Системы типизации лямбда-исчисления

Введение

Денис Москвин

27.02.2011

CS Club при ПОМИ РАН

Что такое типы?

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

Бенджамин Пирс

В рамках курса: программы — λ-термы, вычисление — их редукция.

Типы — синтаксические конструкции, приписываемые термам по определённым правилам:

$$M : \sigma$$

Для чего нужны типы?

- Типы дают частичную спецификацию.

$$\mathrm{f}{:}\mathbb{N}\to\mathbb{N} \qquad\qquad \mathrm{g}{:}(\Pi\mathfrak{n}{:}\mathbb{N}.\,\exists\mathfrak{m}{:}\mathbb{N}.\,\mathfrak{m}>\mathfrak{n})$$

- Правильно типизированные программы не могут «сломаться». Робин Милнер (1978)

$$M{:}\mathbb{N}\ \wedge\ M\twoheadrightarrow\nu\ \Rightarrow\ \nu{:}\mathbb{N}$$

- Типизированные программы всегда завершаются (это не так :)

- Проверка типов отлавливает простые ошибки.

Основная литература (1)

LCWT

Henk Barendregt, Lambda calculi with types,
Handbook of logic in computer science (vol. 2), Oxford University Press, 1993

`ftp://ftp.cs.ru.nl/pub/CompMath.Found/HBK.ps`

Основная литература (2)

TAPL

Benjamin C. Pierce, Types and Programming Languages, MIT Press, 2002

`http://www.cis.upenn.edu/~bcpierce/tapl`

русский перевод:

Бенджамин Пирс, Типы в языках программирования, 2011

`http://newstar.rinet.ru/~goga/tapl/`

Основная литература (3)

ITT

Herman Geuvers, Introduction to Type Theory
Alfa Lernet Summer school 2008, Uruguay

`http://www.cs.ru.nl/H.Geuvers/Uruguay2008SummerSchool.html`

Дополнительная литература (1)

ATTAPL

Benjamin C. Pierce, editor. Advanced Topics in Types and Programming Languages, MIT, 2005

I2FP

John Harrison, Introduction to Functional Programming

`http://www.cl.cam.ac.uk/teaching/Lectures/funprog-jrh-1996`

русский перевод:

`http://code.google.com/p/funprog-ru/`

Дополнительная литература (2)

ЛИСС

Х. Барендрегт, Ламбда-исчисление, его синтаксис и семантика, М:Мир, 1985

H. P. Barendregt, The Lambda calculus. Its Syntax and Semantics. NHPC, 1981

ОЯП

Дж. Митчелл, Основания языков программирования, М.-Ижевск, НИЦ РХД, 2010

John C. Mitchell, Foundations for Programming Languages, MIT Press, 1996

Системы типизации лямбда-исчисления

Лекция 1. Система λ-исчисления без типов

Денис Москвин

27.02.2011

CS Club при ПОМИ РАН

Неформальное введение (1)

В λ-исчислении две операции: применение и абстракция.

Применение (Application):

$$\mathrm{F}\,\mathrm{X}$$

Програмистский взгляд:
F (алгоритм) применяется к X (входные данные).

Допустимо самоприменение $\mathrm{F}\,\mathrm{F}$.

Неформальное введение (2)

Абстракция (Abstraction):
Пусть $M \equiv M[x]$ — выражение, содержащее x. Тогда

$$\lambda x.\, M$$

обозначает функцию

$$x \mapsto M[x],$$

то есть каждому x сопоставляется $M[x]$.

Если x в $M[x]$ отсутствует, то $\lambda x.\, M$ — константная функция со значением M.

Неформальное введение (3)

Применение и абстракция работают совместно:

$$\underbrace{(\lambda x.\, 2 \times x + 1)}_{F}\ \underbrace{42}_{X} \;=\; 2 \times 42 + 1 \quad (= 85).$$

То есть $(\lambda x.\, 2 \times x + 1)\ 42$ — применение функции $x \mapsto 2 \times x + 1$ к аргументу 42, дающее в результате $2 \times 42 + 1$.

В общем случае имеем β-преобразование

$$(\lambda x.\, M)\, N = M[x := N],$$

где $M[x := N]$ обозначает подстановку N вместо x в M.

Термы (1)

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

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

В абстрактном синтаксисе

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

Соглашение. Произвольные термы пишем заглавными буквами, переменные — строчными.

Термы (2)

Примеры λ-термов:

$$x$$

$$(x\,z)$$

$$(\lambda x.\,(x\,z))$$

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

$$((\lambda y.\,((\lambda x.\,(x\,z))\,y))\,w)$$

$$(\lambda z.\,(\lambda w.\,((\lambda y.\,((\lambda x.\,(x\,z))\,y))\,w)))$$

Термы (3)

Соглашения:

- Внешние скобки опускаются.

- Применение ассоциативно *влево*:

$$F\,X\,Y\,Z \quad \text{обозначает} \quad (((F\,X)\,Y)\,Z)$$

- Абстракция ассоциативна *вправо*:

$$\lambda x\,y\,z.\,M \quad \text{обозначает} \quad (\lambda x.\,(\lambda y.\,(\lambda z.\,(M))))$$

Термы (4)

Те же примеры, с использованием соглашений

$$
\begin{aligned}
x &\equiv x \\
(x\,z) &\equiv x\,z \\
(\lambda x.\,(x\,z)) &\equiv \lambda x.\,x\,z \\
((\lambda x.\,(x\,z))\,y) &\equiv (\lambda x.\,x\,z)\,y \\
((\lambda y.\,((\lambda x.\,(x\,z))\,y))\,w) &\equiv (\lambda y.\,(\lambda x\,.\,x\,z)\,y)\,w \\
(\lambda z.\,(\lambda w.\,((\lambda y.\,((\lambda x.\,(x\,z))\,y))\,w))) &\equiv \lambda z\,w.\,(\lambda y.\,(\lambda x\,.\,x\,z)\,y)\,w
\end{aligned}
$$

Свободные и связанные переменные (1)

Абстракция $\lambda x\,.\,M[x]$ *связывает* дотоле *свободную* переменную x в терме M.

Примеры:

$$(\lambda y.\,(\lambda x.\,x\,z)\,y)\,w$$

Переменные x и y — связанные, а z и w — свободные.

$$(\lambda x.\,(\lambda x.\,x\,z)\,x)\,x$$

Переменная x — связанная (дважды!) и свободная, а z — свободная.

Свободные и связанные переменные (2)

Множество $\mathrm{FV}(T)$ ***свободных (free) переменных*** в λ-терме T определяется индуктивно:

$$\begin{aligned} \mathrm{FV}(x) &= \{x\}; \\ \mathrm{FV}(M\,N) &= \mathrm{FV}(M) \cup \mathrm{FV}(N); \\ \mathrm{FV}(\lambda x.\, M) &= \mathrm{FV}(M) \setminus \{x\}. \end{aligned}$$

Множество $\mathrm{BV}(T)$ ***связанных (bound) переменных***:

$$\begin{aligned} \mathrm{BV}(x) &= \varnothing; \\ \mathrm{BV}(M\,N) &= \mathrm{BV}(M) \cup \mathrm{BV}(N); \\ \mathrm{BV}(\lambda x.\, M) &= \mathrm{BV}(M) \cup \{x\}. \end{aligned}$$

Свободные и связанные переменные (3)

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

Классические комбинаторы:

$$\mathbf{I} \equiv \lambda x.\, x;$$

$$\omega \equiv \lambda x.\, x\, x; \qquad \mathbf{\Omega} \equiv \omega\, \omega = (\lambda x.\, x\, x)(\lambda x.\, x\, x);$$

$$\mathbf{K} \equiv \lambda x\, y.\, x; \qquad \mathbf{K}_* \equiv \lambda x\, y.\, y;$$

$$\mathbf{S} \equiv \lambda f\, g\, x.\, f\, x\, (g\, x);$$

$$\mathbf{B} \equiv \lambda f\, g\, x.\, f\, (g\, x).$$

Функции нескольких переменных, каррирование

Шонфинкель (1924): функции нескольких переменных могут быть описаны последовательным применением. Пусть $\varphi(x, y, z)$ — терм, зависящий от x, y, z.

$$\Phi_{x,y} = \lambda z.\, \varphi(x, y, z)$$
$$\Phi_x = \lambda y.\, \Phi_{x,y} = \lambda y.\, (\lambda z.\, \varphi(x, y, z))$$
$$\Phi = \lambda x.\, \Phi_x = \lambda x.\, (\lambda y.\, (\lambda z.\, \varphi(x, y, z))) = \lambda x\, y\, z.\, \varphi(x, y, z)$$

Тогда

$$\Phi\, X\, Y\, Z = ((\Phi\, X)\, Y)\, Z = (\Phi_X\, Y)\, Z = \Phi_{X,Y}\, Z = \varphi(X, Y, Z).$$

В общем случае

$$(\lambda \overrightarrow{x}.\, \varphi(\overrightarrow{x}))\, \overrightarrow{N} = \varphi(\overrightarrow{N}).$$

Тождественное равенство термов

Имена связанных переменных не важны. Переименуем x в y:

$$\lambda x.\, M[x], \quad \lambda y.\, M[y]$$

Они ведут себя (при подстановках) одинаково:

$$(\lambda x.\, M[x])\, N = M[x := N], \quad (\lambda y.\, M[y])\, N = M[y := N]$$

Поэтому $M \equiv N$ обозначает, что M и N – это один и тот же терм с точностью до переименования связанных переменных. Например,

$$\begin{aligned} (\lambda x.\, x)\, z &\equiv (\lambda x.\, x)\, z; \\ (\lambda x.\, x)\, z &\equiv (\lambda y.\, y)\, z. \end{aligned}$$

Иногда такое переименование называют α-преобразованием и пишут $M \equiv_\alpha N$.

Подстановка (1)

$M[x := N]$ обозначает ***подстановку*** N вместо свободных вхождений x в M.

Правила подстановки:

$$\begin{aligned}
x[x := N] &\equiv N;\\
y[x := N] &\equiv y;\\
(P\,Q)[x := N] &\equiv (P[x := N])\,(Q[x := N]);\\
(\lambda y.\,P)[x := N] &\equiv \lambda y.\,(P[x := N]),\ y \notin FV(N);\\
(\lambda x.\,P)[x := N] &\equiv (\lambda x.\,P).
\end{aligned}$$

Подразумевается, что $x \not\equiv y$.

Пример:

$$((\lambda x.\,(\lambda x.\,x\,z)\,x)\,x)[x := N] \equiv (\lambda x.\,(\lambda x.\,x\,z)\,x)\,N$$

Подстановка (2)

Неприятность: $(\lambda y.\, x\, y)[x := y]$ ($y \in \mathrm{FV}(N)$ в четвёртом правиле).

Соглашение Барендрегта: Имена связанных переменных всегда будем выбирать так, чтобы они отличались от свободных переменных в терме (термах).

Например, вместо

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

будем писать

$$y(\lambda x\, y'.\, x\, y'\, z)$$

Тогда можно использовать подстановку без оговорки о свободных и связанных переменных.

Лемма подстановки

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

$$M[x := N][y := L] \equiv M[y := L][x := N[y := L]].$$

Доказательство. Индукцией по структуре M.
1. $M = z$. Тривиально.
2. $M = x$.

$$x[x := N][y := L] = N[y := L];$$
$$x[y := L][x := N[y := L]] = x[x := N[y := L]] = N[y := L].$$

3. $M = y$.

$$y[x := N][y := L] = y[y := L] = L;$$
$$y[y := L][x := N[y := L]] = L[x := N[y := L]] = L, \text{ т.к. } x \notin FV(L).$$

Из Пирса

Доказательства программ настолько скучны, что социальные механизмы математики на них не работают.

Ричард Де Милло, Ричард Липтон и Алан Перлис, 1979

...Поэтому при верификации не стоит рассчитывать на социальные механизмы.

Дэвид Дилл, 1999

Лемма подстановки (2)

4. $M = P\,Q$. Имеем IH: для P и Q лемма верна.

$$\begin{array}{rcl}(P\,Q)[x := N][y := L] & = & (P[x := N][y := L])(Q[x := N][y := L]) \\ & =_{IH} & (P[y := L][x := N[y := L]])(Q[y := L][x := N[y := L]]) \\ & = & (P\,Q)[y := L][x := N[y := L]].\end{array}$$

5. $M = \lambda z.\, P$. Имеем IH: для P лемма верна.

▶ 5(a). $z \notin FV(N) \cup FV(L)$.

$$\begin{array}{rcl}(\lambda z.\, P)[x := N][y := L] & = & \lambda z.\, P[x := N][y := L] \\ & =_{IH} & \lambda z.\, P[y := L][x := N[y := L]] \\ & = & (\lambda z.\, P)[y := L][x := N[y := L]].\end{array}$$

▶ 5(b). $z \in FV(N) \cup FV(L)$?

6. $M = \lambda x.\, P$?

7. $M = \lambda y.\, P$?

Завершите доказательство.

Преобразования (конверсии): β

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

$$(\lambda x\,.\,M)N = M[x := N] \qquad (\beta)$$

- «Логические» аксиомы и правила:

$$M = M; \quad M = N \Rightarrow N = M; \quad M = N, N = L \Rightarrow M = L;$$
$$M = M' \Rightarrow MZ = M'Z; \qquad M = M' \Rightarrow ZM = ZM';$$
$$M = M' \Rightarrow \lambda x.\,M = \lambda x.\,M' \quad (\text{правило } \xi).$$

- Если $M = N$ доказуемо в λ-исчислении, пишут $\lambda \vdash M = N$.

Преобразования (конверсии): α и η

Иногда вводят:

- схему аксиом α-преобразования:

$$\lambda x\,.\,M = \lambda y\,.\,M[x := y] \quad (\alpha)$$

  в предположении, что $y \notin \mathrm{FV}(M)$;

- схему аксиом η-преобразования:

$$\lambda x\,.\,M\,x = M \qquad (\eta)$$

  в предположении, что $x \notin \mathrm{FV}(M)$.

Преобразования (конверсии): α

Для рассуждений достаточно соглашения Барендрегта, но для компьютерной реализации α-преобразование полезно:

Пусть $\omega \equiv \lambda x.\, x\, x$ и $1 \equiv \lambda y\, z.\, y\, z$. Тогда

$$\begin{aligned}
\omega\, 1 &\equiv (\lambda x.\, x\, x)(\lambda y\, z.\, y\, z) \\
&= (\lambda y\, z.\, y\, z)(\lambda y\, z.\, y\, z) \\
&= \lambda z.\, (\lambda y\, z.\, y\, z)\, z \\
&\equiv \lambda z.\, (\lambda y\, z'.\, y\, z')\, z \\
&= \lambda z\, z'.\, z\, z' \\
&\equiv \lambda y\, z.\, y\, z \\
&\equiv 1.
\end{aligned}$$

Преобразования (конверсии): α

Индексы Де Брауна (De Bruijn) представляют альтернативный способ представления термов.

Переменные не именуются, а нумеруются (индексируются), индекс показывает, сколько лямбд назад переменная была связана:

$$\begin{aligned} \lambda x.\,(\lambda y.\,x\,y) &\leftrightarrow \lambda\,(\lambda\,2\,1) \\ \lambda x.\,x\,(\lambda y.\,x\,y\,y) &\leftrightarrow \lambda\,1\,(\lambda\,2\,1\,1) \end{aligned}$$

Подробнее [ЛИСС, Приложение С], [TAPL, 6]

Преобразования (конверсии): η

η-преобразование обеспечивает принцип ***экстенсиональности***: две функции считаются экстенсионально эквивалентными, если они дают одинаковый результат при одинаковом вводе:

$$\forall x \ : \ F\,x = G\,x.$$

Выбирая $y \notin FV(F) \cup FV(G)$, получаем ($\xi$, затем η)

$$\begin{aligned} F\,y &= G\,y \\ \lambda y.\,F\,y &= \lambda y.\,G\,y \\ F &= G \end{aligned}$$

Термовые уравнения

Схема β-редукции даёт нам возможность решать простейшие уравнения на термы.

Пример: найти F, такой что $\forall M, N, L \quad \lambda \vdash F\,M\,N\,L = M\,L\,(N\,L)$.

$$
\begin{aligned}
F\,M\,N\,L &= M\,L\,(N\,L)\\
F\,M\,N &= \lambda z.\, M\,z\,(N\,z)\\
F\,M &= \lambda y.\, \lambda z.\, M\,z\,(y\,z)\\
F &= \lambda x\,y\,z.\, x\,z\,(y\,z)
\end{aligned}
$$

А если уравнение рекурсивное, например, $F\,M = M\,F$?

Теорема неподвижной точки (1)

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

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

Док-во. Введем $W \equiv \lambda x.\, F\,(x\,x)$ и $X \equiv WW$. Тогда

$$X \equiv WW \equiv (\lambda x.\, F(x\,x))\,W = F\,(WW) \equiv FX \quad \blacksquare$$

Теорема. Существует комбинатор неподвижной точки

$$\mathbf{Y} \equiv \lambda f.\, (\lambda x.\, f\,(x\,x))(\lambda x.\, f\,(x\,x)),$$

такой что $\forall F \quad F(\mathbf{Y}\,F) = \mathbf{Y}\,F$.

Док-во. $\mathbf{Y}\,F \equiv (\lambda x.\, F\,(x\,x))(\lambda x.\, F\,(x\,x)) = F\,(\underbrace{(\lambda x.\, F\,(x\,x))(\lambda x.\, F\,(x\,x))}_{\mathbf{Y}\,F}) \equiv$

$F(\mathbf{Y}\,F) \quad \blacksquare$

Теорема неподвижной точки (2)

Y-комбинатор позволяет ввести рекурсию в λ-исчисление.

Факториал рекурсивно:

$$\mathsf{FAC} = \lambda \mathtt{n}.\,\mathsf{IIF}\,(\mathsf{ISZRO}\ \mathtt{n})\ \mathsf{1}\ (\mathsf{MULT}\ \mathtt{n}\,(\mathsf{FAC}\,(\mathsf{PRED}\ \mathtt{n})))$$

Переписываем в виде

$$\mathsf{FAC} = (\lambda \mathtt{f}\,\mathtt{n}.\,\mathsf{IIF}\,(\mathsf{ISZRO}\ \mathtt{n})\ \mathsf{1}\ (\mathsf{MULT}\ \mathtt{n}\,(\mathtt{f}\,(\mathsf{PRED}\ \mathtt{n}))))\,\mathsf{FAC}$$

Отсюда видно, что FAC — неподвижная точка для функции $\mathsf{F} \equiv \lambda \mathtt{f}\,\mathtt{n}.\,\mathsf{IIF}\,(\mathsf{ISZRO}\ \mathtt{n})\ \mathsf{1}\ (\mathsf{MULT}\ \mathtt{n}\,(\mathtt{f}\,(\mathsf{PRED}\ \mathtt{n})))$:

$$\mathsf{FAC} = \mathbf{Y}\,\mathsf{F}$$

Теорема неподвижной точки (3)

Как работает $\mathsf{FAC} \equiv \mathbf{Y}\,\mathsf{F}$?

$$
\begin{aligned}
\mathsf{FAC}\ 3 &= (\mathbf{Y}\,\mathsf{F})\,3 \\
&= \mathsf{F}\,(\mathbf{Y}\,\mathsf{F})\,3 \\
&= \mathsf{IIF}\,(\mathsf{ISZRO}\ 3)\ 1\ (\mathsf{MULT}\ 3\,((\mathbf{Y}\,\mathsf{F})\,(\mathsf{PRED}\ 3))) \\
&= \mathsf{MULT}\ 3\,((\mathbf{Y}\,\mathsf{F})\,2) \\
&= \mathsf{MULT}\ 3\,(\mathsf{F}\,(\mathbf{Y}\,\mathsf{F})\,2) \\
&= \mathsf{MULT}\ 3\,(\mathsf{MULT}\ 2\,((\mathbf{Y}\,\mathsf{F})\,1)) \\
&= \mathsf{MULT}\ 3\,(\mathsf{MULT}\ 2\,(\mathsf{MULT}\ 1\,((\mathbf{Y}\,\mathsf{F})\,0))) \\
&= \mathsf{MULT}\ 3\,(\mathsf{MULT}\ 2\,(\mathsf{MULT}\ 1\ 1)) \\
&= 6
\end{aligned}
$$

Домашнее задание

Докажите,
– что $\mathbf{S\,K\,K} = \mathbf{I}$, $\mathbf{B} = \mathbf{S}\,(\mathbf{K\,S})\,\mathbf{K}$.
– что применение некоммутативно и неассоциативно.

Завершите доказательство леммы подстановки.

Реализуйте алгоритм подстановки на каком-либо ЯП.

Сконструируйте
– «пожиратель», то есть такой терм F, который для любого M обеспечивает $\mathrm{F\,M} = \mathrm{F}$.
– терм F таким образом, чтобы для любого M выполнялось $\mathrm{F\,M} = \mathrm{M\,F}$.
– терм F таким образом, чтобы для любых термов M и N выполнялось $\mathrm{F\,M\,N} = \mathrm{N\,F}\,(\mathrm{N\,M\,F})$.

Литература (1)

LCWT гл. 2.1
Henk Barendregt, Lambda calculi with types,
Handbook of logic in computer science (vol. 2), Oxford University Press, 1993

TAPL гл. 5, 6
Benjamin C. Pierce, Types and Programming Languages, MIT Press, 2002

Литература (2)

I2FP гл. 2

John Harrison, Introduction to Functional Programming

ЛИСС гл. 2

Х. Барендрегт, Ламбда-исчисление, его синтаксис и семантика, М:Мир, 1985