Komponenten- und Service-orientierte Softwarekonstruktion

Lecture 2: Types and combinators

Jakob Rehof
LS XIV – Software Engineering

TU Dortmund
Sommersemester 2017

SS 2017

Simple types

Definition 1 (Simple types)

Let $\mathbb{V}$ denote a denumerable set of *type variables*, ranged over by metavariables $\alpha, \beta, \gamma, \ldots$, and let b range over a set $\mathbb{B}$ of *type constants*. The set $\mathbb{T}$ of *simple types*, ranged over by $\tau, \sigma, \rho, \ldots$, is defined inductively by:

- $\alpha \in \mathbb{V} \Rightarrow \alpha \in \mathbb{T}$
- $b \in \mathbb{B} \Rightarrow b \in \mathbb{T}$
- $\tau \in \mathbb{T}, \sigma \in \mathbb{T} \Rightarrow \tau \to \sigma \in \mathbb{T}$

Type variables and type constants are referred to as *atoms*. We write $\tau \to \sigma \to \rho$ for $\tau \to (\sigma \to \rho)$.

A *type environment* is a finite set Γ of *type assumptions* of the form $\Gamma = \{(x_1 : \tau_1), \ldots, (x_n : \tau_n)\}$ with $x_i \not\equiv x_j$ for $i \neq j$. We let $\mathsf{Dm}(\Gamma) = \{x \in \mathcal{V} \mid \exists \tau \in \mathbb{T}.\ (x : \tau) \in \Gamma\}$. We write $\Gamma, x : \tau$ for $\Gamma \cup \{(x : \tau)\}$.

Simple typed λ-calculus, $\lambda^{\rightarrow}$

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}(\mathsf{var})$$

$$\frac{\Gamma, x : \tau \vdash M : \sigma}{\Gamma \vdash \lambda x.M : \tau \rightarrow \sigma}(\rightarrow\mathsf{I})$$

$$\frac{\Gamma \vdash M : \tau \rightarrow \sigma \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \sigma}(\rightarrow\mathsf{E})$$

Explicit typing

$$\frac{}{\Gamma, x : \tau \vdash^* x : \tau}(\mathsf{var})$$

$$\frac{\Gamma, x : \tau \vdash^* M : \sigma}{\Gamma \vdash^* \lambda x{:}\,\tau.M : \tau \to \sigma}(\to\mathsf{I})$$

$$\frac{\Gamma \vdash^* M : \tau \to \sigma \quad \Gamma \vdash^* N : \tau}{\Gamma \vdash^* MN : \sigma}(\to\mathsf{E})$$

Exercise 1

Show by induction on M: if $\Gamma \vdash^* M : \sigma$ and $\Gamma \vdash^* M : \tau$, then $\sigma \equiv \tau$.

Basis lemma

Let $\Gamma \downarrow_V = \{(x : \tau) \in \Gamma \mid x \in V\}$.

Lemma 2

- *If* $\Gamma \subseteq \Gamma'$ *then* $\Gamma \vdash M : \tau$ *implies* $\Gamma' \vdash M : \tau$,
- *If* $\Gamma \vdash M : \tau$, *then* $\mathsf{FV}(M) \subseteq \mathsf{Dm}(\Gamma)$,
- *If* $\Gamma \vdash M : \tau$, *then* $\Gamma \downarrow_{\mathsf{FV}(M)} \vdash M : \tau$.

Exercise 2

Prove Lemma 2.

Inversion

Lemma 3 (Generation lemma)

Suppose that $\Gamma \vdash M : \tau$.

1. *If* $M \equiv x$, *then* $\Gamma = \Gamma', x : \tau$,
2. *If* $M \equiv \lambda x.N$, *then* $\tau \equiv \rho \to \sigma$ *and* $\Gamma, x : \rho \vdash N : \sigma$,
3. *If* $M \equiv PQ$, *then* $\Gamma \vdash P : \sigma \to \tau$ *and* $\Gamma \vdash Q : \sigma$ *for some* σ.

Proof.

Immediate by inspection of the last rule used in the derivation of $\Gamma \vdash M : \tau$. □

Exercise 3

Call a term M *typable, if* $\Gamma \vdash M : \tau$ *for some* Γ *and* τ.

Show that every subterm of a typable term is typable.

Substitutivity

A *type substitution* is a map $S : \mathbb{V} \to \mathbb{T}$, and it is lifted homomorphically to a map $S : \mathbb{T} \to \mathbb{T}$. Let $S(\Gamma) = \{(x : S(\tau)) \mid (x : \tau) \in \Gamma\}$.

Lemma 4 (Substitution)

1. *If* $\Gamma, x : \tau \vdash M : \sigma$ *and* $\Gamma \vdash N : \tau$, *then* $\Gamma \vdash M[x := N] : \sigma$.
2. *If* $\Gamma \vdash M : \tau$, *then* $S(\Gamma) \vdash M : S(\tau)$, *for any type substitution* S.

Exercise 4

Prove Lemma 4 by induction on derivations.

Subject reduction

Theorem 5 (Subject Reduction)

If $\Gamma \vdash M : \tau$ *and* $M \twoheadrightarrow_\beta N$, *then* $\Gamma \vdash N : \tau$.

Proof.

(Sketch) First prove the statement for the case when M is a redex and N its reduct, using the substitution lemma. Then prove the statement when $M \to_\beta N$ by reduction of a redex R in M, by induction on C where $M \equiv C[R]$. Then prove the statement by induction in the length of the reduction $M \twoheadrightarrow_\beta N$. □

Exercise 5

Complete the proof sketch of Theorem 5.

Strong normalization

Definition 6

A λ-term M is called *strongly normalizing*, if every β-reduction sequence starting from M is finite, and *weakly normalizing*, if there exists a finite reduction sequence starting from M.

Theorem 7

Every typable term in $\lambda^{\rightarrow}$ is strongly normalizing.

Proof.

See lecture notes. □

Combinators

Let $\mathcal{C}$ be a set of *combinator symbols*, ranged over by X, Y, Z. The set $\Xi_{\mathcal{C}}$ of *combinatory expressions*, ranged over by F, G, H are defined inductively by:

- $X \in \mathcal{C} \Rightarrow X \in \Xi_{\mathcal{C}}$,
- $x \in \mathcal{V} \Rightarrow x \in \Xi_{\mathcal{C}}$,
- $F, G \in \Xi_{\mathcal{C}} \Rightarrow (FG) \in \Xi_{\mathcal{C}}$

Consider the set $\mathsf{SKI} = \{\mathbf{S}, \mathbf{K}, \mathbf{I}\}$ of combinator symbols (referred to as a combinatory *base*) and define the notion of *weak reduction* $\rhd_w$ on Ξ_{SKI} by setting, for all $X, Y, Z \in \Xi_{\mathsf{SKI}}$:

$$\begin{array}{lll} \mathbf{I}F & \rhd_w & F \\ \mathbf{K}FG & \rhd_w & F \\ \mathbf{S}FGH & \rhd_w & (FH)(GH) \end{array}$$

Let $\rightarrow_w$ and $\twoheadrightarrow_w$ be the reduction relations on Ξ_{SKI} induced by $\rhd_w$, by closing $\rhd_w$ under contexts $C ::= [] \mid (CF) \mid (FC), \quad (F \in \Xi_{\mathsf{SKI}})$.

Combinatory bases

By choosing different sets $\mathfrak{B} \subseteq \mathcal{C}$ of combinators (such as $\mathfrak{B} = \mathsf{SKI} = \{\mathbf{S}, \mathbf{K}, \mathbf{I}\}$) we can study different combinatory calculi, since in each case we can consider a $\mathfrak{B}$-calculus generated from the combinators in $\mathfrak{B}$. In such cases, we refer to the set $\mathfrak{B}$ as a *combinatory base*.

Exercise 6

Show that the combinator $\mathbf{I}$ *can be coded in terms of* $\mathbf{S}$ *and* $\mathbf{K}$*. Hint: Notice that* $(\mathbf{K}F)(\mathbf{K}F) \twoheadrightarrow_w F$*.*

In other words, the base $SKI = \{\mathbf{S}, \mathbf{K}, \mathbf{I}\}$ *is redundant. Or, in yet other words, the base* $SK = \{\mathbf{S}, \mathbf{K}\}$ *is complete with respect to SKI-calculus. For this reason, one also talks about SK-calculus.*

Combinatory bases

Schönfinkel used, in addition to $\mathbf{S}$ and $\mathbf{K}$, the combinators $\mathbf{B}$ and $\mathbf{C}$ with the definitions

$$\begin{array}{lll} \mathbf{B}FGH & \rhd_B & FGH \\ \mathbf{C}FGH & \rhd_C & FHG \end{array}$$

But they are not strictly needed, for we can take

$$\begin{array}{lll} \mathbf{B} & \equiv & \mathbf{S}(\mathbf{KS})\mathbf{K} \\ \mathbf{C} & \equiv & \mathbf{S}(\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{KS})\mathbf{K}))\mathbf{S})(\mathbf{KK}) \end{array}$$

Combinatory bases

Exercise 7 (One point basis)

Define the combinator $\mathbf{X}$ *by the rule*

$$(\mathbf{X}F) \rhd_X ((F\mathbf{S})\mathbf{K})$$

Show that $(\mathbf{XX}) \twoheadrightarrow_X \mathbf{SK}(\mathbf{KK})$.

Show that $\mathbf{X}(\mathbf{X}(\mathbf{XX})) \twoheadrightarrow_X \mathbf{K}$.

Show that $\mathbf{X}(\mathbf{X}(\mathbf{X}(\mathbf{XX}))) \twoheadrightarrow_X \mathbf{S}$.

Conclude that $\{\mathbf{X}\}$ *is complete with respect to SKI-calculus.*

Define the map $(\_)_\Lambda : \Xi_{\mathsf{SKI}} \to \Lambda$ by induction on expressions in Ξ_{SKI}:

$$\begin{array}{lcl} (x)_\Lambda & \equiv & x, \text{ for } x \in \mathcal{V} \\ (\mathbf{I})_\Lambda & \equiv & \lambda x.x \\ (\mathbf{K})_\Lambda & \equiv & \lambda xy.x \\ (\mathbf{S})_\Lambda & \equiv & \lambda xyz.(xz)(yz) \\ (FG)_\Lambda & \equiv & (F)_\Lambda (G)_\Lambda \end{array}$$

Proposition 1

If $F \twoheadrightarrow_w G$, then $(F)_\Lambda \twoheadrightarrow_\beta (G)_\Lambda$.

Exercise 8

Prove Proposition 1 by induction on the length of $F \twoheadrightarrow_w G$.

Define, for each $x \in \mathcal{V}$, the "bracket abstraction" map $[x] : \Xi_{\mathsf{SKI}} \to \Xi_{\mathsf{SKI}}$ by induction on expressions in Ξ_{SKI}:

$$\begin{array}{lcl} [x]x & \equiv & \mathbf{I} \\ [x]F & \equiv & \mathbf{K}F, \text{ if } x \notin \mathsf{FV}(F) \\ [x](FG) & \equiv & \mathbf{S}([x]F)([x]G), \text{ otherwise} \end{array}$$

Proposition 2 (Combinatory completeness)

1. $\forall F \in \Xi_{\mathsf{SKI}}.\forall G \in \Xi_{\mathsf{SKI}}.\ ([x]F)G \twoheadrightarrow_w F[x := G]$
2. $\forall F \in \Xi_{\mathsf{SKI}}.\ ([x]F)_\Lambda \twoheadrightarrow_\beta \lambda x.(F)_\Lambda$
3. $\forall x \in \mathcal{V}.\ \forall F \in \Xi_{\mathsf{SKI}}.\ \exists H \in \Xi_{\mathsf{SKI}}.\ \forall G \in \Xi_{\mathsf{SKI}}.\ HG \twoheadrightarrow_w F[x := G]$

Exercise 9

Prove Proposition 2.

Define the map $(\_)_\Xi : \Lambda \to \Xi_{\mathsf{SKI}}$ by induction on λ-terms:

$$\begin{array}{lcl} (x)_\Xi & \equiv & x, \text{ for } x \in \mathcal{V} \\ (MN)_\Xi & \equiv & (M)_\Xi (N)_\Xi \\ (\lambda x.M)_\Xi & \equiv & [x](M)_\Xi \end{array}$$

Proposition 3

For all $M \in \Lambda$, one has $((M)_\Xi)_\Lambda \twoheadrightarrow_\beta M$.

Exercise 10

Prove Proposition 3.

Combinatory logic SKI

$$\frac{}{\Gamma, x : \tau \vdash_{\mathrm{SKI}} x : \tau}(\mathsf{var})$$

$$\frac{}{\Gamma \vdash_{\mathrm{SKI}} \mathbf{I} : \tau \to \tau}(\mathbf{I})$$

$$\frac{}{\Gamma \vdash_{\mathrm{SKI}} \mathbf{K} : \tau \to \sigma \to \tau}(\mathbf{K})$$

$$\frac{}{\Gamma \vdash_{\mathrm{SKI}} \mathbf{S} : (\tau \to \sigma \to \rho) \to (\tau \to \sigma) \to \tau \to \rho}(\mathbf{S})$$

$$\frac{\Gamma \vdash_{\mathrm{SKI}} F : \tau \to \sigma \quad \Gamma \vdash_{\mathrm{SKI}} G : \tau}{\Gamma \vdash_{\mathrm{SKI}} (FG) : \sigma}(\to\mathsf{E})$$

Notice that variables x have fixed, *monomorphic types*, whereas combinators $\mathbf{S}, \mathbf{K}, \mathbf{I}$ have infinitely many types (their types are *schematic* or *polymorphic*). We shall return to this important point in Lecture 5.

Combinatory logic SKI

Lemma 8 (Deduction theorem for SKI)

If $\Gamma, x : \sigma \vdash_{\mathrm{SKI}} F : \tau$, *then* $\Gamma \vdash_{\mathrm{SKI}} [x]F : \sigma \to \tau$.

Proposition 4

1. *If* $\Gamma \vdash_{\mathrm{SKI}} F : \tau$, *then* $\Gamma \vdash_{\Lambda} (F)_{\Lambda} : \tau$.
2. *If* $\Gamma \vdash_{\Lambda} M : \tau$, *then* $\Gamma \vdash_{\mathrm{SKI}} (M)_{\Xi} : \tau$.

Exercise 11

Prove Proposition 4. Hint: The first statement is proven by induction on the derivation of $\Gamma \vdash_{\mathrm{SKI}} F : \tau$. *The second statement is proven by induction on the derivation of* $\Gamma \vdash_{\Lambda} M : \tau$ *using Lemma 8.*

Decision problems

- *Type checking* ($\Gamma \vdash M : \tau$?). Given Γ, M and τ, does $\Gamma \vdash M : \tau$ hold?
- *Typability* ($? \vdash M :?$). Given M, are there Γ and τ such that $\Gamma \vdash M : \tau$?
- *Inhabitation* ($\Gamma \vdash ? : \tau$). Given Γ and τ, does there exist a term ("inhabitant") M such that $\Gamma \vdash M : \tau$?

Type checking and typability for $\lambda^{\rightarrow}$ are linear time solvable.

Inhabitation for $\lambda^{\rightarrow}$ is PSPACE-complete. See Lecture 3.

Inhabitation can be seen as a program synthesis problem:

- Construct *program* M satisfying *specification* τ.