

Komponenten- und Service-orientierte Softwarekonstruktion

Lecture 1: λ -Calculus

Jakob Rehof
LS XIV – Software Engineering



TU Dortmund
Sommersemester 2017

SS 2017



Let \mathcal{V} denote a denumerable set of λ -variables, v_1, v_2, v_3, \dots

Let x, y, z, \dots denote a denumerable set of *metavariables* ranging over \mathcal{V} .

Definition 1

The set Λ of λ -terms, ranged over by M, N, P, Q, \dots , is defined inductively by:

- $x \in \mathcal{V} \Rightarrow x \in \Lambda$ (variables)
- $M, N \in \Lambda \Rightarrow (MN) \in \Lambda$ (application)
- $M \in \Lambda, x \in \mathcal{V} \Rightarrow (\lambda x M) \in \Lambda$ (abstraction)



Notation 1

We may abbreviate $((FM_1)M_2) \cdots M_n$ by

$$FM_1M_2 \cdots M_n$$

for $n \geq 0$.

We may abbreviate $(\lambda x_1(\lambda x_2(\cdots(\lambda x_n M) \cdots)))$ by

$$\lambda x_1.\lambda x_2.\cdots \lambda x_n.M$$

for $n \geq 0$.

And further, $\lambda x_1.\lambda x_2.\cdots \lambda x_n.M$ by $\lambda x_1x_2.\cdots x_n.M$.



Definition 2 (Free and bound variables)

The set of *free variables* of M , denoted $FV(M)$, is defined by induction on M :

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

A variable $x \in FV(M)$ is called *bound* in $\lambda x.M$ (by that λ). A term $M \in \Lambda$ is called *closed*, if $FV(M) = \emptyset$.



Definition 3 (Syntactic identity)

We write $M \equiv N$ to denote that M and N are the same term under renaming of bound variables (α -conversion). For example,

$$\begin{aligned}(\lambda x.x)z &\equiv (\lambda x.x)z \\(\lambda x.x)z &\equiv (\lambda y.y)z \\(\lambda x.x)x &\equiv (\lambda y.y)x \\(\lambda x.x)z &\not\equiv (\lambda x.y)z \quad (\text{provided } x \neq y)\end{aligned}$$

Note that $x \equiv y$ is possible, unless we have stipulated that $x \neq y$.

Definition 4 (Substitution)

Let $M[x := N]$ denote the result of substituting N for the free occurrences of x in M , defined by induction on M :

$$\begin{aligned}x[x := N] &\equiv N \\y[x := N] &\equiv y, \text{ if } x \neq y \\(PQ)[x := N] &\equiv (P[x := N]Q[x := N]) \\(\lambda y.P)[x := N] &\equiv \lambda y.P[x := N], \text{ if } x \neq y \\(\lambda x.P)[x := N] &\equiv \lambda x.P\end{aligned}$$



Lemma 5 (Substitution lemma)

Assume $x \neq y$ and $x \notin \text{FV}(P)$. Then

$$M[x := N][y := P] \equiv M[y := P][x := N[y := P]]$$

Exercise 1

Prove Lemma 5 by induction on M .

Notion of reduction

Definition 6

A *notion of reduction* on a set \mathbf{D} is a binary relation $\triangleright \subseteq \mathbf{D} \times \mathbf{D}$.

Definition 7 (Notion of β -reduction)

The notion of β -reduction is the relation $\triangleright_\beta \subseteq \Lambda \times \Lambda$ defined by

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

for all $M, N \in \Lambda$. An expression of the form $(\lambda x.M)N$ is called a *redex*.

Contexts

Contexts C over Λ are expressions of the form

$$C ::= \square \mid (CM) \mid (MC) \mid \lambda x.C$$

We think of \square as a hole and of C as a term with a hole in it.

If C is a context and M a term then $C[M]$ is the term which arises from exchanging \square with M (“hole-filling”).

For example, if $C \equiv \lambda x.\lambda y.y\square$, then

$$C[\lambda z.(xz)] \equiv \lambda x.\lambda y.y(\lambda z.(xz))$$

Note that variable capture can happen through hole-filling.

Definition 8

A notion of reduction $\triangleright \subseteq \Lambda \times \Lambda$ satisfying

$$\forall C. M \triangleright N \Rightarrow C[M] \triangleright C[N]$$

is called a *reduction relation*. The smallest reduction relation containing \triangleright is called the *reduction relation induced by \triangleright* (sometimes also called the compatible closure of \triangleright).

Definition 9 (β -reduction)

The relation \rightarrow_β (β -reduction) is the reduction relation induced by \triangleright_β .

If $M \rightarrow_\beta N$, then N is called a *reduct* of M .

A *normal form* is a term with no redex.



Definition 10 (Multi-step β -reduction)

The relation $\twoheadrightarrow_{\beta}$ is the reflexive transitive closure of \rightarrow_{β} .

Definition 11 (β -conversion)

The relation $=_{\beta}$ (β -conversion) is the reflexive symmetric transitive closure of \rightarrow_{β} .

If $M =_{\beta} N$, we say that M and N are β -equivalent.



λ -calculus combinators

A λ -calculus *combinator* is a closed λ -term.

Some interesting combinators are:

$$\mathbf{I} \equiv \lambda x.x, \mathbf{K} \equiv \lambda xy.x, \mathbf{K}^* \equiv \lambda xy.y, \mathbf{S} \equiv \lambda xyz.(xz)(yz)$$

$$\omega \equiv \lambda x.xx, \Omega \equiv \omega\omega$$

$$\mathbf{Y} \equiv \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)), \Theta \equiv (\lambda x.\lambda y.(y(xxy)))(\lambda x.\lambda y.(y(xxy)))$$

Exercise 2

Show that, for all terms F , we have $\mathbf{Y}F =_{\beta} F(\mathbf{Y}F)$ and $\Theta F \rightarrow_{\beta} F(\Theta F)$.



Theorem 12 (Church-Rosser, confluence)

For every $M, P_1, P_2 \in \Lambda$, if $M \rightarrow_{\beta} P_1$ and $M \rightarrow_{\beta} P_2$, then there exists $Q \in \Lambda$ such that $P_1 \twoheadrightarrow_{\beta} Q$ and $P_2 \twoheadrightarrow_{\beta} Q$.

Corollary 13 (Uniqueness of normal forms)

If N_1 and N_2 are normal forms such that $M \twoheadrightarrow_{\beta} N_1$ and $M \twoheadrightarrow_{\beta} N_2$, then $N_1 \equiv N_2$. If $P =_{\beta} Q$, then P and Q have the same normal form if any.

Exercise 3

Prove Corollary 13, by using Theorem 12.

Church numerals

For $P, M \in \Lambda$, define $P^n(M)$ by induction on $n \geq 0$:

- $P^0(M) \equiv M$
- $P^{n+1}(M) \equiv P(P^n(M))$

The *Church numerals* $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \dots$ are defined by $\mathbf{c}_n \equiv \lambda f. \lambda x. f^n(x)$.

Define $\mathbf{A}_+ \equiv \lambda xypq. xp(ypq)$, $\mathbf{A}_\times \equiv \lambda xyz. x(yz)$, $\mathbf{A}_e \equiv \lambda xy. yx$. Then one can prove:

$$\mathbf{A}_+ \mathbf{c}_n \mathbf{c}_m =_\beta \mathbf{c}_{n+m},$$

$$\mathbf{A}_\times \mathbf{c}_n \mathbf{c}_m =_\beta \mathbf{c}_{n \times m},$$

$$\mathbf{A}_e \mathbf{c}_n \mathbf{c}_m =_\beta \mathbf{c}_{n^m} \quad (m > 0).$$



Turing completeness

A numerical function $f : \mathbb{N}^k \rightarrow \mathbb{N}$ is called λ -definable if there exists a combinator P_f such that

$$P_f \mathbf{c}_{n_1} \dots \mathbf{c}_{n_k} =_{\beta} \mathbf{c}_{f(n_1, \dots, n_k)}$$

Theorem 14 (Kleene)

Every recursive function is λ -definable.