2011

Кафедра математических и информационных технологий
Санкт-Петербургского академического университета

# Система $\lambda\!\to$ а ля Карри

| Предтермы: | Редукция: |
|---|---|
| $\Lambda ::= \quad V$<br>$\quad\quad \mid \quad M\,N$<br>$\quad\quad \mid \quad \lambda x.\, M$ | $(\lambda x.\, M)\,N \to_\beta M[x := N]$ |
| **Типы:**<br>$\mathbb{T} ::= \quad \mathbb{V}$<br>$\quad\quad \mid \quad \sigma \to \tau$<br>**Контексты:**<br>$\Gamma ::= \quad \varnothing$<br>$\quad\quad \mid \quad \Gamma, x{:}\sigma$ | **Типизация:**<br>$\dfrac{x{:}\sigma \in \Gamma}{\Gamma \vdash x{:}\sigma}$<br>$\dfrac{\Gamma \vdash M{:}\sigma\to\tau \quad \Gamma \vdash N{:}\sigma}{\Gamma \vdash (M\,N){:}\tau}$<br>$\dfrac{\Gamma, x{:}\sigma \vdash M{:}\tau}{\Gamma \vdash (\lambda x.\, M){:}\sigma\to\tau}$ |

Здесь $V = \{a, b, \ldots\}$, $\mathbb{V} = \{\alpha, \beta, \ldots\}$ и $x \in V$; $M, N \in \Lambda$; $\sigma, \tau \in \mathbb{T}$.

# Система $\lambda\!\to$ а ля Чёрч

| Предтермы: | Редукция: |
|---|---|
| $\Lambda_{\mathbb{T}} ::= V \mid M\,N \mid \lambda x:\sigma.\,M$ | $(\lambda x:\sigma.\,M)\,N \to_\beta M[x := N]$ |
| **Типы:** $\mathbb{T} ::= \mathbb{V} \mid \sigma\to\tau$ <br> **Контексты:** $\Gamma ::= \varnothing \mid \Gamma, x:\sigma$ | **Типизация:** $\dfrac{x:\sigma \in \Gamma}{\Gamma \vdash x:\sigma}$ <br> $\dfrac{\Gamma \vdash M:\sigma\to\tau \quad \Gamma \vdash N:\sigma}{\Gamma \vdash (M\,N):\tau}$ <br> $\dfrac{\Gamma, x:\sigma \vdash M:\tau}{\Gamma \vdash (\lambda x:\sigma.\,M):\sigma\to\tau}$ |

Здесь $V = \{a, b, \ldots\}$, $\mathbb{V} = \{\alpha, \beta, \ldots\}$ и $x \in V$; $M, N \in \Lambda_{\mathbb{T}}$; $\sigma, \tau \in \mathbb{T}$.

## Главный тип (principle type)

В версии Чёрча $\lambda{\to}$ термы атрибутированны типами, поэтому тип терма единственен. Для $\mathbf{S}_{\sigma\tau\rho} \equiv \lambda f^{\sigma\to\tau\to\rho}\, g^{\sigma\to\tau}\, z^{\sigma}.\, f\, z\, (g\, z)$:

$$\mathbf{S}_{\sigma\tau\rho} : (\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho$$
$$\mathbf{S}_{\sigma\tau\sigma} : (\sigma \to \tau \to \sigma) \to (\sigma \to \tau) \to \sigma \to \sigma$$
$$\mathbf{S}_{(\tau\to\rho)\tau\rho} : ((\tau \to \rho) \to \tau \to \rho) \to ((\tau \to \rho) \to \tau) \to (\tau \to \rho) \to \rho$$

Любой из этих типов можно присвоить терму $\mathbf{S} \equiv \lambda f\, g\, z.\, f\, z\, (g\, z)$ в версии Карри.

Однако, первый «лучше» в том смысле, что остальные получаются из него подстановкой типа вместо типовой переменной.

# Вывод главного типа (пример)

$$\lambda x\, y.\, y\,(\lambda z.\, y\, x) \qquad\qquad \lambda x^{\alpha}\, y^{\beta}.\, \underbrace{y^{\beta}\,(\lambda z^{\gamma}.\, \overbrace{y^{\beta}\, x^{\alpha}}^{\delta})}_{\varepsilon}$$

1. Присвоим типовую переменную всем термовым переменным: $x^{\alpha}, y^{\beta}, z^{\gamma}$.
2. Присвоим типовую переменную всем *аппликативным* подтермам: $(y\, x){:}\delta$, $(y\,(\lambda z.\, y\, x)){:}\varepsilon$.
3. Выпишем уравнения (ограничения) на типы, необходимые для типизируемости терма: $\beta \sim \alpha \to \delta$, $\beta \sim (\gamma \to \delta) \to \varepsilon$.
4. Найдём *главный унификатор* для типовых переменных (подстановку), дающий решения уравнений:
$\alpha := \gamma \to \delta$, $\beta := (\gamma \to \delta) \to \varepsilon$, $\delta := \varepsilon$.
5. Главный тип $(\lambda x\, y.\, y\,(\lambda z.\, y\, x)){:}(\gamma \to \varepsilon) \to ((\gamma \to \varepsilon) \to \varepsilon) \to \varepsilon$.

## Подстановка типов

***Подстановка типов*** — это операция $S{:}\mathbb{T} \to \mathbb{T}$, такая что

$$S(\sigma \to \tau) \;\equiv\; S(\sigma) \to S(\tau)$$

Обычно подстановка тождественна на всех типовых переменных, кроме конечного носителя $\sup(S) = \{\alpha \mid S(\alpha) \not\equiv \alpha\}$.

Пример подстановки $S = [\alpha := \gamma \to \beta, \beta := \alpha \to \gamma]$.

Тождественную подстановку (с пустым носителем) обозначим $[]$.

Подстановка выполняется *параллельно*; для $\tau = \alpha \to \beta \to \gamma$

$$\begin{aligned} S(\tau) &= [\alpha := \gamma \to \beta, \beta := \alpha \to \gamma](\alpha \to \beta \to \gamma) \\ &= (\gamma \to \beta) \to (\alpha \to \gamma) \to \gamma \end{aligned}$$

## Композиция подстановок

***Композиция подстановок*** — подстановка с носителем, являющимся объединением носителей. Для

$$\begin{aligned} S &= [\alpha := \gamma \to \beta, \beta := \alpha \to \gamma]; \\ T &= [\alpha := \beta \to \gamma, \gamma := \beta] \text{ имеем} \\ T \circ S &= [\alpha := T(S(\alpha)), \beta := T(S(\beta)), \gamma := T(S(\gamma))], \text{ то есть} \\ T \circ S &= [\alpha := \beta \to \beta, \beta := (\beta \to \gamma) \to \beta, \gamma := \beta] \end{aligned}$$

Подстановки — моноид относительно $\circ$ с единицей $[]$.

Сравним, например, $(T \circ S)(\tau)$ и $T(S(\tau))$ для $\tau = \alpha \to \beta \to \gamma$

$$\begin{aligned} (T \circ S)(\tau) &= (\beta \to \beta) \to ((\beta \to \gamma) \to \beta) \to \beta; \\ T(S(\tau)) &= T((\gamma \to \beta) \to (\alpha \to \gamma) \to \gamma) \\ &= (\beta \to \beta) \to ((\beta \to \gamma) \to \beta) \to \beta \end{aligned}$$

## Унификатор

***Унификатор*** для типов $\sigma$ и $\tau$ — это подстановка $S$, такая что $S(\sigma) \equiv S(\tau)$.

Пример. Пусть $\sigma = \beta \to \alpha \to \beta$ и $\tau = (\gamma \to \gamma) \to \delta$. Их унификатор

$$\begin{aligned} S &= [\beta := \gamma \to \gamma, \delta := \alpha \to \gamma \to \gamma] \\ S(\sigma) \equiv S(\tau) &= (\gamma \to \gamma) \to \alpha \to \gamma \to \gamma \end{aligned}$$

Унификатор $S$ — это ***главный унификатор*** для $\sigma$ и $\tau$, если для любого другого унификатора $S'$ существует подстановка T, такая что

$$S' \equiv T \circ S$$

Пример.

$$\begin{aligned} S' &= [\beta := \gamma \to \gamma, \alpha := \varepsilon \to \varepsilon, \delta := (\varepsilon \to \varepsilon) \to \gamma \to \gamma] \\ S' &= [\alpha := \varepsilon \to \varepsilon] \circ S \end{aligned}$$

## Теорема унификации

**Теорема унификации.** Существует алгоритм унификации $\mathcal{U}$, который для заданных типов $\sigma$ и $\tau$ возвращает:

- главный унификатор $S$ для $\sigma$ и $\tau$, если $\sigma$ и $\tau$ могут быть унифицированы;
- сообщение об ошибке в противном случае.

Алгоритм $\mathcal{U}(\sigma, \tau)$ позволяет искать «минимальное» решение уравнения на типы $\sigma \sim \tau$.

Ключевой момент всех рассуждений про унификацию:

$$\sigma_1 \to \sigma_2 \equiv \tau_1 \to \tau_2 \;\Leftrightarrow\; \sigma_1 \equiv \tau_1 \;\wedge\; \sigma_2 \equiv \tau_2$$

## Вывод типов : алгоритм унификации $\mathcal{U}$

$$
\begin{array}{lcl}
\mathcal{U}(\alpha, \alpha) & = & [\,] \\
\mathcal{U}(\alpha, \tau) \mid \alpha \in \mathrm{FV}(\tau) & = & \text{ошибка} \\
\mathcal{U}(\alpha, \tau) \mid \alpha \notin \mathrm{FV}(\tau) & = & [\alpha := \tau] \\
\mathcal{U}(\sigma_1 \to \sigma_2, \alpha) & = & \mathcal{U}(\alpha, \sigma_1 \to \sigma_2) \\
\mathcal{U}(\sigma_1 \to \sigma_2, \tau_1 \to \tau_2) & = & \mathcal{U}\left(\mathcal{U}(\sigma_2, \tau_2)\sigma_1, \mathcal{U}(\sigma_2, \tau_2)\tau_1\right) \circ \mathcal{U}(\sigma_2, \tau_2)
\end{array}
$$

▶ $\mathcal{U}(\sigma, \tau)$ завершается, поскольку на каждом шаге сокращает либо количество $\to$, либо количество типовых переменных.

▶ $\mathcal{U}(\sigma, \tau)$ унифицирует. По индукции; используем, что если $S$ унифицирует $(\sigma, \tau)$, то $S \circ [\alpha := \rho]$ унифицирует $(\sigma \to \alpha, \tau \to \rho)$.

▶ $\mathcal{U}(\sigma, \tau)$ даёт главный унификатор. По индукции; см. [TAPL 22.4]

## Унификация системы уравнений на типы

Если есть система уравнений $E = \{\sigma_1 \sim \tau_1, \ldots, \sigma_n \sim \tau_n\}$, то её унификатором называют такую подстановку $S$, что

$$S(\sigma_1) \equiv S(\tau_1) \ \wedge \ \ldots \ \wedge \ S(\sigma_n) \equiv S(\tau_n)$$

При этом пишут $S \models E$ (подстановка $S$ унифицирует систему $E$).

Главный унификатор для системы $E$ ищется с помощью алгоритма $U$ следующим образом:

$$U(E) = U(\sigma_1 \to \ldots \to \sigma_n, \tau_1 \to \ldots \to \tau_n)$$

## Алгоритм унификации $\mathcal{U}$: пример

$$
\begin{array}{lcl}
\mathcal{U}(\alpha, \alpha) & = & [\,] \\
\mathcal{U}(\alpha, \tau) \mid \alpha \in \mathrm{FV}(\tau) & = & \text{ошибка} \\
\mathcal{U}(\alpha, \tau) \mid \alpha \notin \mathrm{FV}(\tau) & = & [\alpha := \tau] \\
\mathcal{U}(\sigma_1 \to \sigma_2, \alpha) & = & \mathcal{U}(\alpha, \sigma_1 \to \sigma_2) \\
\mathcal{U}(\sigma_1 \to \sigma_2, \tau_1 \to \tau_2) & = & \mathcal{U}\,(\mathcal{U}(\sigma_2, \tau_2)\sigma_1, \mathcal{U}(\sigma_2, \tau_2)\tau_1) \circ \mathcal{U}(\sigma_2, \tau_2)
\end{array}
$$

Для $\lambda x\, y.\, y\,(\lambda z.\, y\, x)$ система уравнений на типы имела вид $E = \{\beta \sim (\gamma \to \delta) \to \varepsilon,\ \beta \sim \alpha \to \delta\}$. Алгоритм $\mathcal{U}$ даёт:

$$
\begin{array}{rcl}
\mathcal{U}(E) & = & \mathcal{U}(\beta \to \beta, ((\gamma \to \delta) \to \varepsilon) \to (\alpha \to \delta)) \\
& = & \mathcal{U}(\mathcal{U}(\beta, \alpha \to \delta)\beta, \mathcal{U}(\beta, \alpha \to \delta)(\gamma \to \delta) \to \varepsilon) \circ \mathcal{U}(\beta, \alpha \to \delta) \\
& = & \mathcal{U}(\alpha \to \delta, (\gamma \to \delta) \to \varepsilon) \circ [\beta := \alpha \to \delta] \\
& = & [\alpha := \gamma \to \varepsilon] \circ [\delta := \varepsilon] \circ [\beta := \alpha \to \delta] \\
& = & [\alpha := \gamma \to \varepsilon, \delta := \varepsilon, \beta := (\gamma \to \varepsilon) \to \varepsilon]
\end{array}
$$

## Домашнее задание

1. Проследите за изменениями в работе алгоритма $\mathfrak{U}$, при перестановке элементов в E: $E = \{\beta \sim \alpha \to \delta, \beta \sim (\gamma \to \delta) \to \varepsilon\}$ Что изменится?

2. Реализуйте алгоритм $\mathfrak{U}$ на Haskell.

3. Реализуйте алгоритмы PP и PT на Haskell.

## Алгоритм построения ограничений

**Теорема.** Для любых контекста $\Gamma$, терма $M \in \Lambda$ ($FV(M) \subseteq \mathrm{dom}(\Gamma)$) и типа $\sigma \in \mathbb{T}$ существует такое множество уравнений на типы $E = E(\Gamma, M, \sigma)$, что для любой подстановки $S$:

- $S \models E(\Gamma, M, \sigma) \;\Rightarrow\; S(\Gamma) \vdash M : S(\sigma)$;
- $S(\Gamma) \vdash M : S(\sigma) \;\Rightarrow\; S' \models E(\Gamma, M, \sigma)$, для некоторой $S'$, имеющего тот же эффект, что и $S$, на типовых переменных в $\Gamma$ и $\sigma$.

Алгоритм $E$ можно задать так

$$\begin{aligned} E(\Gamma, x, \sigma) &= \{\sigma \sim \Gamma(x)\} \\ E(\Gamma, M\,N, \sigma) &= E(\Gamma, M, \alpha \to \sigma) \cup E(\Gamma, N, \alpha) \\ E(\Gamma, \lambda x.\, M, \sigma) &= E(\Gamma \cup \{x : \alpha\}, M, \beta) \cup \{\alpha \to \beta \sim \sigma\} \end{aligned}$$

Здесь $\alpha$ и $\beta$ — всякий раз «свежие»!

## Главная пара (Principle Pair) и главный тип (Principle Type)

Для $M \in \Lambda$ ***главной парой*** называют пару $(\Gamma, \sigma)$, такую что

- $\Gamma \vdash M : \sigma$
- $\Gamma' \vdash M : \sigma' \;\Rightarrow\; \exists S \;\left[S(\Gamma) \subseteq \Gamma' \;\wedge\; S(\sigma) \equiv \sigma'\right]$

Если $(\Gamma, \sigma) = PP(M)$, то $FV(M) = dom(\Gamma)$.

Пример. Для $M = \lambda x.\, x\, y$ имеем $PP(M) = (y : \alpha, (\alpha \to \beta) \to \beta)$

$$y : \alpha \vdash (\lambda x.\, x\, y) : (\alpha \to \beta) \to \beta$$

Для $M \in \Lambda^0$ ***главным типом*** называют тип $\sigma$, такой что

- $\vdash M : \sigma$
- $\vdash M : \sigma' \;\Rightarrow\; \exists S \;\left[S(\sigma) \equiv \sigma'\right]$

## Теорема Хиндли – Милнера

Существует алгоритм $\mathrm{PP}$, возвращающий для $M \in \Lambda$

- главную пару $(\Gamma, \sigma)$, если $M$ имеет тип;
- сообщение об ошибке в противном случае.

Пусть $\mathrm{FV}(M) = \{x_1, \ldots, x_n\}$, $\Gamma_0 = \{x_1 : \alpha_1, \ldots, x_n : \alpha_n\}$ и $\sigma_0 = \beta$.

Алгоритм $\mathrm{PP}$ можно задать так

$$\begin{array}{lcl} \mathrm{PP}(M) \mid \mathrm{U}(\mathrm{E}(\Gamma_0, M, \sigma_0)) = \text{ошибка} & = & \text{ошибка} \\ \mathrm{PP}(M) \mid \mathrm{U}(\mathrm{E}(\Gamma_0, M, \sigma_0)) = S & = & (S(\Gamma_0), S(\sigma_0)) \end{array}$$

## Теорема Хиндли – Милнера (следствие)

Существует алгоритм PT, возвращающий для $M \in \Lambda^0$

- главный тип $\sigma$, если $M$ имеет тип;
- сообщение об ошибке в противном случае.

**Доказательство.** Пусть $M$ замкнут и $PP(M) = (\Gamma, \sigma)$. Но $\Gamma = \varnothing$ и мы можем положить $PT(M) = \sigma$. ■

# к д/з

Определение лямбда-терма

```
type Sym = String

data Expr
        = Var Sym
        | App Expr Expr
        | Lam Sym Expr
        deriving (Eq, Read, Show)
```

То есть `App (Lam "x" $ Lam "y" $ Var "x") (Lam "z" $ Var "z")` кодирует $(\lambda x.\lambda y.\, x)\,(\lambda z.\, z)$.

# К Д/З (2)

Свободные переменные

```
freeVars :: Expr -> [Sym]
```

Попробуйте реализовать.

# к д/з (3)

Свободные переменные

```
freeVars :: Expr -> [Sym]
freeVars (Var v)     = [v]
freeVars (App t1 t2) = freeVars t1 `union` freeVars t2
freeVars (Lam v t)   = freeVars t \\ [v]
```

# К Д/З (4)

Типы

```
data Type
    = TVar Sym
    | Arrow Type Type
    deriving (Eq, Read, Show)
```

То есть `Arrow (Arrow (TVar "a") (TVar "b")) (TVar "c")`
кодирует $(a \to b) \to c$.

Удобно заменить конструктор данных `Arrow`
на инфиксный `:->`

# к д/з (5)

## Контексты

```
newtype Env = Env [(Sym, Type)] deriving Show

emptyEnv :: Env
emptyEnv = Env []

extend :: Sym -> Type -> Env -> Env
extend s t (Env r) = Env $ (s, t) : r
```