

LMSE

Logische Methoden des Software
Engineerings

Prof. Dr. Jakob Rehof

Lehrstuhl XIV, Software Engineering

Lecture Notes: LCHI

Lectures on the
Curry-Howard Isomorphism

Morten Heine B. Sørensen
University of Copenhagen

Paweł Urzyczyn
University of Warsaw

Datei: *curry-howard-notes.pdf*

Diese Vorlesung

- ❖ Definition des Lambda Kalküls
- ❖ Beta-Reduktion
- ❖ Church-Rosser Eigenschaft (Einleitung)

Lesen und Übungen für nächste Woche

- Lesen: LCHI vom Anfang bis Abschn. 1.4 (S.8)
- Übungen
 - Exercise 1.7.4
 - Exercise 1.7.5
 - Exercise 1.7.6

Type free (untyped) lambda calculus

1.1.1. DEFINITION. Let

$$V = \{v_0, v_1, \dots\}$$

denote an infinite alphabet. The set Λ^- of *pre-terms* is the set of strings defined by the grammar:

$$\Lambda^- ::= V \mid (\Lambda^- \Lambda^-) \mid (\lambda V \Lambda^-)$$

Pre-terms

1.1.2. EXAMPLE. The following are pre-terms.

- (i) $((v_0 v_1) v_2) \in \Lambda^-$;
- (ii) $(\lambda v_0 (v_0 v_1)) \in \Lambda^-$;
- (iii) $((\lambda v_0 v_0) v_1) \in \Lambda^-$;
- (iv) $((\lambda v_0 (v_0 v_0)) (\lambda v_1 (v_1 v_1))) \in \Lambda^-$.

1.1.3. NOTATION. We use uppercase letters, e.g., K, L, M, N, P, Q, R with or without subscripts to denote arbitrary elements of Λ^- and lowercase letters, e.g., x, y, z with or without subscripts to denote arbitrary elements of V .

1.1.4. TERMINOLOGY.

- (i) A pre-term of form x (i.e., an element of V) is called a *variable*;
- (ii) A pre-term of form $(\lambda x M)$ is called an *abstraction* (over x);
- (iii) A pre-term of form $(M N)$ is called an *application* (of M to N).

Notational conventions

1.1.5. NOTATION. We use the shorthands

- (i) $(K L M)$ for $((K L) M)$;
- (ii) $(\lambda x \lambda y M)$ for $(\lambda x (\lambda y M))$;
- (iii) $(\lambda x M N)$ for $(\lambda x (M N))$;
- (iv) $(M \lambda x N)$ for $(M (\lambda x N))$.

1.1.7. NOTATION. We write $\lambda x_1 \dots x_n.M$ for $\lambda x_1 \dots \lambda x_n M$. As a special case, we write $\lambda x.M$ for $\lambda x M$.

Beispiel

1.1.9. EXAMPLE. The pre-terms in Example 1.1.2 can be written as follows, respectively:

- (i) $v_0 v_1 v_2$;
- (ii) $\lambda v_0.v_0 v_1$;
- (iii) $(\lambda v_0.v_0) v_1$;
- (iv) $(\lambda v_0.v_0 v_0) \lambda v_1.v_1 v_1$.

1.1.10. REMARK. The conventions mentioned above are used in the remainder of these notes. However, we refrain from using them—wholly or partly—when we find this more convenient. For instance, we might prefer to write $(\lambda v_0.v_0 v_0) (\lambda v_1.v_1 v_1)$ for the last term in the above example.

Free variables, closed term

1.1.11. DEFINITION. For $M \in \Lambda^-$ define the set $\text{FV}(M) \subseteq V$ of *free variables* of M as follows.

$$\begin{aligned}\text{FV}(x) &= \{x\}; \\ \text{FV}(\lambda x.P) &= \text{FV}(P) \setminus \{x\}; \\ \text{FV}(P Q) &= \text{FV}(P) \cup \text{FV}(Q).\end{aligned}$$

If $\text{FV}(M) = \{\}$ then M is called *closed*.

Beispiel

1.1.12. EXAMPLE. Let x, y, z denote distinct variables. Then

(i) $FV(x\ y\ z) = \{x, y, z\}$;

(ii) $FV(\lambda x.x\ y) = \{y\}$;

(iii) $FV((\lambda x.x\ x)\ \lambda y.y\ y) = \{\}$.

Substitution

1.1.13. DEFINITION. For $M, N \in \Lambda^-$ and $x \in V$, the *substitution of N for x in M* , written $M[x := N] \in \Lambda^-$, is defined as follows, where $x \neq y$:

$$\begin{aligned}x[x := N] &= N; \\y[x := N] &= y; \\(P Q)[x := N] &= P[x := N] Q[x := N]; \\(\lambda x.P)[x := N] &= \lambda x.P; \\(\lambda y.P)[x := N] &= \lambda y.P[x := N], && \text{if } y \notin \text{FV}(N) \text{ or } x \notin \text{FV}(P); \\(\lambda y.P)[x := N] &= \lambda z.P[y := z][x := N], && \text{if } y \in \text{FV}(N) \text{ and } x \in \text{FV}(P).\end{aligned}$$

where z is chosen as the $v_i \in V$ with minimal i such that $v_i \notin \text{FV}(P) \cup \text{FV}(N)$ in the last clause.

Beispiel

1.1.14. EXAMPLE. If x, y, z are distinct variables, then for a certain variable u :

$$((\lambda x.x\ yz)\ (\lambda y.x\ y\ z)\ (\lambda z.x\ y\ z))[x := y] = (\lambda x.x\ yz)\ (\lambda u.y\ u\ z)\ (\lambda z.y\ y\ z)$$

Alpha-equivalence (alpha-conversion)

1.1.15. DEFINITION. Let α -equivalence, written $=_\alpha$, be the smallest relation on Λ^- , such that

$$\begin{array}{ll} P =_\alpha P & \text{for all } P; \\ \lambda x.P =_\alpha \lambda y.P[x := y] & \text{if } y \notin \text{FV}(P), \end{array}$$

and closed under the rules:

$$\begin{array}{ll} P =_\alpha P' & \Rightarrow \quad \forall x \in V : \quad \lambda x.P =_\alpha \lambda x.P'; \\ P =_\alpha P' & \Rightarrow \quad \forall Z \in \Lambda^- : \quad P Z =_\alpha P' Z; \\ P =_\alpha P' & \Rightarrow \quad \forall Z \in \Lambda^- : \quad Z P =_\alpha Z P'; \\ P =_\alpha P' & \Rightarrow \quad P' =_\alpha P; \\ P =_\alpha P' \ \& \ P' =_\alpha P'' & \Rightarrow \quad P =_\alpha P''. \end{array}$$

Beispiel

1.1.16. EXAMPLE. Let x, y, z denote different variables. Then

- (i) $\lambda x.x =_{\alpha} \lambda y.y$;
- (ii) $\lambda x.x z =_{\alpha} \lambda y.y z$;
- (iii) $\lambda x.\lambda y.x y =_{\alpha} \lambda y.\lambda x.y x$;
- (iv) $\lambda x.x y \neq_{\alpha} \lambda x.x z$.

Terms (modulo alpha-equivalence)

1.1.17. DEFINITION. Define for any $M \in \Lambda^-$, the *equivalence class* $[M]_\alpha$ by:

$$[M]_\alpha = \{N \in \Lambda^- \mid M =_\alpha N\}$$

Then define the set Λ of λ -terms by:

$$\Lambda = \Lambda^- / =_\alpha = \{[M]_\alpha \mid M \in \Lambda^-\}$$

1.1.19. NOTATION. We write M instead of $[M]_\alpha$ in the remainder. This leads to ambiguity: is M a pre-term or a λ -term? In the remainder of these notes, M should always be construed as $[M]_\alpha \in \Lambda$, *except when explicitly stated otherwise*.

Free variables (modulo alpha)

1.1.20. DEFINITION. For $M \in \Lambda$ define the set $\text{FV}(M) \subseteq V$ of *free variables* of M as follows.

$$\begin{aligned}\text{FV}(x) &= \{x\}; \\ \text{FV}(\lambda x.P) &= \text{FV}(P) \setminus \{x\}; \\ \text{FV}(P Q) &= \text{FV}(P) \cup \text{FV}(Q).\end{aligned}$$

If $\text{FV}(M) = \{\}$ then M is called *closed*.

1.1.21. REMARK. According to Notation 1.1.19, what we really mean by this is that we define FV as the map from Λ to subsets of V satisfying the rules:

$$\begin{aligned}\text{FV}([x]_\alpha) &= \{x\}; \\ \text{FV}([\lambda x.P]_\alpha) &= \text{FV}([P]_\alpha) \setminus \{x\}; \\ \text{FV}([P Q]_\alpha) &= \text{FV}([P]_\alpha) \cup \text{FV}([Q]_\alpha).\end{aligned}$$

Substitution (modulo alpha)

1.1.22. DEFINITION. For $M, N \in \Lambda$ and $x \in V$, the *substitution of N for x in M* , written $M\{x := N\}$, is defined as follows:

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

1.1.23. EXAMPLE.

- (i) $(\lambda x.x y)[x := \lambda z.z] = \lambda x.x y$;
- (ii) $(\lambda x.x y)[y := \lambda z.z] = \lambda x.x \lambda z.z$.

Beta notion of reduction

1.2.1. DEFINITION. Let \rightarrow_β be the smallest relation on Λ such that

$$(\lambda x.P) Q \rightarrow_\beta P[x := Q],$$

and closed under the rules:

$$\begin{aligned} P \rightarrow_\beta P' &\Rightarrow \forall x \in V : \lambda x.P \rightarrow_\beta \lambda x.P' \\ P \rightarrow_\beta P' &\Rightarrow \forall Z \in \Lambda : P Z \rightarrow_\beta P' Z \\ P \rightarrow_\beta P' &\Rightarrow \forall Z \in \Lambda : Z P \rightarrow_\beta Z P' \end{aligned}$$

A term of form $(\lambda x.P) Q$ is called a β -*redex*, and $P[x := Q]$ is called its β -*contractum*. A term M is a β -*normal form* if there is no term N with $M \rightarrow_\beta N$.

Beta-reduction and conversion

1.2.2. DEFINITION.

- (i) The relation \twoheadrightarrow_β (*multi-step β -reduction*) is the transitive-reflexive closure of \rightarrow_β ; that is, \twoheadrightarrow_β is the smallest relation closed under the rules:

$$\begin{aligned} P \rightarrow_\beta P' &\Rightarrow P \twoheadrightarrow_\beta P'; \\ P \twoheadrightarrow_\beta P' \ \&\ \ P' \twoheadrightarrow_\beta P'' &\Rightarrow P \twoheadrightarrow_\beta P''; \\ P \twoheadrightarrow_\beta P. & \end{aligned}$$

- (ii) The relation $=_\beta$ (*β -equality*) is the transitive-reflexive-symmetric closure of \rightarrow_β ; that is, $=_\beta$ is the smallest relation closed under the rules:

$$\begin{aligned} P \rightarrow_\beta P' &\Rightarrow P =_\beta P'; \\ P =_\beta P' \ \&\ \ P' =_\beta P'' &\Rightarrow P =_\beta P''; \\ P =_\beta P; & \\ P =_\beta P' &\Rightarrow P' =_\beta P. \end{aligned}$$

Warning (many equalities!)

1.2.3. WARNING. In these notes, the symbol $=$ without any qualification is used to express the fact that two objects, e.g., pre-terms or λ -terms are identical. This symbol is very often used in the literature for β -equality.

Beispiel

1.2.4. EXAMPLE.

- (i) $(\lambda x.x x) \lambda z.z \rightarrow_{\beta} (x x)[x := \lambda z.z] = (\lambda z.z) \lambda y.y;$
- (ii) $(\lambda z.z) \lambda y.y \rightarrow_{\beta} z[z := \lambda y.y] = \lambda y.y;$
- (iii) $(\lambda x.x x) \lambda z.z \twoheadrightarrow_{\beta} \lambda y.y;$
- (iv) $(\lambda x.x) y z =_{\beta} y ((\lambda x.x) z).$

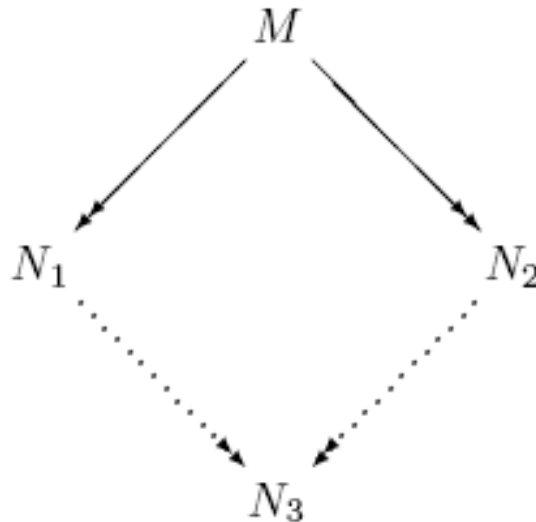
The Church-Rosser theorem (confluence)

$$\mathbf{K} (\mathbf{I} \mathbf{I}) \rightarrow_{\beta} \lambda x. (\mathbf{I} \mathbf{I})$$

$$\mathbf{K} (\mathbf{I} \mathbf{I}) \rightarrow_{\beta} \mathbf{K} \mathbf{I}$$

Church-Rosser (confluence) property

If $M \rightarrow_{\beta} N_1$, $M \rightarrow_{\beta} N_2$, then for some N_3 one has $N_1 \rightarrow_{\beta} N_3$ and $N_2 \rightarrow_{\beta} N_3$; in diagram



Frage: Warum nicht direkt durch Induktion beweisen?

Frage: Warum nicht von lokaler Konfluenz generalisieren?

Diamond property

1.4.1. DEFINITION. A relation $>$ on Λ satisfies the *diamond property* if, for all $M_1, M_2, M_3 \in \Lambda$, if $M_1 > M_2$ and $M_1 > M_3$, then there exists an $M_4 \in \Lambda$ such that $M_2 > M_4$ and $M_3 > M_4$.

Frage: Hat beta-Reduktion die Diamanteigenschaft?