Лекция 4.
Экзистенциальные типы
Типовая система Хиндли-Милнера

Абстрактные типы данных

Стек α из значений типа υ: контейнер, соответствующий интерфейсу

метод	тип	комментарий
`empty`	α	(конструктор)
`push`	$\upsilon \to \alpha \to \alpha$	
`pop`	$\alpha \to \alpha \& \upsilon$	

Возможны разные реализации интерфейса.

Замечание: Мы понимаем АТД как набор функций, без собственных данных.
Напомним, что `a.method(...)` — другая запись для `method(a, ...)`.

Пример определения и применения АТД

```
abstype stack with
    empty : stack
    push : int * stack -> stack
    pop : stack -> stack * int
is pack Maybe Int,
    empty = None
    push (n,s) = Some n
    pop s = case s with None -> 0 | Some v -> v
in
    stack::pop(stack::push(12,stack::empty))
```

Экзистенциальные типы

Экзистенциальный тип — тип, соответствующий квантору существования в смысле изоморфизма Карри-Ховарда. Соответствует абстрактному типу данных.

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

АТД имеет интерфейс φ, тип АТД α реализуется типом θ, а сам интерфейс — термом M:

$$\frac{\Gamma \vdash M : \varphi[\alpha := \theta]}{\Gamma \vdash (\textsf{pack } M, \theta \textsf{ to } \exists \alpha.\varphi) : \exists \alpha.\varphi}$$

... и если вычисление $N : \psi$ работает при условии наличия какой-то реализации АТД $x : \varphi$ в контексте, то нам достаточно АТД $P : \exists \alpha.\varphi$ для получения результата:

$$\frac{\Gamma \vdash P : \exists \alpha.\varphi \qquad \Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash \textsf{abstype } \alpha \textsf{ with } x : \varphi \textsf{ is } P \textsf{ in } N : \psi} (\alpha \notin FV(\Gamma, \psi))$$

Стек в F

$$\frac{\Gamma \vdash M : \varphi[\alpha := \theta]}{\Gamma \vdash (\text{pack } M, \theta \text{ to } \exists\alpha.\varphi) : \exists\alpha.\varphi} \qquad \frac{\Gamma \vdash P : \exists\alpha.\varphi \qquad \Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash \text{abstype } \alpha \text{ with } x : \varphi \text{ is } P \text{ in } N : \psi}$$

Интерфейс стека (возьмём υ как чёрчевский нумерал):

$$\varphi := (\underbrace{\alpha}_{empty} \& \underbrace{(\upsilon\&\alpha \to \alpha)}_{push})\& \underbrace{(\alpha \to \alpha\&\upsilon)}_{pop}$$

Какое-нибудь вычисление — скажем, `pop(push(12,empty))`:

$$x : \varphi \vdash \underbrace{\pi_R\,((\pi_R x)((\pi_R(\pi_L x))\langle 12, \pi_L(\pi_L x)\rangle))}_{N} : \upsilon$$

И простая реализация, для $\theta := (\gamma \to \gamma) \vee \upsilon$ — это `Maybe Int`:

$$\vdash \langle\langle(In_L\ \lambda x.x), \lambda n.In_R\ (\pi_L\ n)\rangle, \lambda n.\mathtt{case}\ \ (\lambda x.0)\ (\lambda x.x)\ n\rangle : \varphi[\alpha := \theta]$$

Раскрываем $\exists$ через $\forall$

Напомним, что $\exists\alpha.\varphi := \forall\beta.(\forall\alpha.\varphi \to \beta) \to \beta$.

$$\frac{\Gamma \vdash M : \varphi[\alpha := \theta]}{\Gamma \vdash (\text{pack } M, \theta \text{ to } \exists\alpha.\varphi) : \exists\alpha.\varphi}$$

Перепишем это правило только через базовые конструкции системы F:

$$\frac{\Gamma \vdash M : \varphi[\alpha := \theta]}{\Gamma \vdash \Lambda\beta.\lambda e^{\forall\alpha.\varphi\to\beta}.(e\ \theta)\ M : \forall\beta.(\forall\alpha.\varphi \to \beta) \to \beta}$$

«Пусть есть вычисление e, использующее АТД α с интерфейсом φ, возвращающее β. Тогда, имея конкретный тип реализации АТД θ и саму реализацию АТД $M : \varphi[\alpha := \theta]$, то с помощью вычисления e возможно вычислить результат и вернуть значение типа β».
Сравните с case для алгебраического типа и вспомните действия редактора связей (линкера).

Раскроем abstype

$$\frac{\Gamma \vdash P : \exists \alpha.\varphi \qquad \Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash \text{abstype } \alpha \text{ with } x : \varphi \text{ is } P \text{ in } N : \psi}$$

Перепишем это правило через базовые конструкции системы F:

$$\frac{\Gamma \vdash P : \forall \beta.(\forall \alpha.\varphi \to \beta) \to \beta \qquad \Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash (P\ \psi)\ (\Lambda \alpha.\lambda x^{\varphi}.N) : \psi}$$

Вспомним pack:

$$\frac{\Gamma \vdash M : \varphi[\alpha := \theta]}{\Gamma \vdash \Lambda \beta.\lambda e^{\forall \alpha.\varphi \to \beta}.(e\ \theta)\ M : \forall \beta.(\forall \alpha.\varphi \to \beta) \to \beta}$$

Результат:

$$\begin{aligned}
&((\Lambda \beta.\lambda e^{\forall \alpha.\varphi \to \beta}.(e\ \theta)\ M)\ \psi)\ (\Lambda \alpha.\lambda x^{\varphi}.N) \to_\beta \\
&(\lambda e^{\forall \alpha.\varphi \to \psi}.(e\ \theta)\ M)\ (\Lambda \alpha.\lambda x^{\varphi}.N) \to_\beta \\
&(\Lambda \alpha.\lambda x^{\varphi}.N)\ \theta\ M \to_\beta \\
&(\lambda x^{\varphi[\alpha:=\theta]}.N[\alpha := \theta])\ M \to_\beta N[\alpha := \theta][x := M]
\end{aligned}$$

Пример реализации на Хаскеле

```
{-# LANGUAGE RankNTypes #-}
data AbstractStack = AS (forall b . (forall a .
     ( a, Integer -> a -> a, a -> (a, Integer) )
     -> b) -> b)

abstype :: AbstractStack -> Integer
abstype stack =
  case stack of
    AS r -> r x where
      x (empty, push, pop) =
        let (stk, v) = pop (push 12 $ push 5 empty) in
        let (stk2, v2) = pop stk in
        v + v2

packedStack :: AbstractStack
packedStack = AS (\t -> t ( [], \i -> \l -> i:l, \(i:l) -> (l,i) ) )

main = do print (abstype packedStack)
```

Общие свойства системы F

В системе F (в варианте по Чёрчу, так и в варианте по Карри) имеют место теорема Чёрча-Россера и сильная нормализация.

Разрешимость задач типизации системы F:

	По Чёрчу	По Карри
$\Gamma \vdash M : \sigma$	да	нет
$\Gamma \vdash M : ?$	да	нет
$\Gamma \vdash ? : \sigma$	нет	нет
$? \vdash M : \sigma$	нет	нет
$? \vdash M : ?$	нет	нет

Ранг типа

Напомним, что $\exists\alpha.\varphi := \forall\beta.(\forall\alpha.\varphi \to \beta) \to \beta$.

Определение

Функция «ранг типа» $rk \subseteq T \times \mathbb{N}_0$. $rk(\sigma) = [mrk(\sigma), +\infty) \cap \mathbb{N}_0$, где mrk:

$$mrk(\tau) = \begin{cases} 0, & \tau \text{ без кванторов} \\ \max(mrk(\sigma), 1), & \tau = \forall x.\sigma \\ \max(mrk(\sigma_1) + 1, mrk(\sigma_2)), & \tau = \sigma_1 \to \sigma_2 \end{cases}$$

Лемма

Если $rk(\sigma, 1)$, то для формулы σ найдётся эквивалентная формула с поверхностными кванторами.

Пример

$0 \notin rk(\forall\alpha.\gamma \to \beta)$; $1 \notin rk((\forall\alpha.\gamma \to \beta) \to f) = \{2, 3, \dots\}$

$1 \notin rk(\exists\alpha.\gamma) = rk(\forall\beta.(\forall\alpha.\gamma \to \beta) \to \beta) = \{2, 3, \dots\}$

$1 \in rk(\forall\alpha.\delta \to \forall\beta.\delta \to \forall\gamma.\delta)$

Типовая система Хиндли-Милнера: язык

Определение

Тип (τ) *и типовая схема:*

$$\tau ::= \alpha \mid (\tau \to \tau) \qquad \sigma ::= \forall x.\sigma \mid \tau$$

Пред-лямбда-терм (типизация по Карри)

$$H ::= x \mid (H\ H) \mid (\lambda x.H) \mid (\mathtt{let}\ x = H\ \mathtt{in}\ H)$$

Редукция для let:

$$\mathtt{let}\ x = E_1\ \mathtt{in}\ E_2 \to_\beta E_2[x := E_1]$$

Пример

$$\mathtt{let}\ Inc = \lambda n.\lambda f.\lambda x.n\ f\ (f\ x)\ \mathtt{in}\ Inc(Inc\ \overline{0}) \twoheadrightarrow_\beta \overline{2}$$

Типовая система Хиндли-Милнера: специализация

Определение

Пусть $\sigma_1 = \forall\alpha_1.\forall\alpha_2.\ldots.\forall\alpha_n.\tau_1$. *Тогда* σ_2 *— частный случай или специализация* σ_1 *(обозначается как* $\sigma_1 \sqsubseteq \sigma_2$*), если*

$$\sigma_2 = \forall\beta_1.\forall\beta_2.\ldots.\forall\beta_m.\tau_1[\alpha_1 := S(\alpha_1),\ldots,\alpha_n := S(\alpha_n)] \text{ и } \beta_i \notin FV(\forall\alpha_1.\forall\alpha_2.\ldots.\forall\alpha_n.\tau_1)$$

Пример

$$\forall\alpha.\alpha\to\alpha \sqsubseteq \forall\beta_1.\forall\beta_2.(\beta_1\to\beta_2)\to(\beta_1\to\beta_2)$$

Типовая система Хиндли-Милнера: правила вывода

$$\frac{}{\Gamma, x : \sigma \vdash x : \sigma}\ x \notin FV(\Gamma) \qquad \frac{\Gamma \vdash E_0 : \tau \to \tau' \qquad \Gamma \vdash E_1 : \tau}{\Gamma \vdash E_0\ E_1 : \tau'} \qquad \frac{\Gamma, x : \tau \vdash E : \tau'}{\Gamma \vdash \lambda x.E : \tau \to \tau'}$$

$$\frac{\Gamma \vdash E_0 : \sigma \qquad \Gamma, x : \sigma \vdash E_1 : \tau}{\Gamma \vdash let\ x = E_0\ in\ E_1 : \tau} \qquad \frac{\Gamma \vdash E : \sigma'}{\Gamma \vdash E : \sigma}\ \sigma' \sqsubseteq \sigma \qquad \frac{\Gamma \vdash E : \sigma}{\Gamma \vdash E : \forall \alpha.\sigma}\ \alpha \notin FV(\Gamma)$$

Пример

$$\cfrac{\cfrac{\overline{x : \alpha \vdash x : \alpha}}{\vdash \lambda x.x : \alpha \to \alpha}}{\vdash \lambda x.x : \forall \alpha.\alpha \to \alpha}$$

$$\cfrac{\cfrac{\overline{\mathtt{id} : \forall \alpha.\alpha \to \alpha \vdash \mathtt{id} : \forall \alpha.\alpha \to \alpha}}{\mathtt{id} : \forall \alpha.\alpha \to \alpha \vdash \mathtt{id} : \mathtt{int} \to \mathtt{int}}\ S(\alpha) = \mathtt{int} \qquad \mathtt{id} : \forall \alpha.\alpha \to \alpha \vdash 0 : \mathtt{int}}{\mathtt{id} : \forall \alpha.\alpha \to \alpha \vdash \mathtt{id}\ 0 : \mathtt{int}}$$

Отсюда: `let id=`$\lambda x.x$ in $\langle$`id 0`, `id «a»`$\rangle$: `int&string`

Алгоритм реконструкции типа W

На вход подаются Γ, M, на выходе наиболее общая пара: $\langle S, \tau\rangle = W(\Gamma, M)$

1. $M = x$, $x : \tau \in \Gamma$ (иначе ошибка)
   - τ' — τ без кванторов, все свободные переменные переименованы в свежие.

   возвращаем $\langle \varnothing, \tau' \rangle$; например, $W(\{x : \forall \alpha.\varphi, y : \beta\}, x) = \langle \varnothing, \varphi[\alpha := \gamma]\rangle$
2. $M = \lambda n.E$
   - $\Gamma' = \{x : \sigma \mid x : \sigma \in \Gamma, x \neq n\} \cup \{n : \alpha\}$, α — свежая типовая переменная
   - $\langle S,\ \tau\rangle = W(\Gamma', E)$

   возвращаем $\langle S, S(\alpha) \rightarrow \tau\rangle$
3. $M = P\ Q$
   - $\langle S_1, \tau_1\rangle = W(\Gamma, P)$; $\langle S_2, \tau_2\rangle = W(S_1(\Gamma), Q)$
   - $S_3 = \mathcal{U}\big[S_2(\tau_1), \tau_2 \rightarrow \alpha\big]$, α — свежая

   возвращаем $\langle S_3 \circ S_2 \circ S_1, S_3(\alpha)\rangle$
4. $M = (let\ \ n = P\ in\ Q)$
   - $\langle S_1, \tau_1\rangle = W(\Gamma, P)$
   - $\Gamma' = \{x : \sigma \mid x : \sigma \in \Gamma, x \neq n\} \cup \{n : \forall \alpha_1 \dots \alpha_k.\tau_1\}$, где $\alpha_1 \dots \alpha_k$ — все свободные переменные τ_1
   - $\langle S_2, \tau_2\rangle = W(S_1(\Gamma'), Q)$

   возвращаем $\langle S_2 \circ S_1, \tau_2\rangle$

Рекурсия в HM: делаем HM тьюринг-полной

1. Рекурсия для термов. Y-комбинатор. Добавим специальное правило вывода:

$$\frac{}{Y : \forall\alpha.(\alpha \to \alpha) \to \alpha}$$

2. Рекурсия для типов. Рассмотрим список

$$\texttt{Nil} = In_L\ 0 \qquad \texttt{Cons}\ e\ l = In_R\langle e, l\rangle \qquad \texttt{List} : ?$$

   Заметим, что при попытке выписать уравнение для типа мы получим рекурсию:

$$\tau = \texttt{Int} \vee \langle\texttt{Int}, \tau\rangle$$

   Рекурсивный тип надо добавить явно:

$$\tau = \mu\alpha.\texttt{Int} \vee \langle\texttt{Int}, \alpha\rangle$$

   Мю-оператор — это Y-комбинатор для типов. Как его добавить в типовую систему?

Эквирекурсивные и изорекурсивные типы: $\mu\alpha.\sigma(\alpha)$

- Эквирекурсивные типы. Считаем, что $\alpha = \sigma(\alpha)$. Например, в Java:

```
public abstract class Enum<E extends Enum<E>>
        implements Constable, Comparable<E>, Serializable
{ ... }
```

  Уравнение (частный случай): $E = Enum(E)$, или $E = \mu\varepsilon.Enum(\varepsilon)$.
- Изорекурсивные типы. $\alpha \neq \sigma(\alpha)$, но есть изоморфизм:

$$\texttt{roll} : \sigma(\alpha) \rightarrow \alpha \qquad \texttt{unroll} : \alpha \rightarrow \sigma(\alpha)$$

  Например, для `struct List { List* next; int value; }`:

Комп.	В C++	Пример
`roll`	взятие ссылки	`List a; a.next = NULL; return len(&a)`
`unroll`	разыменование	`len (List* a) { return (*a).next ? ... : 0 }`

Разрешимость задачи реконструкции типа в разных вариантах F

Ранг типов	Собственное название	Разрешимость
0	$\lambda_{\to}$	разрешимо (лекция 2)
1	HM	разрешимо (алгоритм W)
2		разрешимо
$\geqslant 3$		неразрешимо