Билеты к экзамену по функциональному программированию

Тамарин Вячеслав

January 28, 2021

1 Сравнение функционального и императивного подходов к программированию

1.1 Императивное программирование

Вычисление (программа) описывается в терминах **инструкций**, изменяющих **состояние** вычислителя.

- Инструкции выполняются последовательно;
- Состояние изменяется инструкциями **присваивания** значений изменяемым переменным;
- Если механизм условного исполнения (if, switch);
- Инструкции можно повторять с помощью циклов (while, for);
- Типы данных описываются с оглядкой на их физическое представление в памяти.

Такой стиль иногда называют стилем фон Неймана.

```
long factorial(int n) {
        long res = 1;
        for (int i = 0; i < n; i++)
                res *= i;
        return res;
}
```

Выполнение программы — переход вычислителя из начального состояния в конечное с помощью последовательных инструкций.

Часть конечного состояния может интерпретироваться как результат вычислений.

1.2 Функциональное программирование

Функциональная программа — **выражение**, ее выполнение — вычисление (**редукция**) этого выражения.

```
factorial n = if n == 0 then 1 else n * factorial(n-1)
```

Выполнение программы — редукция выражения с помощью **подстановки** определений функций в места их „вызова” с заменой формальных параметров на фактические.

```
factorial 3
-> if 3 == 0 then 1 else 3 * factorial (3 - 1)
-> ...
-> 3 * factorial (3 - 1)
-> 3 * if (3 - 1) == 0 then 1 else (3 - 1) * factorial ((3 - 1) - 1)
-> ...
-> 3 * 2 * 1 * 1
-> 6
```

- Нет состояний — нет изменяемых переменных;
- Нет переменных — нет присваивания;
- Нет циклов, так как нет различий между итерациями – состояниями
- Последовательность не важна, поскольку выражения независимы.
- Рекурсия — замена циклов;
- Функции высших порядков;
- Сопоставление с образцом;
- Все функции — чистые.

1.3 Сильные стороны ФП

- Регулярный синтаксис, удобство анализа кода;
- Мощная типизация, при этом можно практически не использовать типы в коде за счет эффективным алгоритмам вывода типов;
- Возможность генерации программ по набору свойств;
- Эффективная доказуемость свойств программ алгебраическими методами;
- Высокоурвневые оптимизации на базе эквивалентных преобразований.

2 Основы λ-исчисления. λ-термы, свободные и связанные переменные.

λ-исчисление — формальная система, лежащая в основе ФП. Разработано Алонзо Чёрчем в 1930-х для формализации и анализа понятия вычислимости.
В λ-исчислении тремя основными понятиями являются:

- применение (аппликация, application) — задает синтаксис применения функции к ее фактическим аргументам. Применение функции F к аргументу X записывается как FX;
- лямбда-абстрацкия (abstraction) — описывает синтаксис определения функции на основе параметризованного выражения, представляющего ее тело. Абстракция по x: $\lambda x.\, F$;
- редукция (reduction) — определяет отношение вычисления, основывающегося на подстановке фактических параметров вместо формальных.

Определение 1: λ-термы

Множество **λ-термов** Λ индуктивно строится из переменных $V = \{x, y, z, \ldots\}$ с помощью применения и абстракции:

$$\begin{aligned} x \in V &\implies x \in \Lambda \\ M, N \in \Lambda &\implies (MN) \in \Lambda \\ M \in \Lambda,\ x \in V &\implies (\lambda x.\, M) \in \Lambda \end{aligned}$$

В абстрактном синтаксисе

$$\Lambda ::= V \mid (\Lambda\Lambda) \mid (\lambda V.\, \Lambda).$$

Терм может быть *переменной* (например, $x \in \Lambda$), *аппликацией* (например, $(xy) \in \Lambda$) или *лямбда-абстракцией* (например, $\lambda x.\, ((x(yx))) \in \Lambda$).

Определение 2: Подтермы

Множество **подтермов** терма Q определяется индуктивно:

$$\begin{array}{lcl} \mathtt{subterms}(x) & = & \{x\} \\ \mathtt{subterms}(MN) & = & \{MN\} \cup \mathtt{subterms}(M) \cup \mathtt{subterms}(N) \\ \mathtt{subterms}(\lambda x.\, M) & = & \{\lambda x.\, M\} \cup \mathtt{subterms}(M) \end{array}$$

2.1 Соглашения

Общеприняты следующие соглашения для термов:

- Внешние скобки опускаются
- Применение ассоциативно *влево*:
$$FXYZ \equiv (((FX)Y)Z).$$
- Абстракция ассоциативна *вправо*:
$$\lambda xyz.\, M \equiv (\lambda x.\, (\lambda y.\, (\lambda z.\, M))).$$
- Тело абстракции простирается вправо насколько это возможно:
$$\lambda x.\, MNK \equiv \lambda x.\, (MNK).$$

Определение 3: Свободные переменные

Множество **свободных переменных** в терме Q ($FV(Q)$) определяется следующим образом:

$$\begin{array}{lcl} FV(x) & = & \{x\} \\ FV(MN) & = & FV(M) \cup FV(N) \\ FV(\lambda x.\, M & = & FV(M) \setminus \{x\} \end{array}$$

Определение 4: Связвнные переменные

Множество **связанных переменных** в терме Q ($BV(Q)$) определяется следующим образом:

$$\begin{array}{lcl} BV(x) & = & \varnothing \\ BV(MN) & = & BV(M) \cup BV(N) \\ BV(\lambda x.\, M & = & BV(M) \cup \{x\} \end{array}$$

2.2 Комбинаторы

Определение 5: Комбинатор

M — **замкнутый λ-терм (комбинатор)**, если $FV(M) = \varnothing$.
Множество замкнутых λ-термов обозначается Λ^0.

1. $I = \lambda x.\, x$
2. $\omega = \lambda x.\, xx$
3. $\Omega = \omega\omega = (\lambda x.\, xx)\,(\lambda x.\, xx)$
4. $K = \lambda xy.\, x$
5. $K_* = \lambda xy.\, y$
6. $C = \lambda fxy.\, fyx$
7. $B = \lambda fgx.\, f(gx)$
8. $S = \lambda fgx.\, fx(gx)$

3 Подстановка λ-терма. Лемма подстановки.

Определение 6: Подстановка

Подстановка терма N вместо *свободных* вхождений переменной x в терм M ($[x \mapsto N]M$) задается следующими правилами:

$$
\begin{array}{lcll}
[x \mapsto N]x & = & N & \\
[x \mapsto N]y & = & y & \\
[x \mapsto N]PQ & = & ([x \mapsto N]P)\,([x \mapsto N]Q) & \\
[x \mapsto N]\lambda x.\, P & = & \lambda x.\, P & \\
[x \mapsto N]\lambda y.\, P & = & \lambda y.\, [x \mapsto N]P, & \text{если } y \notin FV(N) \\
[x \mapsto N]\lambda y.\, P & = & \lambda x.\, [y \mapsto z]P\,([y \mapsto z]P)\,, & \text{если } y \in FV(N)
\end{array}
$$

Предполагается, что x и y различны, а z — „свежая” переменная, то есть $z \notin FV(P) \cup FV(N)$.

Пример 1 (Пример подстановки).

$$[x \mapsto uv]\,((\lambda x.\, (\lambda x.\, xz)x)\, x) = (\lambda x.\, (\lambda x.\, xz)x)\,(uv).$$

Лемма 1 (подстановки). Пусть $M, N, L \in \Lambda$, $x \not\equiv y$ и $x \notin FV(L)$. Тогда

$$[y \mapsto L][x \mapsto N]M = [x \mapsto [y \mapsto L]N][y \mapsto L]M.$$

4 α- и β-конверсии. η-конверсия и экстенсиональная эквивалентность.

4.1 α-эквивалентность

Позволяет менять переменную на „свежую”. Главная аксиома α-преобразования:

$$\lambda x.\, M =_\alpha \lambda y.\, [x \mapsto y]M, \text{ если } y \notin FV(M). \qquad \text{(правило } \alpha\text{)}$$

Аналогично β-эквивалентности можно определить совместимость.

4.2 β-эквивалентность

Основная схема аксиом: для любых $M, N \in \Lambda$:

$$(\lambda x.\, M)N =_\beta [x \mapsto N]M. \qquad \text{(правило } \beta\text{)}$$

Чтобы сделать $=_\beta$ отношением эквивалентности добавим логические аксиомы и правила:

$$
\begin{array}{lcl}
 & & M =_\beta M \\
M =_\beta N & \Rightarrow & M =_\beta N \\
M =_\beta N,\ N =_\beta L & \Rightarrow & M =_\beta L
\end{array}
$$

Также определим правила совместимости:

$$
\begin{array}{l}
M =_\beta N \Rightarrow ZM =_\beta ZN \\
M =_\beta N \Rightarrow MZ =_\beta NZ \\
M =_\beta N \Rightarrow \lambda x.\, M =_\beta \lambda x.\, N \qquad \text{(правило } \xi\text{)}
\end{array}
$$

Если $M =_\beta N$ доказуемо в λ-исчислении, пишут

$$\lambda \vdash M =_\beta N.$$

4.3 η-эквивалентность

Схема аксиом η-преобразования:

$$\lambda x.\, Mx =_\eta M, \text{ если } x \notin FV(M). \qquad \text{(правило } \eta)$$

Аналогично можем определить правила и совместимость, как для β.

Смысл этой эквивалентности в том, что поведение данных двух термов одинаково; для произвольного N будет верно

$$(\lambda x.\, M)N =_\beta MN.$$

Принцип экстенсиональности Две функции считаются **экстенсионально эквивалентными**, если они дают одинаковый результат при одинаковом входе:

$$\forall N\colon FN =_\beta GN.$$

Выберем $u \notin FV(F) \cup FV(G)$, теперь

$$Fy =_\beta Gy \Longrightarrow \lambda y.\, Fy =_\beta \lambda y.\, Gy \Longrightarrow F =_{\beta\eta} G$$

5 Кодирование булевых значений, кортежей в чистом бестиповом λ-исчислении.

5.1 Булевы значения

$$\begin{aligned}
\mathtt{tru} &\equiv \lambda tf.\, t \\
\mathtt{fls} &\equiv \lambda tf.\, f
\end{aligned}$$

5.2 Булевы операции

$$\begin{aligned}
\mathtt{if} &\equiv \lambda bxy.\, bxy \\
\mathtt{not} &\equiv \lambda b.\, b\,\mathtt{fls}\,\mathtt{tru} \\
\mathtt{and} &\equiv \lambda xy.\, xy\,\mathtt{fls} \\
\mathtt{or} &\equiv \lambda xy.\, x\,\mathtt{tru}\,y
\end{aligned}$$

5.3 Кортежи

Пара и стандартные операции:

$$\begin{aligned}
\mathtt{pair} &\equiv \lambda xyf.\, fxy \\
\mathtt{fst} &\equiv \lambda p.\, p\,\mathtt{tru} \\
\mathtt{snd} &\equiv \lambda p.\, p\,\mathtt{fls}
\end{aligned}$$

6 Кодирование чисел Чёрча в чистом бестиповом λ-исчислении.

$$\begin{aligned}
0 &\equiv \lambda sz.\, z \\
1 &\equiv \lambda sz.\, sz \\
2 &\equiv \lambda sz.\, s(sz) \\
3 &\equiv \lambda sz.\, s(s(sz)) \\
4 &\equiv \lambda sz.\, s(s(s(sz))) \\
&\ldots
\end{aligned}$$

Определим $F^n(X)$, где $n \in \mathbb{N}$, $F, Z \in \Lambda$:

$$\begin{aligned}
F^0(X) &\equiv X \\
F^{n+1}(X) &\equiv F(F^n(X))
\end{aligned}$$

Теперь число Чёрча принимает следующий вид

$$n \equiv \lambda sz.\ s^n(z).$$

Функция проверки на ноль:

$$\mathtt{iszero} \equiv \lambda n.\ n(\lambda x.\ \mathtt{fls})\,\mathtt{tru}.$$

Переход к следующему числу:

$$\mathtt{succ} \equiv \lambda nsz.\ s(nsz).$$

Сложение:

$$\mathtt{plus} \equiv \lambda mnsz.\ ms(nsz).$$

Умножение:

$$\begin{aligned} \mathtt{mult} &\equiv \lambda mn.\ m(\mathtt{plus}\, n)0 \\ \mathtt{mult} &\equiv \lambda mnsz.\ m(ns)x \end{aligned}$$

7 Теорема о неподвижной точке. Y-комбинатор.

7.1 Решение уравнений на термы

Например, хотим найти F такой, что $\forall M, N, L\colon \lambda \vdash FMNL = ML(NL)$.

$$\begin{aligned} FMNL &= ML(NL) \\ FMNL &= (\lambda l.\ Ml(Nl))L \\ FMN &= \lambda l.\ Ml(Nl) \\ FM &= \lambda n.\ \lambda l.\ Ml(nl) \\ F &= \lambda mnl.\ ml(nl) \end{aligned}$$

А что делать, если уравнение рекурсивное? Например, $FM = MF$.

7.2 Теоремы о неподвижной точке

Теорема 1. Для любого λ-терма F существует неподвижная точка:

$$\forall F \in \Lambda\ \exists X \in \Lambda\colon \lambda \vdash FX = X.$$

Proof. Пусть $W \equiv \lambda x.\ F(xx)$ и $X \equiv WW$. Тогда

$$X \equiv WW \equiv (\lambda x.\ F(xx))W = F(WW) \equiv FX.$$

□

Теорема 2. Существует комбинатор неподвижной точки $\mathbf{Y}$ такой, что

$$\forall F\colon F(\mathbf{Y}F) = \mathbf{Y}F.$$

Proof. Пусть $\mathbf{Y} = \lambda f.\ (\lambda x.\ f(xx))(\lambda x.\ f(xx))$. Тогда

$$\mathbf{Y}F \equiv (\lambda x.\ F(xx))(\lambda x.\ F(xx)) = F(\underbrace{(\lambda x.\ F(xx))(\lambda x.\ F(xx))}_{\mathbf{Y}F}) \equiv F(\mathbf{Y}F).$$

□

7.3 Рекурсия

Y-комбинатор позволяет ввести рекурсию в λ-исчисление. Например, можно задать факториал рекурсивно:

$$\mathtt{fac} = \lambda n.\, \mathtt{if}(\mathtt{iszero}\, n)1(\mathtt{mult}\, n(\mathtt{fac}(\mathtt{pred}\, n))).$$

Можно переписать:

$$\mathtt{fac} = \underbrace{(\lambda f n.\, \mathtt{if}(\mathtt{iszero}\, n)1(\mathtt{mult}\, n(f(\mathtt{pred}\, n))))}_{\mathtt{fac}'}\, \mathtt{fac}\,.$$

Тогда $\mathtt{fac}$ — неподвижная точка для $\mathtt{fac}'$, поэтому $\mathtt{fac} = \mathrm{Y}\,\mathtt{fac}'$.

8 Редексы. Одношаговая и многошаговая редукция. Нормальная форма. Редукционные графы.

Если мы хотим доказать равенство, то эффективным способом будет сокращение всех редексов:

$$\begin{aligned} \mathrm{KI} &\equiv (\lambda xy.\, x)\,(\lambda z.\, z) = \lambda yz.\, z \\ \mathrm{IIK}_* &\equiv (\lambda x.\, x)\mathrm{IK}_* = \mathrm{IK}_* = (\lambda x.\, x)(\lambda yz.\, z) = \lambda yz.\, z \end{aligned}$$

Но как доказать неравенство?

8.1 Редукция

Определение 7: Совместимое отношение

Бинарное отношение $\mathcal{R}$ над Λ называют **совместимым** (с операциями λ-исчисления), если для всех $M, N, Z \in \Lambda$:

$$M\ \mathcal{R}\ N \quad\Rightarrow\quad \begin{aligned} (ZM)\ \mathcal{R}\ (ZN), \\ (MZ)\ \mathcal{R}\ (NZ), \\ (\lambda x.\, M)\ \mathcal{R}\ (\lambda x.\, N) \end{aligned}$$

Определение 8

Совместимое отношение эквивалентности называют **отношением конгруэнтоности** над Λ.
Совместимое, рефлексивное и транзитивное отношение называют **отношением редукции** над Λ.

Определение 9

Бинарное отношение **β-редукции за один шаг** $\to_\beta$ над Λ:

$$\begin{aligned} (\lambda x.\, M)N &\longrightarrow_\beta && [x \mapsto N]M \\ M \longrightarrow_\beta N &\Longrightarrow && ZM \longrightarrow_\beta ZN \\ M \longrightarrow_\beta N &\Longrightarrow && MZ \longrightarrow_\beta NZ \\ M \longrightarrow_\beta N &\Longrightarrow && \lambda x.\, M \longrightarrow_\beta \lambda x.\, N \end{aligned}$$

Это правило и совместимость.

Определение 10: β-редукция

Бинарное отношение **β-редукции** $\twoheadrightarrow_\beta$ над Λ определяется индуктивно:

$$\begin{aligned} && M \twoheadrightarrow_\beta M && \text{(refl)} \\ M \longrightarrow_\beta N &\Rightarrow & M \twoheadrightarrow_\beta N && \text{(ini)} \\ M \twoheadrightarrow_\beta N,\ N \twoheadrightarrow_\beta L &\Rightarrow & M \twoheadrightarrow_\beta L && \text{(trans)} \end{aligned}$$

Это отношение — транзитивное, рефлексивное замыкание $\longrightarrow_\beta$, поэтому является отношением редукции.

Определение 11: Конвертируемость

Бинарное отношение **конвертируемости** $=_\beta$ над Λ определяется индуктивно:

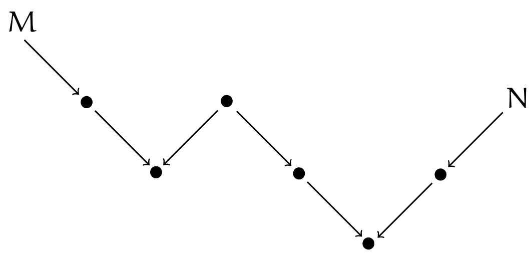
$$M \twoheadrightarrow_\beta N \Rightarrow M =_\beta N \qquad \text{(ini)}$$
$$M =_\beta N \Rightarrow N =_\beta M \qquad \text{(sym)}$$
$$M =_\beta N,\ N =_\beta L \Rightarrow M =_\beta L \qquad \text{(trans)}$$

Это отношение является отношением конгруэнтности.

Утверждение 1. $M =_\beta N \iff \lambda \vdash M = N$.

Proof. Индукция по количеству шагов, расписываем по определению. □

Два терма M, N связаны отношением $=_\beta$, если есть связывающая цепочка стрелок $\longrightarrow_\beta$:



Редукционный граф для терма $M \in \Lambda$ (нотация $G_\beta(M)$) — ориентированный мультиграф с вершинами в $\{N \mid M \twoheadrightarrow_\beta N\}$, ребро проводится между N и L, если $N \longrightarrow_\beta L$.

Отношение β-эквивалентности не является разрешимым в общем случае (то есть для пары термов не существует универсального алгоритма, проверяющего эквивалентность).

8.2 Нормальная форма

Это то, что лежит в самом низу на картинке.

Определение 12

λ-терм M *находится* в **β-нормальной форме** (β-NF), если в нем нет подтермов, являющихся β-редексами.
λ-терм M *имеет* β-нормальную форму, если для некоторого N выполняется $M =_\beta N$ и N находится в β-NF.

Пример 2.

- $\lambda xy.\, x(\lambda z.\, zx)y$ находится в β-NF
- $(\lambda x.\, xx)y$ не находится в β-NF, но имеет в качестве β-NFтерм yy
- Ω не имеет нормальной формы

9 Теорема Чёрча-Россера и ее следствия.

Теорема 3 (Черч, Россер). Если $M \twoheadrightarrow_\beta N$, $M \twoheadrightarrow_\beta K$, то существует такой L, что $N \twoheadrightarrow_\beta L$ и $K \twoheadrightarrow_\beta L$.

Иначе говоря, β-редукция обладает *свойством ромба* или *конфлюентностью*:

$$
\begin{tikzcd}
 & M \arrow[dl, two heads] \arrow[dr, two heads] & \\
N \arrow[dr, two heads, dotted] & & K \arrow[dl, two heads, dotted] \\
 & L &
\end{tikzcd}
$$

Следствие 1 (о существовании общего редукта). Если $M =_\beta N$, то существует L такой, что $M \twoheadrightarrow_\beta L$ и $N \twoheadrightarrow_\beta L$.

Proof. Индукция по генерации $=_\beta$. Разберем три случая вывода $M =_\beta N$:

$M \twoheadrightarrow_\beta N$. Возьмем $L = N$.

$N =_\beta M$. По предположению индукции есть общий β-редукт L_1 для N и M. Можем взять $L = L_1$.

$M =_\beta N'$, $N' =_\beta N$. Тогда

$$
\begin{tikzcd}
M \arrow[dr, two heads] & \text{(по ПИ)} & N' \arrow[dl, two heads] \arrow[dr, two heads] & \text{(по ПИ)} & N \arrow[dl, two heads] \\
 & L_1 \arrow[dr, two heads, dotted] & \text{(CR)} & L_2 \arrow[dl, two heads, dotted] & \\
 & & L & &
\end{tikzcd}
$$

□

Следствие 2 (о редуцируемости к β-NF). Если M имеет N в качестве β-NF, то $M \twoheadrightarrow_\beta N$.

Следствие 3 (о единственности β-NF). λ-терм имеет не более одной β-NF.

10 Стратегии редукции. Теорема о нормализации. Механизмы вызова в функциональных языках.

Рассмотрим три варианта терма:

- Переменная v, редукция завершена, ничего интересного
- Абстракция $\lambda x.\, M$, просто редуцируем M
- Аппликация MN. Разбираем аппликацию влево (в M) до не-аппликации. Пока получаем аппликацию, разделяем дальше. Два варианта остановки:
  - Переменная. Тогда мы получили такой вид
    $$(\ldots((vN_1)N_2)\ldots N_k).$$
    Теперь нам нужно просто проредуцировать каждый N_i, общая структура от этого не изменится.
  - Абстракция. Получили:
    $$(\ldots(((\lambda x.\, M')N_1)N_2)\ldots N_k).$$
    Будем работать дальше. Здесь есть две стратегии:
    * **Нормальная стратегия**: сразу сокращаем редекс $(\lambda x.\, M)N_1$.

* **Аппликативная стратегия**: сначала редуцируем отдельно все N_i слева направо до нормальной формы N_j', а уже потом сокращаем редекс $(\lambda x.\, M)N_1'$.
  Также можно редуцировать только N_1 и сразу сократить.

Разбор терма можно представить в виде дерева, где узлы @ задают аппликацию, а узлы λ — абстракцию.

Теорема 4. Лямбда-терм может иметь одну из двух форм:

$$\lambda \vec{x}.\, y\vec{N} \equiv \lambda x_1 x_2 \dots x_n.\, yN_1, N_2, \dots N_k, \quad n, k \geqslant 0 \qquad \text{(HNF)}$$
$$\lambda \vec{x}.\, (\lambda z.\, M)\vec{N} \equiv \lambda x_1 x_2 \dots x_n.\, (\lambda z.\, M)N_1, N_2, \dots N_k, \quad n \geqslant 0, k > 0$$

Определение 13

Головная нормальная форма — форма следующего вида:

$$\lambda \vec{x}.\, y\vec{N} \equiv \lambda x_1 x_2 \dots x_n.\, yN_1, N_2, \dots N_k, \quad n, k \geqslant 0. \qquad \text{(HNF)}$$

Слабая головная нормальная форма (WHNF) — это HFN или лямбда-абстракция (то есть не редекс на верхнем уровне).
Переменная y называется **головной переменной**. Редекс $(\lambda z.M)N_1$ называется **головным редексом**.

Переменная y может совпадать с одним из x_i.

В Haskell вычисления как раз форсируются до слабой нормальной формы.

Теорема 5 (о нормализации)**.** Если терм M имеет нормальную форму, то последовательное сокращение самого левого внешнего редекса приводит к этой нормальной форме.

То есть, если HNF есть, то нормальная стратегия гарантировано приводит к ней.

10.1 Механизмы вызова в ФЯ

Аппликативная стратегия более эффективная, хоть и не всегда приводит к HNF, поэтому применяется во многих языках программирования.

Пусть N — очень большой терм, который вычисляется сутки. Если мы запустим нормальную стратегию на

$$(\lambda x.\, Fx(Gx)x)N,$$

получим $FN(GN)N$ и N придется в дальнейших редукциях сократить трижды. Но для такого нормальная стратегия не вычислит N ни разу:

$$(\lambda xy.\, y)N \longrightarrow_\beta \lambda y.\, y.$$

Аппликативная стратегия будет вычислять N только один раз в каждом из примеров.

В „ленивых" языках программирования (Haskell, Clean) используется похожая на нормальную стратегия, а для решения проблем с эффективностью используют механизм *разделения*: вместо непосредственной подстановки терма подставляют указатель на терм, который хранится в памяти как *отложенное вычисление*.

Механизм вызова в функциональных языках:

* *вызов по значению* — аппликативый порядок редукций до WHNF
* *вызов по имени* — нормальный порядок редукций до WHNT
* *вызов по необходимости* — „вызов по имени" с разделением.

11 Функция предшествования для чисел Черча. Комбинатор примитивной рекурсии.

11.1 Функция предшествования

Введем вспомогательные функции:

$$\begin{aligned} \mathtt{zp} &\equiv \mathtt{pair}\,0\,0 \\ \mathtt{sp} &\equiv \lambda\, p.\, \mathtt{pair}(\mathtt{snd}\, p)(\mathtt{succ}(\mathtt{snd}\, p))) \end{aligned}$$

Вторая функция позволяет, приняв пару $(\_, j)$, вернуть пару $(j, j+1)$:

$$\mathtt{sp}(\mathtt{pair}\, i\, j) = \mathtt{pair}\, j\ (j+1).$$

Если взять композицию $m > 0$ раз от $\mathtt{zp}$:

$$\begin{aligned} \mathtt{sp}^0(\mathtt{zp}) &= \mathtt{pair}\,0\,0 \\ \mathtt{sp}^m(\mathtt{zp}) &= \mathtt{pair}(m-1)\ m \end{aligned}$$

Тогда можно определить функцию предшествования:

$$\mathtt{pred} = \lambda\, m.\, \mathtt{fst}(m\ \mathtt{sp}\, \mathtt{zp}).$$

Тогда

$$\mathtt{minus} = \lambda\, m\, n.\, n\ \mathtt{pred}\, m.$$

11.2 Комбинатор примитивной рекурсии

Обобщим конструкцию. Второй элемент пары оставим счетчиком, а в первом храним результат.

$$\begin{aligned} \mathtt{xz} &\equiv \lambda\, x.\, \mathtt{pair}\, x\, 0 \\ \mathtt{fs} &\equiv \lambda\, f\, p.\, \mathtt{pair}(f(\mathtt{fst}\, p)(\mathtt{snd}\, p))(\succ (\mathtt{snd}\, p)) \\ \mathtt{rec} &\equiv \lambda\, m\, f\, x.\, \mathtt{fst}(m(\mathtt{fs}\, f)(\mathtt{xz}\, x)) \end{aligned}$$

Теперь можно выразить $\mathtt{pred}$:

$$\mathtt{pred}' = \lambda\, n.\, \mathtt{rec}\, n\, (K_*)\, 0.$$

И факториал:

$$\mathtt{fac} = \lambda\, n.\, \mathtt{rec}\, n\ (\lambda\, x\, y.\, \mathtt{mult}\, x(\mathtt{succ}\, y))\ 1.$$

11.3 Списки

$$\begin{aligned} \mathtt{nil} &\equiv \lambda\, c\, n.\, n \\ \mathtt{cons} &\equiv \lambda\, e\, l\, c\, n.\, c\, e(l\, c\, n) \end{aligned}$$

Теперь

$$\begin{aligned} &[\,] = \mathtt{nil} \\ &[5,3,2] = \mathtt{cons}\,5(\mathtt{cons}\,3(\mathtt{cons}\,2\,\mathtt{nil})) = \lambda\, c\, n.\, c\,5(c\,3(c\,2\,n)) \end{aligned}$$

$$\mathtt{cons} \equiv \lambda\, l.\, l\,(\lambda\, h\, t.\, \mathtt{fls})\, \mathtt{tru}.$$

12 Просто типизированное λ-исчисление в стиле Карри. Предтермы. Утверждения о типизации. Контексты. Правила типизации.

12.1 Понятие типа

Система типов — гибко управляемый синтаксический метод доказательства отсутствия в программе определенных видов поведения при помощи классификации выражений языка по разновидностям вычисляемых ими значений (Бенджамин Пирс).

В λ-исчислении:

- выражения — λ-термы
- вычисления — их редукция
- значения — $(WH)NF$

Типы рассматриваются как *синтаксические* конструкции, приписываемые термам про определенным правилам:

$$M\colon \sigma.$$

12.2 Для чего нужны типы?

- Частичная спецификация:

$$f\colon \mathbb{N} \to \mathbb{N}, \qquad g\colon (\forall n\colon \mathbb{N}.\ \exists m\colon \mathbb{N}.\ m \leqslant n).$$

- Проверка типов помогает отлавливать простые ошибки

12.3 Тип функции

Стрелка — базовый способ конструирования типа, задающая функциональный тип. Например, для функции $I \equiv \lambda x.\ x$ может быть приписан тип:

$$I\colon \alpha \to \alpha.$$

В общем случае типом функции из α в β будет $\alpha \to \beta$.

Если $y\colon \alpha$ является аргументом функции $f\colon \alpha \to \beta$, то возвращаемое значение $f\,y$ имеет тип β.

Гипотеза о типе переменных записывают в контексте:

$$y\colon a \vdash (Iy)\colon \alpha.$$

12.4 Системы в стиле Карри

Оставляем термы такими же, что и в бестиповой теории. Каждый терм может обладать множеством различных типов (пустое, одно-, многоэлементное, бесконечное).

12.5 Просто типизированное λ-исчисление

Определение 14

Множество типов $\mathbb{T}$ системы $\lambda_\to$ определяется индуктивно:

$$\begin{aligned} &\alpha, \beta \ldots \in \mathbb{T} && \text{(переменные типа)}\\ &\sigma, \tau \in \mathbb{T} \Rightarrow (\sigma \to \tau) \in \mathbb{T} && \text{(типы пространства функций)} \end{aligned}$$

В абстрактном синтаксисе: $\mathbb{T} ::= \mathbb{V} \mid (\mathbb{T} \to \mathbb{T})$, где $\mathbb{V} = \{\alpha, \beta \ldots\}$.

Стрелка *правоассоциативна*: если $\sigma_1, \dots \sigma_n \in \mathbb{T}$, то

$$\sigma_1 \to \sigma_2 \to \dots \to \sigma_n \equiv (\sigma_1 \to (\sigma_2 \to \dots \to (\sigma_{n-1} \to \sigma_n) \dots)).$$

Всякий тип может быть записан в виде:

$$\sigma_1 \to \dots \to \sigma_n \to \alpha.$$

Определение 15: Предтермы

Множество **предтермов** Λ строится из переменных из $V = \{x, y, z, \dots\}$ с помощью аппликации и абстракции:

$$\begin{aligned} x \in V &\implies & x \in \Lambda \\ M, N \in \Lambda &\implies & (MN) \in \Lambda \\ M \in \Lambda, x \in V &\implies & (\lambda x.\, M) \in \Lambda \end{aligned}$$

В абстрактном стиле: $\Lambda ::= V \mid (\Lambda\Lambda) \mid (\lambda V.\, \Lambda)$.

Предтермы системы в стиле Карри — термы бестипового λ-исчисления.

Определение 16

Утверждение о типизации $\lambda_{\to}$ по Карри имеет вид $M\colon \tau$, где субъект $M \in \Lambda$ и предикат $\tau \in \mathbb{T}$.

Пример 3. Примеры утверждений типизации

$$\begin{aligned} &\lambda x.\, x\colon \alpha \to \alpha \\ &\lambda x.\, x\colon (\alpha \to \beta) \to \alpha \to \beta \\ &\lambda x y.\, x\colon \alpha \to \beta \to \alpha \end{aligned}$$

Определение 17: Объвление

Объявление — утверждение типизации с термовой переменной в кресле субъекта.

Пример 4. Примеры объявлений

$$\begin{aligned} &x\colon \alpha \\ &y\colon \beta \\ &f\colon (\alpha \to \beta) \to \gamma \end{aligned}$$

Определение 18: Контекст

Контекст — множество объявлений с *различными* переменными в качестве субъекта:

$$\Gamma = \{x_1\colon \sigma_1,\ x_2\colon \sigma_2, \dots,\ x_n\colon \sigma_n\}.$$

Контекст иногда называют *базисом* или *окружением*.

Фигурные скобки иногда опускают:

$$\Gamma = x\colon \alpha,\ f\colon \alpha \to \beta,\ g\colon (\alpha \to \beta) \to \gamma \equiv x^{\alpha},\ f^{\alpha \to \beta},\ g^{(\alpha \to \beta) \to \gamma}.$$

Контексты можно расширять, добавляя объявление новой переменной.

Также контексты можно рассматривать как частичные функции из V в множество типов $\mathbb{T}$.

12.6 Правила типизации

Определение 19

Утверждение $M\colon \tau$ называется **выводимым** в контексте Γ, обозначается

$$\Gamma \vdash M\colon \tau,$$

если его вывод может быть произведен по правилам:

$$\begin{array}{rcr} x^\sigma \in \Gamma & \Longrightarrow & \Gamma \vdash x\colon \sigma \\ \Gamma \vdash M\colon \sigma \to \tau,\ \Gamma \vdash N\colon \sigma & \Longrightarrow & \Gamma \vdash MN\colon \tau \\ \Gamma,\ x^\sigma \vdash M\colon \tau & \Longrightarrow & \Gamma \vdash \lambda x.\, M\colon \sigma \to \tau \end{array}$$

Если существуют Γ и τ такие, что $\Gamma \vdash M\colon \tau$, то предтерм M называют **(допустимым) термом**.

Запись в виде дерева вывода:

$$\Gamma \vdash x\colon \sigma,\ \text{если } x^\alpha \in \Gamma \qquad \text{(аксиома)}$$

$$\frac{\Gamma \vdash M\colon \sigma \to \tau \qquad \Gamma \vdash N\colon \sigma}{\Gamma \vdash MN\colon \tau} \qquad (\to \mathrm{Elim})$$

$$\frac{\Gamma,\ x^\alpha \vdash M\colon \tau}{\Gamma \vdash \lambda x.\, M\colon \sigma \to \tau} \qquad (\to \mathrm{Intro})$$

13 Просто типизированное λ-исчисление в стиле Черча. Предтермы. Утверждения о типизации. Контексты. Правила типизации.

13.1 Системы в стиле Черча

Термы — аннотированные версии бестиповых термов. Каждый терм имеет обычно уникальный тип, выводимый из способа, которым терм аннотирован.

Определение 20: Предтермы

Множество **предтермов** $\Lambda_{\mathbb{T}}$ строится из переменных из $V = \{x, y, z, \ldots\}$ с помощью аппликации и *аннотированной типами* абстракции:

$$\begin{array}{rcr} x \in V & \Longrightarrow & x \in \Lambda_{\mathbb{T}} \\ M, N \in \Lambda_{\mathbb{T}} & \Longrightarrow & (MN) \in \Lambda_{\mathbb{T}} \\ M \in \Lambda_{\mathbb{T}}, x \in V, \sigma \in \mathbb{T} & \Longrightarrow & (\lambda x^\sigma.\, M) \in \Lambda_{\mathbb{T}} \end{array}$$

В абстрактном стиле: $\Lambda_{\mathbb{T}} ::= V \mid (\Lambda_{\mathbb{T}}\Lambda_{\mathbb{T}}) \mid (\lambda V^{\mathbb{T}}.\, \Lambda_{\mathbb{T}})$.

Определение 21

Утверждение о типизации $\lambda_\to$ по Черчу имеет вид $M\colon \tau$, где субъект $M \in \Lambda^{\mathbb{T}}$ и предикат $\tau \in \mathbb{T}$.

Пример 5. Примеры утверждений типизации

$$\begin{aligned} &\lambda x^\alpha.\, x\colon \alpha \to \alpha \\ &\lambda x^{\alpha\to\beta}.\, x\colon (\alpha \to \beta) \to \alpha \to \beta \\ &\lambda x^\alpha y^\beta.\, x\colon \alpha \to \beta \to \alpha \end{aligned}$$

13.2 Правила типизации

Определение 22

Утверждение $M\colon \tau$ называется **выводимым** в контексте Γ, обозначается

$$\Gamma \vdash M\colon \tau,$$

если его вывод может быть произведен по правилам:

$$\begin{aligned} x^\sigma \in \Gamma &\implies & \Gamma \vdash x\colon \sigma \\ \Gamma \vdash M\colon \sigma \to \tau,\ \Gamma \vdash N\colon \sigma &\implies & \Gamma \vdash MN\colon \tau \\ \Gamma,\ x^\sigma \vdash M\colon \tau &\implies & \Gamma \vdash \lambda x.\, M\colon \sigma \to \tau \end{aligned}$$

Если существуют Γ и τ такие, что $\Gamma \vdash M\colon \tau$, то предтерм M называют **(допустимым) термом**.

Запись в виде дерева вывода:

$$\Gamma \vdash x\colon \sigma,\ \text{если } x^\alpha \in \Gamma \qquad \text{(аксиома)}$$

$$\frac{\Gamma \vdash M\colon \sigma \to \tau \qquad \Gamma \vdash N\colon \sigma}{\Gamma \vdash MN\colon \tau} \qquad (\to \mathrm{Elim})$$

$$\frac{\Gamma,\ x^\alpha \vdash M\colon \tau}{\Gamma \vdash \lambda x^\alpha.\, M\colon \sigma \to \tau} \qquad (\to \mathrm{Intro})$$

14 Свойства систем просто типизированного λ-исчисления

Лемма 2 (об инверсии или о генерации).

- $\Gamma \vdash x\colon \sigma \implies x^\sigma \in \Gamma$
- $\Gamma \vdash \lambda x.\, M\colon \tau \implies \exists \sigma\colon [\Gamma \vdash M\colon \sigma \to \tau \wedge \Gamma \vdash N\colon \sigma]$
- Карри: $\Gamma \vdash \lambda x.\, M\colon \rho \implies \exists \sigma, \tau\colon [\Gamma, x^\sigma \vdash M\colon \tau \wedge \rho \equiv \sigma \to \tau]$
- Черч: $[\Gamma \vdash \lambda x^\sigma.\, M\colon \rho \implies \exists \tau\colon [\Gamma, x^\sigma \vdash M\colon \tau \wedge \rho \equiv \sigma \to \tau]$

Лемма 3 (о типизируемости подтерма). Пусть M' — подтерм M. Тогда для некоторых Γ' и σ'

$$\Gamma \vdash M\colon \sigma \implies \Gamma' \vdash M'\colon \sigma'.$$

То есть, если терм имеет тип, что и подтерм имеет тип.

14.1 Леммы о контекстах

Лемма 4 („разбавления”). Пусть Γ и Δ — контексты, причем $\Gamma \subset \Delta$. Тогда

$$\Gamma \vdash M\colon \sigma \implies \Delta \vdash M\colon \sigma.$$

Расширение контекста не влияет на выводимость утверждения типизации.

Лемма 5 (о свободных переменных). Свободные переменные типизированного терма должны присутствовать в контексте:

$$\Gamma \vdash M\colon \sigma \implies FV(M) \subset \mathtt{dom}(\Gamma).$$

Лемма 6 (сужения). Сужение контекста до множества свободных переменных терма не влияет на выводимость утверждения типизации:

$$\Gamma \vdash M\colon \sigma \implies \Gamma{\restriction}_{FV(M)} \vdash M\colon \sigma.$$

Вместе эти леммы отвечают на вопрос: какой контекст требуется, чтобы произвести присваивание типов? Ответ следующий: в контексте обязательно должны присутствовать свободные переменные типизируемого терма; все остальные переменные опциональны, не влияют на типизацию и могут быть безболезненно отброшены по лемме сужения.

Рассмотрим предтерм xx. Пусть это терм. Тогда есть Γ и τ:

$$\Gamma \vdash xx\colon \tau.$$

По лемме об инверсии существует σ, что правый подтерм $x\colon \sigma$, левый подтерм (тоже x) имеет тип $\sigma \to \tau$.

По лемме о контекстах $x \in \mathrm{dom}(\Gamma)$ и должен иметь там единственное связывание по определению контекста. То есть $\sigma = \sigma \to \tau$ — получили, что тип является подвыражением себя, чего не может быть, так как типы конечны.

Тогда ω, Ω и Yне имеют типа по лемме о типизируемости подтерма.

Определение 23

Для $\sigma, \tau \in \mathbb{T}$ **подстановку** τ вместо α в σ обозначим $[\alpha \mapsto \tau]\sigma$.

Пример 6.
$$[\alpha \mapsto (\gamma \to \gamma)]\alpha \to \beta \to \alpha = (\gamma \to \gamma) \to \beta \to \gamma \to \gamma.$$

Лемма 7 (подстановки типа)**.** Подстановка в выводимое утверждение типизации некоторого типа вместо переменной типа порождает выводимое утверждение типизации.

Карри:
$$\Gamma \vdash M\colon \sigma \Longrightarrow [\alpha \mapsto \tau]\Gamma \vdash M\colon [\alpha \mapsto \tau]\sigma.$$

Черч:
$$\Gamma \vdash M\colon \sigma \Longrightarrow [\alpha \mapsto \tau]\Gamma \vdash [\alpha \mapsto \tau]M\colon [\alpha \mapsto \tau]\sigma.$$

Пример 7. Подстановка $[\alpha \mapsto (\gamma \to \gamma)]$ в утверждение типизации для системы Черча

$$x^\alpha \vdash \lambda y^\alpha z^\beta.\, x\colon \alpha \to \beta \to \alpha$$

осуществляется и в тип, и в терм, и в контекст и дает новое утверждение типизации

$$x^{\gamma \to \gamma} \vdash (\lambda y^{\gamma \to \gamma} z^\beta.\, x)\colon (\gamma \to \gamma) \to \beta \to \gamma \to \gamma.$$

Поскольку первое утверждение типизации выводимо, то и второе тоже выводимо.

Лемма 8 (подстановке терма)**.** Пусть $\Gamma,\ x^\sigma \vdash M\colon \tau$ и $\Gamma \vdash N\colon \sigma$, тогда

$$\Gamma \vdash [x \mapsto N]M\colon \tau.$$

Подходящая по типу подстановка терма сохраняет тип.

Пример 8. Берем выводимое утверждение типизации

$$x^{\gamma \to \gamma} \vdash \lambda y^\beta.\, x\colon \beta \to \gamma \to \gamma.$$

и подставляем в него вместо свободной переменной x типа $\gamma \to \gamma$ терм $\lambda z^\gamma.\, z$ подходящего типа $\gamma \to \gamma$. Получаем
$$\vdash \lambda y^\beta z^\gamma.\, z\colon \beta \to \gamma \to \gamma.$$

Теорема 6 (о редукции субъекта)**.** Пусть $M \twoheadrightarrow_\beta N$, тогда

$$\Gamma \vdash M\colon \sigma \Longrightarrow \Gamma \vdash N\colon \sigma.$$

Тип терма сохраняется про β-редукциях.

Proof. Следует из леммы о подстановке терма. □

Следствие 4. Множество типизируемых в $\lambda_{\to}$ термов замкнуто относительно редукции.

В обратную сторону теорема и следствие не верны для $\lambda_{\to}$.

Теорема 7 (о единственности типа для $\lambda_{\to}$ по Черчу)**.** Пусть $\Gamma \vdash M : \sigma$ и $\Gamma \vdash M : \tau$. Тогда $\sigma \equiv \tau$.
Терм в $\lambda_{\to}$ по Черчу имеет единственный тип.

Следствие 5. Пусть $\Gamma \vdash M : \sigma$, $\Gamma \vdash N : \tau$ и $M =_\beta N$. Тогда $\sigma \equiv \tau$.
Типизируемые β-конвертируемые термы имеют одинаковый тип в $\lambda_{\to}$ по Черчу.

Пример 9 (Контрпример для системы по Карри)**.** Следующие два типа подходят для $K = \lambda x y.\, x$ по Карри:

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

15 Связь между системами Карри и Черча. Проблемы разрешимости. Сильная и слабая нормализация.

Зададим для Черча *стирающее отображение* $|\cdot| : \Lambda_{\mathbb{T}} \to \Lambda$:

$$\begin{aligned} |x| &\equiv x \\ |MN| &\equiv |M||N| \\ |\lambda x^\sigma.\, M| &\equiv \lambda x.\, |M| \end{aligned}$$

Все атрибутированные типами термы из версии по Черчу „проектируются” в термы в версии по Карри:

$$M \in \Lambda_{\mathbb{T}} \wedge \Gamma \vdash M : \sigma \Longrightarrow \Gamma \vdash |M| : \sigma.$$

Также можно „поднять” в обратную строну из Карри в Черча, правильно подобрав типы:

$$M \in \Lambda \wedge \Gamma \vdash M : \sigma \Longrightarrow \exists N \in \Lambda_{\mathbb{T}} : [\Gamma \vdash N : \sigma \wedge |N| \equiv |M|].$$

Для произвольного типа $\sigma \in \mathbb{T}$ обитаемость в версии по Карри равносильна обитаемости в версии по Черчу.

15.1 Проблемы разрешимости

Для любой системы типов важную роль имеют проблемы разрешимости основных задач: есть ли алгоритм, решающий данную задачу? Для $\lambda_{\to}$ все эти задачи разрешимы. Первые две задачи будут

$\vdash M : \sigma?$	Задача проверки типа	ЗПТ, TCP
$\vdash M : ?$	Задача синтеза типа	ЗСТ, TSP / TAP
$\vdash ? : \sigma$	Задача обитаемости типа	ЗОТ, TIP

решены с помощью теоремы Хиндли-Миллера.

Определение 24

Терм называется **слабо (weak) нормализуемым** (WN), если существует последовательность редукций, приводящих его к нормальной форме.
Терм называется **сильно (strong) нормализуемым** (SN), если любая последовательность редукций, приводящих его к нормальной форме.

Пример 10. KIK сильно нормализуем, $KI\Omega$ слабо нормализуем, Ω не нормализуем вообще.

Определение 25

Систему типов называют **слабо нормализуемой**, если все ее допустимые термы слабо нормализуемы. Систему типов называют **сильно нормализуемой**, если все ее допустимые термы сильно нормализуемы.

Теорема 8 (о нормализации $\lambda_{\to}$). Обе системы (по Карри и по Черчу) сильно нормализуемы.

То есть любой допустимый терм в $\lambda_{\to}$ всегда редуцируется к нормальной форме независимо от выбранной стратегии редукции.

16 Понятие главного (наиболее общего) типа. Подстановка типа и их композиция.

Для систем Карри и Черча верна *лемма подстановки типа*:

$$\Gamma \vdash M : \sigma \Longrightarrow [\alpha \mapsto \tau]\Gamma \vdash [\alpha \mapsto \tau]M : [\alpha \mapsto \tau]\sigma.$$

В версии Черча $\lambda_{\to}$ термы атрибутированы типами, поэтому тип терма единственен:

$$\lambda f^{\sigma\to\tau\to\rho} g^{\sigma\to\tau} z^{\sigma}.\, f z(g z) : (\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho$$
$$\lambda f^{\sigma\to\tau\to\sigma} g^{\sigma\to\tau} z^{\sigma}.\, f z(g z) : (\sigma \to \tau \to \sigma) \to (\sigma \to \tau) \to \sigma \to \sigma$$
$$\lambda f^{(\tau\to\rho)\to\tau\to\rho} g^{(\tau\to\rho)\to\tau} z^{(\tau\to\rho)}.\, f z(g z) : ((\tau \to \rho) \to \tau \to \rho) \to ((\tau \to \rho) \to \tau) \to (\tau \to \rho) \to \rho$$

Любой из этих типов можно приписать терму $S = \lambda f g z.\, f z(g z)$ в версии Карри.

Однако, первый „лучше" в том смысле, что остальные получаются из него подстановками типа вместо типовых переменных. Он называется **главным (principle)**. Также он отличается тем, что в нем меньше всего стрелок и максимальное разнообразие типов.

16.1 Вывод главного типа

Пусть у нас есть комбинатор

$$S = \lambda x y.\, y(\lambda z.\, y x).$$

Припишем всем термовым переменным типовую метапеременную:

$$\lambda x^{\alpha} y^{\beta}.\, y^{\beta}(\lambda z^{\gamma}.\, y^{\beta} x^{\alpha}).$$

Припишем типовую переменную метапеременную всем аппликативным подтермам: $(y\,x)' : \delta$, $y(\lambda z.y\,z) : \varepsilon$

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

Теперь можем связать некоторые метапеременные уравнениями, необходимыми для типизируемости терма:

$$\beta \sim \alpha \to \delta, \quad \beta \sim (\gamma \to \delta) \to \varepsilon.$$

Найдем главный унификатор для типовых переменных (подстановку), дающий решения уравнений:

$$\alpha \coloneqq \gamma \to \delta,\ \beta \coloneqq (\gamma \to \delta) \to \varepsilon,\ \delta \coloneqq \varepsilon.$$

Главный тип

$$\lambda x y.\, y(\lambda z.\, y z) : (\gamma \to \varepsilon) \to ((\gamma \to \varepsilon) \to \varepsilon) \to \varepsilon.$$

Определение 26: Подстановка типа

Подстановка типа — операция $S\colon \mathbb{T} \to \mathbb{T}$ такая, что

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

Можно представлять типы как вершины бинарного дерева, пара ребер соответствует подстановке. Обычно подстановка тождественна на всех типовых переменных, кроме конечного носителя

$$\sup(S) = \{\alpha \mid S(\alpha) \not\equiv \alpha\}.$$

Пример подстановки:

$$S = [a :- \gamma \to \beta, \beta := \alpha \to \gamma].$$

Тождественная подстановка (то есть с пустым носителем) обозначают $[]$.

Все подстановки происходят *параллельно*: для $\tau = \alpha \to \beta \to \gamma$:

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

Определение 27: Композиция подстановок

Композиция двух подстановок — подстановка с носителем, являющимся объединением носителей, над которым последовательно выполнены обе подстановки.

Подстановки образуют моноид относительно композиции и с $[]$ в роли нейтрального элемента.

Определение 28: Унификатор

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

Пример 11. Пусть $\sigma = \alpha \to \beta \to \gamma$, $\tau = (\delta \to \varepsilon) \to \phi$. Унификатор:

$$S = [\alpha := \delta \to \varepsilon, \varphi := \beta \to \gamma].$$

В результате такой подстановки

$$S(\sigma) \equiv S(\tau) \equiv (\delta \to \varepsilon) \to (\beta \to \gamma).$$

Когда выделяем одну элементарную подстановку нужно сразу выполнить ее повсюду.

Определение 29: Главный унификатор

Унификатор S — **главный унификатор** для σ и τ, если для любого другого унификатора S' существует подстановка T такая, что

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

17 Алгоритм унификации

Теорема 9 (Робинсон, 1965). Существует алгоритм унификации U, который для заданных типов σ и τ возвращает:

- главный унификатор S для σ и τ, если σ и τ могут быть унифицированы;
- сообщение об ошибке в противном случае.

Алгоритм $U(\sigma, \tau)$ позволяет искать минимальное решение уравнения на типы $\sigma \sim \tau$.

Ключевой момент всех рассуждений про унификацию:

$$\sigma_1 \to \sigma_2 \equiv \tau_1 \to \tau_2 \iff \sigma_1 \equiv \tau_1 \wedge \sigma_2 == \tau_2.$$

17.1 Алгоритм

Опишем алгоритм:

$$\begin{array}{lcl} U(\alpha, \alpha) & = & [] \\ U(\alpha, \tau) \mid \alpha \in FV(\tau) & = & error \\ U(\alpha, \tau) \mid \alpha \notin FV(\tau) & = & [\alpha := \tau] \\ U(\sigma_1 \to \sigma_2, \alpha) & = & U(\alpha, \sigma_1, \sigma_2) \\ U(\sigma_1 \to \sigma_2, \tau_1 \to \tau_2) & = & U(U_2\sigma_1, U_2\sigma_1) \circ U_2 \quad \text{where } U_2 = U(\sigma_2, \sigma_1) \end{array}$$

- $U(\sigma, \tau)$ завершается, так как деревья типа конечны и количество типовых переменных сокращается на 1 через конечное число шагов: интересен только последний шаг (остальные сразу точно завершаются), но мы дойдем до переменных, так как увеличение происходит из-за обход по поддереву, а его высота каждый раз уменьшается, с некоторого момента, общая высота станет уменьшаться; когда мы дошли до переменной, их количество уменьшиться.
- $U(\sigma, \tau)$ унифицирует: по индукции, если S унифицирует (σ, τ), то $S \circ [\alpha := \rho]$ унифицирует $(\sigma \to \alpha, \tau \to \rho)$.
- $U(\sigma, \tau)$ выдает главный унификатор.

18 Алгоритм построения системы ограничений.

Наша первая цель — построить систему ограничений на типы для терма M (возможно незамкнутого).

Для типизации таких термов необходим контекст Γ, в котором объявляются типы всех свободных переменных.

Для подстановки S, унифицирующей систему уравнений на типы

$$E = \{\sigma_1 \sim \tau_1, \ldots \sigma_n \sim \tau_n\},$$

введем обозначение $S \models E$.

Теорема 10 (Теорема о существовании системы ограничений)**.** Для любых терма $M \in \Lambda$, контекста Γ ($FV(M) \subseteq \mathtt{dom}(\Gamma)$) и типа $\sigma \in \mathbb{T}$ существует конечное множество уравнений на типы $E = E(\Gamma, M, \sigma)$, такое что для некоторой подстановки S:

- $S \models E(\Gamma, M, \sigma) \Longrightarrow S(\Gamma) \vdash M \colon S(\sigma)$
- $S(\Gamma) \vdash M \colon S(\sigma) \Longrightarrow S' \models E(\Gamma, M, \sigma)$, для некоторого S', имеющего тот же эффект, что и S, на типовых переменных в Γ и σ.

18.1 Алгоритм

$$\begin{array}{lcl} E(\Gamma, x, \sigma) & = & \{\sigma \Gamma(x)\} \\ E(\Gamma, MN, \sigma) & = & E(\Gamma, M, \alpha \to \sigma) \\ E(\Gamma, \lambda x.\, M, \sigma) & = & E(\Gamma \cup \{x \colon \alpha\}, M, \beta) \cup \{\alpha \to \beta\ \sigma\} \end{array}$$

В первом равенстве контекст Γ рассматривается как функция из множества переменных в множество типов.

Переменные α во втором и третьем равенствах и β в третьем всякий раз должны быть „свежими”!

19 Главная пара и главный тип. Теорема Хиндли-Миллера.

Определение 30: Главная пара

Главная пара для $M \in \Lambda$ — пара (Γ, σ) такая, что

- $\Gamma \vdash M \colon \sigma$
- $\Gamma' \vdash M \colon \sigma' \Longrightarrow \exists S \colon S(\Gamma) \subseteq \Gamma' \land S(\sigma) \equiv \sigma'$

Пример 12. Для $M = \lambda x.\, x\, y$ имеем

$$PP(M) = (y \colon \alpha, (\alpha \to \beta) \to \beta$$
$$y \colon \alpha \vdash (\lambda x.\, x\, y) \colon (\alpha \to \beta) \to \beta$$

Теорема 11 (Теорема Хиндли – Милнера). Существует алгоритм PP, возвращающий для $M \in \Lambda$ главную пару (Γ, σ), если M имеет тип и сообщение об ошибке в противном случае.

Пусть

$$FV(M) = \{x_1, \dots, x_n\}$$

$$\Gamma_0 = x_1 \colon \alpha_1, \dots, x_n \colon \alpha_n, \quad \sigma_0 = \beta.$$

19.1 Алгоритм PP

$$PP(M) \mid U(E(\Gamma_0, M, \sigma_0)) \equiv error = errorPP(M) \mid U(E(\Gamma_0, M, \sigma_0)) \equiv S = (S(\Gamma_0), S(\sigma_0))$$

Стартуем с произвольных переменных типа, приписанных свободным переменным типизируемого терма M и всему терму.

19.2 Главный тип (Principle Type)

Определение 31: Главный тип

Для $M \in \Lambda_0$ **главным типом** называют тип σ, такой что

- $M \colon \sigma$
- $M \colon \sigma' \Longrightarrow \exists S \colon S(\sigma) \equiv \sigma'$

Следствие 6 (Следствие теоремы Хиндли – Милнера). Существует алгоритм PT , возвращающий для $M \in \Lambda_0$ главный тип σ, если M имеет тип и сообщение об ошибке в противном случае.