LMSE

Logische Methoden des Software Engineerings

Prof. Dr. Jakob Rehof

Lehrstuhl XIV, Software Engineering

Simply typed lambda calculus

- Type systems are deductive systems assigning types to lambda terms
- The typable terms are in many (most) cases strict subsets of the set of untyped terms
- Type systems come in different strengths: a hierarchy with simple types at the bottom
- Two kinds of type systems:
  - *a la Curry* : *implicit* type system (type inference)
  - *a la Church* : *explicit* type system (type checking)

Type expressions

3.1.1. DEFINITION.

(i) Let U denote a denumerably infinite alphabet whose members will be called *type variables*. The set Π of *simple types* is the set of strings defined by the grammar:

$$\Pi ::= U \mid (\Pi \to \Pi)$$

We use $\alpha, \beta, \ldots$ to denote arbitrary type variables, and $\sigma, \tau, \ldots$ to denote arbitrary types. We omit outermost parentheses, and omit other parentheses with the convention that $\to$ associates to the right.

Type expressions

(ii) The set C of *contexts* is the set of all sets of pairs of the form

$$\{x_1 : \tau_1, \ldots, x_n : \tau_n\}$$

with $\tau_1, \ldots, \tau_n \in \Pi$, $x_1, \ldots, x_n \in V$ (variables of Λ) and $x_i \neq x_j$ for $i \neq j$.

(iii) The *domain* of a context $\Gamma = \{x_1 : \tau_1, \ldots, x_n : \tau_n\}$ is defined by:

$$\mathrm{dom}(\Gamma) = \{x_1, \ldots, x_n\}$$

We write $x : \tau$ for $\{x : \tau\}$ and Γ, Γ' for $\Gamma \cup \Gamma'$ if $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma') = \{\}$.

(iv) The *range* of a context $\Gamma = \{x_1 : \tau_1, \ldots, x_n : \tau_n\}$ is defined by:

$$|\Gamma| = \{\tau \in \Pi \mid (x : \tau) \in \Gamma, \text{ for some } x\}.$$

Type system (a la Curry)

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}$$ [Var]

$$\frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$ [Abs]

$$\frac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash M\ N : \tau}$$ [App]

Example

3.1.2. EXAMPLE. Let σ, τ, ρ be arbitrary types. Then:

(i) $\vdash \lambda x.x : \sigma \to \sigma$;

(ii) $\vdash \lambda x.\lambda y.x : \sigma \to \tau \to \sigma$;

(iii) $\vdash \lambda x.\lambda y.\lambda z.x\ z\ (y\ z) : (\sigma \to \tau \to \rho) \to (\sigma \to \tau) \to \sigma \to \rho$.

Free variables lemma

3.1.5. LEMMA (Free variables lemma). *Assume that* $\Gamma \vdash M : \sigma$. *Then:*

(i) $\Gamma \subseteq \Gamma'$ *implies* $\Gamma' \vdash M : \sigma$;

(ii) $\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$;

(iii) $\Gamma' \vdash M : \sigma$ *where* $\mathrm{dom}(\Gamma') = \mathrm{FV}(M)$ *and* $\Gamma' \subseteq \Gamma$.

PROOF. (i) by induction on the derivation of $\Gamma \vdash M : \sigma$.

(ii)-(iii) by induction on the derivation of $\Gamma \vdash M : \sigma$.

Generation lemma

3.1.6. LEMMA (Generation lemma).

(i) $\Gamma \vdash x : \sigma$ *implies* $x : \sigma \in \Gamma$;

(ii) $\Gamma \vdash M\ N : \sigma$ *implies that there is a* τ *such that* $\Gamma \vdash M : \tau \to \sigma$ *and* $\Gamma \vdash N : \tau$.

(iii) $\Gamma \vdash \lambda x.M : \sigma$ *implies that there are* τ *and* ρ *such that* $\Gamma, x : \tau \vdash M : \rho$ *and* $\sigma = \tau \to \rho$.

PROOF. By induction on the length of the derivation. □

Type substitution

3.1.7. DEFINITION. The *substitution of type τ for type variable α in type σ*, written $\sigma[\alpha := \tau]$, is defined by:

$$
\begin{array}{lcll}
\alpha[\alpha := \tau] & = & \tau & \\
\beta[\alpha := \tau] & = & \beta & \text{if } \alpha \neq \beta \\
(\sigma_1 \to \sigma_2)[\alpha := \tau] & = & \sigma_1[\alpha := \tau] \to \sigma_2[\alpha := \tau] &
\end{array}
$$

The notation $\Gamma[\alpha := \tau]$ stands for the context $\{(x : \sigma[\alpha := \tau]) \mid (x : \sigma) \in \Gamma\}$.

Substitution lemma

3.1.8. PROPOSITION (Substitution lemma).

(i) *If* $\Gamma \vdash M : \sigma$, *then* $\Gamma[\alpha := \tau] \vdash M : \sigma[\alpha := \tau]$.

(ii) *If* $\Gamma, x : \tau \vdash M : \sigma$ *and* $\Gamma \vdash N : \tau$ *then* $\Gamma \vdash M[x := N] : \sigma$.

PROOF. By induction on the derivation of $\Gamma \vdash M : \sigma$ and generation of $\Gamma, x : \tau \vdash M : \sigma$, respectively. □

- The substitution property is a limited form of polymorphism

Subject reduction

3.1.9. PROPOSITION (Subject reduction). *If* $\Gamma \vdash M : \sigma$ *and* $M \to_\beta N$, *then* $\Gamma \vdash N : \sigma$.

PROOF. By induction on the derivation of $M \to_\beta N$ using the substitution lemma and the generation lemma. □

3.1.11. COROLLARY. *If* $\Gamma \vdash M : \sigma$ *and* $M \twoheadrightarrow_\beta N$, *then* $\Gamma \vdash N : \sigma$.

3.1.10. REMARK. The similar property

$$\Gamma \vdash N : \sigma \;\&\; M \twoheadrightarrow_\beta N \;\Rightarrow\; \Gamma \vdash M : \sigma$$

is called *subject expansion* and does *not* hold in $\lambda\!\to$, see Exercise 3.6.2.

Simple types a la Church

3.2.1. DEFINITION.

(i) The set Λ_Π of pseudo-terms is defined by the following grammar:

$$\Lambda_\Pi ::= V \mid (\lambda x{:}\,\Pi\, \Lambda_\Pi) \mid (\Lambda_\Pi\, \Lambda_\Pi)$$

where V is the set of (λ-term) variables and Π is the set of simple types.[1] We adopt the same terminology, notation, and conventions for pseudo-terms as for λ-terms, see 1.3–1.10, mutatis mutandis.

(ii) The *typability* relation $\Vdash$ on $C \times \Lambda_\Pi \times \Pi$ is defined by:[2]

$$\frac{}{\Gamma, x:\tau \vdash^* x:\tau} \qquad \frac{\Gamma, x:\sigma \vdash^* M:\tau}{\Gamma \vdash^* \lambda x{:}\sigma.M : \sigma \to \tau} \qquad \frac{\Gamma \vdash^* M:\sigma \to \tau \quad \Gamma \vdash^* N:\sigma}{\Gamma \vdash^* M\ N:\tau}$$

where we require that $x \notin \mathrm{dom}(\Gamma)$ in the first and second rule.

(iii) The simply typed λ-calculus à la Church ($\lambda{\to}$ *à la Church,* for short) is the triple $(\Lambda_\Pi, \Pi, \Vdash)$.

(iv) If $\Gamma \vdash^* M:\sigma$ then we say that M *has type* σ *in* Γ. We say that $M \in \Lambda_\Pi$ is *typable* if there are Γ and σ such that $\Gamma \vdash^* M:\sigma$.

Example

3.2.2. EXAMPLE. Let σ, τ, ρ be arbitrary simple types. Then:

(i) $\vdash^* \lambda x{:}\sigma.x : \sigma \to \sigma$;

(ii) $\vdash^* \lambda x{:}\sigma.\lambda y{:}\tau.x : \sigma \to \tau \to \sigma$;

(iii) $\vdash^* \lambda x{:}\sigma{\to}\tau{\to}\rho.\lambda y{:}\sigma{\to}\tau.\lambda z{:}\sigma.(x\ z)\ y\ z : (\sigma{\to}\tau{\to}\rho){\to}(\sigma{\to}\tau){\to}\sigma{\to}\rho.$

Special properties of Church-system

3.2.12. PROPOSITION (Uniqueness of types).

(i) *If* $\Gamma \vdash^* M : \sigma$ *and* $\Gamma \vdash^* M : \tau$ *then* $\sigma = \tau$.

(ii) *If* $\Gamma \vdash^* M : \sigma$ *and* $\Gamma \vdash^* N : \tau$ *and* $M =_\beta N$, *then* $\sigma = \tau$.

Height of a type

3.4.1. DEFINITION. Define the function $h : \Pi \to \mathbb{N}$ by:

$$\begin{aligned} h(\alpha) &= 0 \\ h(\tau \to \sigma) &= 1 + \max(h(\tau), h(\sigma)) \end{aligned}$$

Normalization

3.4.2. THEOREM (Weak normalization). *Suppose* $\Gamma \vdash^* M : \sigma$. *Then there is a finite reduction* $M_1 \to_\beta M_2 \to_\beta \ldots \to_\beta M_n \in \mathrm{NF}_\beta$.

PROOF. We use a proof idea due independently to Turing and Prawitz.

Define the *height* of a redex $(\lambda x{:}\tau.P^\rho)R$ to be $h(\tau \to \rho)$. For $M \in \Lambda_\Pi$ with $M \notin \mathrm{NF}_\beta$ define

$$m(M) = (h(M), n)$$

where

$$h(M) = \max\{h(\Delta) \mid \Delta \text{ is a redex in } M\}$$

and n is the number of redex occurrences in M of height $h(M)$. If $M \in \mathrm{NF}_\beta$ we define $h(M) = (0, 0)$.

Normalization

We show by induction on lexicographically ordered pairs $m(M)$ that if M is typable in $\lambda\!\!\rightarrow$ à la Church, then M has a reduction to normal-form.

Let $\Gamma \vdash M : \sigma$. If $M \in \mathrm{NF}_\beta$ the assertion is trivially true. If $M \notin \mathrm{NF}_\beta$, let Δ be the rightmost redex in M of maximal height h (we determine the position of a subterm by the position of its leftmost symbol, i.e., the rightmost redex means the redex which *begins* as much to the right as possible).

Let M' be obtained from M by reducing the redex Δ. The term M' may in general have more redexes than M. But we claim that the number of redexes of height h in M' is smaller than in M. Indeed, the redex Δ has disappeared, and the reduction of Δ may only create new redexes of height less than h. To see this, note that the number of redexes can increase by either copying existing redexes or by creating new ones.

Normalization

Now observe that if a new redex is created then one of the following cases must hold:

1. The redex Δ is of the form $(\lambda x{:}\tau.\ \ldots x P^{\rho}\ \ldots)(\lambda y^{\rho}.Q^{\mu})^{\tau}$, where $\tau = \rho \to \mu$, and reduces to $\ldots\ (\lambda y^{\rho}.Q^{\mu})P^{\rho}\ \ldots$. There is a new redex $(\lambda y^{\rho}.Q^{\mu})P^{\rho}$ of height $h(\tau) < h$.

2. We have $\Delta = (\lambda x{:}\tau.\lambda y{:}\rho.R^{\mu})P^{\tau}$, occurring in the context $\Delta^{\rho\to\mu}Q^{\rho}$. The reduction of Δ to $\lambda y{:}\rho.R_1^{\mu}$, for some R_1, creates a new redex $(\lambda y{:}\rho.R_1^{\mu})Q^{\rho}$ of height $h(\rho \to \mu) < h(\tau \to \rho \to \mu) = h$.

3. The last case is when $\Delta = (\lambda x{:}\tau.x)(\lambda y^{\rho}.P^{\mu})$, with $\tau = \rho \to \mu$, and it occurs in the context $\Delta^{\tau}Q^{\rho}$. The reduction creates the new redex $(\lambda y^{\rho}.P^{\mu})Q^{\rho}$ of height $h(\tau) < h$.

Normalization

The other possibility of adding redexes is by copying. If we have $\Delta = (\lambda x{:}\tau.P^{\rho})Q^{\tau}$, and P contains more than one free occurrence of x, then all redexes in Q are multiplied by the reduction. But we have chosen Δ to be the rightmost redex of height h, and thus all redexes in Q must be of smaller heights, because they are to the right of Δ.

Thus, in all cases $m(M) > m(M')$, so by the induction hypothesis M' has a normal-form, and then M also has a normal-form. $\square$

Expressibility

3.5.1. DEFINITION. Let

$$\mathbf{int} = (\alpha \to \alpha) \to (\alpha \to \alpha)$$

where α is an arbitrary type variable. A numeric function $f : \mathbb{N}^n \to \mathbb{N}$ is $\lambda\!\to$*-definable* if there is an $F \in \Lambda$ with $\vdash F : \mathbf{int} \to \cdots \to \mathbf{int} \to \mathbf{int}$ ($n+1$ occurrences of **int**) such that

$$F\ c_{n_1}\ \ldots\ c_{n_m} =_\beta c_{f(n_1,\ldots,n_m)}$$

for all $n_1, \ldots, n_m \in \mathbb{N}$.

Expressibility

3.5.5. DEFINITION. The class of *extended polynomials* is the smallest class of numeric functions containing the

(i) *projections:* $U_i^m(n_1, \ldots, n_m) = n_i$ for all $1 \leq i \leq m$;

(ii) *constant functions:* $k(n) = k$;

(iii) *signum function:* $sg(0) = 0$ and $sg(m+1) = 1$.

and closed under *addition* and *multiplication*:

(i) *addition:* if $f : \mathbb{N}^k \to \mathbb{N}$ and $g : \mathbb{N}^l \to \mathbb{N}$ are extended polynomials, then so is $(f + g) : \mathbb{N}^{k+l} \to \mathbb{N}$

$$(f + g)(n_1, \ldots, n_k, m_1, \ldots, m_l) = f(n_1, \ldots, n_k) + g(m_1, \ldots, m_l)$$

(ii) *multiplication:* if $f : \mathbb{N}^k \to \mathbb{N}$ and $g : \mathbb{N}^l \to \mathbb{N}$ are extended polynomials, then so is $(f \cdot g) : \mathbb{N}^{k+l} \to \mathbb{N}$

$$(f \cdot g)(n_1, \ldots, n_k, m_1, \ldots, m_l) = f(n_1, \ldots, n_k) \cdot g(m_1, \ldots, m_l)$$

3.5.6. THEOREM (Schwichtenberg). *The $\lambda{\to}$-definable functions are exactly the extended polynomials.*