LMSE 1
Logische Methoden des Software Engineerings
Vertiefungsmodul 1

Prof. Dr. Jakob Rehof

M.Sc. Andrej Dudenhefner

Lehrstuhl XIV, Software Engineering

Diese Vorlesung

- Ausdrucksmächtigkeit des Kalküls
- Fixpunktkombinatoren
- Turing-Vollständigkeit und Unentscheidbarkeit

Lesen und Übungen

- Lesen: LCHI, bis Ende Kap.1
- Übungen:
  - Übungen auf Folie 8 (Beweisen Sie (i) und (ii) auf Folie 6)
  - 1.7.14
  - 1.7.15
  - 1.7.16
  - 1.7.17
  - 1.7.18 (Hilfe: Wikipedia, „Church encodings“)
  - 1.7.20 (Hilfe: Wikipedia, „Church encodings“)
  - Schauen Sie sich auch die Liste-Kodierungen an, bei Wikipedia, „Church encodings“

Expressibility

- Church numerals (Church, Rosser)
- Booleans, pairs and other data types
- Fixed point operators
- Coding of recursive functions (lambda definability theorem, Kleene)
- Undecidability (Curry, Scott) [Rice‘s theorem for lambda calculus]

Arithmetic

2.14. Definition. (i) $F^n(M)$ with $F \in \Lambda$ and $n \in \mathbb{N}$ is defined inductively as follows.

$$\begin{aligned} F^0(M) &\equiv M; \\ F^{n+1}(M) &\equiv F(F^n(M)). \end{aligned}$$

(ii) The *Church numerals* $\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots$ are defined by

$$\boldsymbol{c}_n \equiv \lambda f x.f^n(x).$$

Arithmetic

PROPOSITION (J.B. Rosser). *Define*

$$\begin{aligned} \mathbf{A}_{+} &\equiv \lambda xypq.xp(ypq); \\ \mathbf{A}_{*} &\equiv \lambda xyz.x(yz); \\ \mathbf{A}_{\exp} &\equiv \lambda xy.yx. \end{aligned}$$

Then one has for all $n, m \in \mathbb{N}$

(i) $\mathbf{A}_{+}\boldsymbol{c}_n\boldsymbol{c}_m = \boldsymbol{c}_{n+m}$.

(ii) $\mathbf{A}_{*}\boldsymbol{c}_n\boldsymbol{c}_m = \boldsymbol{c}_{n*m}$.

(iii) $\mathbf{A}_{\exp}\boldsymbol{c}_n\boldsymbol{c}_m = \boldsymbol{c}_{(n^m)}$, *except for* $m = 0$ *(Rosser started counting from* 1*).*

LEMMA. (i) $(\boldsymbol{c}_n x)^m(y) = x^{n*m}(y)$.
(ii) $(\boldsymbol{c}_n)^m(x) = \boldsymbol{c}_{(n^m)}(x)$, *for* $m > 0$.

PROOF. (i) Induction on m. If $m = 0$, then LHS $= y =$ RHS. Assume (i) is correct for m (Induction Hypothesis: IH). Then

$$\begin{aligned}
(\boldsymbol{c}_n x)^{m+1}(y) &= \boldsymbol{c}_n x((\boldsymbol{c}_n x)^m(y)) \\
&= \boldsymbol{c}_n x(x^{n*m}(y)) \quad \text{by IH,} \\
&= x^n(x^{n*m}(y)) \\
&\equiv x^{n+n*m}(y) \\
&\equiv x^{n*(m+1)}(y).
\end{aligned}$$

(ii) Induction on $m > 0$. If $m = 1$, then LHS $\equiv \boldsymbol{c}_n x \equiv$ RHS. If (ii) is correct for m, then

$$\begin{aligned}
(\boldsymbol{c}_n)^{m+1}(x) &= \boldsymbol{c}_n((\boldsymbol{c}_n)^m(x)) \\
&= \boldsymbol{c}_n(\boldsymbol{c}_{(n^m)}(x)) \quad \text{by IH,} \\
&= \lambda y.(\boldsymbol{c}_{(n^m)}(x))^n(y) \\
&= \lambda y.x^{n^m*n}(y) \quad \text{by (i),} \\
&= \boldsymbol{c}_{(n^{m+1})}x.
\end{aligned}$$

PROOF OF THE PROPOSITION. (i) Exercise.

(ii) Exercise. Use Lemma 2.16 (i).

(iii) By Lemma 2.16 (ii) we have for $m > 0$

$$\begin{aligned} \mathsf{A}_{\exp}\boldsymbol{c}_n\boldsymbol{c}_m &= \boldsymbol{c}_m\boldsymbol{c}_n \\ &= \lambda x.(\boldsymbol{c}_n)^m(x) \\ &= \lambda x.\boldsymbol{c}_{(n^m)}x \\ &= \boldsymbol{c}_{(n^m)}, \end{aligned}$$

since $\lambda x.Mx = M$ if $M \equiv \lambda y.M'[y]$ and $x \notin \mathrm{FV}(M)$. Indeed,

$$\begin{aligned} \lambda x.Mx &\equiv \lambda x.(\lambda y.M'[y])x \\ &= \lambda x.M'[x] \\ &\equiv \lambda y.M'[y] \\ &\equiv M. \ \square \end{aligned}$$

Booleans and conditionals

1.5.6. PROPOSITION. *Define*

$$\begin{aligned}\mathbf{true} &= \lambda x.\lambda y.x;\\ \mathbf{false} &= \lambda x.\lambda y.y;\\ \mathbf{if}\ B\ \mathbf{then}\ P\ \mathbf{else}\ Q &= B\,P\,Q.\end{aligned}$$

Then

$$\begin{aligned}\mathbf{if\ true\ then}\ P\ \mathbf{else}\ Q &=_\beta P;\\ \mathbf{if\ false\ then}\ P\ \mathbf{else}\ Q &=_\beta Q.\end{aligned}$$

Pairing

1.5.7. PROPOSITION. *Define*

$$\begin{aligned} [P,Q] &= \lambda x.x\ P\ Q; \\ \pi_1 &= \lambda x.\lambda y.x; \\ \pi_2 &= \lambda x.\lambda y.y. \end{aligned}$$

Then

$$\begin{aligned} [P,Q]\ \pi_1 &=_\beta P; \\ [P,Q]\ \pi_2 &=_\beta Q. \end{aligned}$$

1.5.8. REMARK. Note that we do not have $[M\ \pi_1, M\ \pi_2] =_\beta M$ for all $M \in \Lambda$; that is, our pairing operator is not *surjective.*

1.5.9. REMARK. The construction is easily generalized to tuples $[M_1, \ldots, M_n]$ with projections π_i where $i \in \{1, \ldots, n\}$.

Fixed point theorem

1.5.10. THEOREM (Fixed point theorem). *For all F there is an X such that*

$$F\ X =_\beta X$$

In fact, there is a λ-term $\mathbf{Y}$ such that, for all F:

$$F\ (\mathbf{Y}\ F) =_\beta \mathbf{Y}\ F$$

PROOF. Put

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

Then

$$\begin{array}{rcl} \mathbf{Y}\ F & = & (\lambda f.(\lambda x.f\ (x\ x))\ \lambda x.f\ (x\ x))\ F \\ & =_\beta & (\lambda x.F\ (x\ x))\ \lambda x.F\ (x\ x) \\ & =_\beta & F\ ((\lambda x.F\ (x\ x))\ \lambda x.F\ (x\ x)) \\ & =_\beta & F\ ((\lambda f.(\lambda x.f\ (x\ x))\ \lambda x.f\ (x\ x))\ F) \\ & = & F\ (\mathbf{Y}\ F) \end{array}$$

... and more ☺

Another common fixed point combinator is
the Turing fixed-point combinator (named after its discoverer, Alan Turing):

Θ = **(λx. λy. (y (x x y))) (λx. λy. (y (x x y)))**

It also has a simple call-by-value form:

Θv = **(λx. λy. (y (λz. x x y z))) (λx. λy. (y (λz. x x y z)))**

Some fixed point combinators, such as this one (constructed by J.W.Klop)
are useful chiefly for amusement:

Yk = **(L L L L L L L L L L L L L L L L L L L L L L L L L)**

where:

L = λabcdefghijklmnopqstuvwxyzr. (r (t h i s i s a f i x e d p o i n t c o m b i n a t o r))

Wikipedia: Fixed point combinator

Lambda-definability

1.5.13. DEFINITION.

(i) A *numeric function* is a map

$$f : \mathbb{N}^m \to \mathbb{N}.$$

(ii) A numeric function $f : \mathbb{N}^m \to \mathbb{N}$ is λ-*definable* if there is an $F \in \Lambda$ such that

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

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

Recursive functions

1.5.15. DEFINITION. The class of *recursive functions* is the smallest class of numeric functions containing the *initial functions*

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

(ii) *successor:* $S^+(n) = n + 1$;

(iii) *zero:* $Z(n) = 0$.

and closed under *composition*, *primitive recursion*, and *minimization*:

Recursive functions

(i) *composition:* if $g : \mathbb{N}^k \to \mathbb{N}$ and $h_1, \ldots, h_k : \mathbb{N}^m \to \mathbb{N}$ are recursive, then so is $f : \mathbb{N}^m \to \mathbb{N}$ defined by

$$f(n_1, \ldots, n_m) = g(h_1(n_1, \ldots, n_m), \ldots, h_k(n_1, \ldots, n_m)).$$

(ii) *primitive recursion:* if $g : \mathbb{N}^m \to \mathbb{N}$ and $h : \mathbb{N}^{m+2} \to \mathbb{N}$ are recursive, then so is $f : \mathbb{N}^{m+1} \to \mathbb{N}$ defined by

$$\begin{aligned} f(0, n_1, \ldots, n_m) &= g(n_1, \ldots, n_m); \\ f(n+1, n_1, \ldots, n_m) &= h(f(n, n_1, \ldots, n_m), n, n_1, \ldots, n_m). \end{aligned}$$

(iii) *minimization:* if $g : \mathbb{N}^{m+1} \to \mathbb{N}$ is recursive and for all $n_1, \ldots, n_m$ there is an n such that $g(n, n_1, \ldots, n_m) = 0$, then $f : \mathbb{N}^m \to \mathbb{N}$ defined as follows is also recursive[3]

$$f(n_1, \ldots, n_m) = \mu n.g(n, n_1, \ldots, n_m) = 0$$

Definability of the initial functions

1.5.16. LEMMA. *The initial functions are λ-definable.*

PROOF. With

$$\begin{aligned} \mathbf{U}_i^m &= \lambda x_1 \ldots \lambda x_m . x_i \\ \mathbf{S}^+ &= \lambda x . \lambda s . \lambda z . s\ (x\ s\ z) \\ \mathbf{Z} &= \lambda x . c_0 \end{aligned}$$

the necessary properties hold.

Composition

1.5.17. LEMMA. *The λ-definable functions are closed under composition.*

PROOF. If $g : \mathbb{N}^k \to \mathbb{N}$ is λ-definable by $G \in \Lambda$ and $h_1, \ldots, h_k : \mathbb{N}^m \to \mathbb{N}$ are λ-definable by some $H_1, \ldots, H_k \in \Lambda$, then $f : \mathbb{N}^m \to \mathbb{N}$ defined by

$$f(n_1, \ldots, n_m) = g(h_1(n_1, \ldots, n_m), \ldots, h_k(n_1, \ldots, n_m))$$

is λ-definable by

$$F = \lambda x_1. \ldots \lambda x_m. G\,(H_1\, x_1 \ldots x_m) \ldots (H_k\, x_1 \ldots x_m),$$

as is easy to verify. □

Primitive recursion

1.5.18. LEMMA. *The λ-definable functions are closed under primitive recursion.*

PROOF. If $g : \mathbb{N}^m \to \mathbb{N}$ is λ-definable by some $G \in \Lambda$ and $h : \mathbb{N}^{m+2} \to \mathbb{N}$ is λ-definable by some $H \in \Lambda$, then $f : \mathbb{N}^{m+1} \to \mathbb{N}$ defined by

$$\begin{array}{lcl} f(0, n_1, \ldots, n_m) & = & g(n_1, \ldots, n_m); \\ f(n+1, n_1, \ldots, n_m) & = & h(f(n, n_1, \ldots, n_m), n, n_1, \ldots, n_m), \end{array}$$

is λ-definable by $F \in \Lambda$ where

$$\begin{array}{lcl} F & = & \lambda x.\lambda x_1 \ldots \lambda x_m . x\, T\, [c_0, G\, x_1 \ldots x_n]\, \pi_2; \\ T & = & \lambda p.[\mathbf{S}^+\, (p\, \pi_1), H\, (p\, \pi_2)\, (p\, \pi_1)\, x_1 \ldots x_m]. \end{array}$$

Minimization

1.5.19. LEMMA. *The λ-definable functions are closed under minimization.*

PROOF. If $g : \mathbb{N}^{m+1} \to \mathbb{N}$ is λ-definable by $G \in \Lambda$ and for all $n_1, \ldots, n_m$ there is an n, such that $g(n, n_1, \ldots, n_m) = 0$, then $f : \mathbb{N}^m \to \mathbb{N}$ defined by

$$f(n_1, \ldots, n_m) = \mu m.g(n, n_1, \ldots, n_m) = 0$$

is λ-definable by $F \in \Lambda$, where

$$F = \lambda x_1 \ldots . \lambda x_m . H\ c_0$$

and where $H \in \Lambda$ is such that

$$H =_\beta \lambda y.\mathbf{if}\ (\mathbf{zero?}\ (G\ x_1 \ldots x_m\ y))\ \mathbf{then}\ y\ \mathbf{else}\ H\ (\mathbf{S}^+\ y).$$

Here,

$$\mathbf{zero?} = \lambda x.x\ (\lambda y.\mathbf{false})\ \mathbf{true}$$

We leave it as an exercise to verify that the required properties hold. □

Kleene‘s theorem

The following can be seen as a form of *completeness* of the λ-calculus.

1.5.20. THEOREM (Kleene). *All recursive functions are λ-definable.*

PROOF. By the above lemmas. □

The converse also holds, as one can show by a routine argument. Similar results hold for partial functions as well—see [7].

Gödelization

1.5.21. DEFINITION. Let $\langle \bullet, \bullet \rangle : \mathbb{N}^2 \to \mathbb{N}$ be a bijective, recursive function. The map $\# : \Lambda^- \to \mathbb{N}$ is defined by:

$$\begin{array}{lcl} \#(v_i) & = & \langle 0, i \rangle \\ \#(\lambda x.M) & = & \langle 2, \langle \#(x), \#(M) \rangle \rangle \\ \#(M\ N) & = & \langle 3, \langle \#(M), \#(N) \rangle \rangle \end{array}$$

For $M \in \Lambda$, we take $\#(M)$ to be the least possible number $\#(M')$ where M' is an alpha-representative of M. Also, for $M \in \Lambda$, we define $\ulcorner M \urcorner = c_{\#(M)}$.

Recursive sets of codes

1.5.22. DEFINITION. Let $A \subseteq \Lambda$.

(i) A is *closed under* $=_\beta$ if

$$M \in A \ \& \ M =_\beta N \Rightarrow N \in A$$

(ii) A is *non-trival* if

$$A \neq \emptyset \ \& \ A \neq \Lambda$$

(iii) A is *recursive* if

$$\#A = \{\#(M) \mid M \in A\}$$

is recursive.

1.5.23. THEOREM (Curry, Scott). *Let A be non-trivial and closed under $=_\beta$. Then A is not recursive.*

PROOF (J. Terlouw). Suppose A is recursive. Define

$$B = \{M \mid M\ \lceil M \rceil \in A\}$$

There exists an $F \in \Lambda$ with

$$\begin{array}{lcl} M \in B & \Leftrightarrow & F\ \lceil M \rceil =_\beta c_0; \\ M \notin B & \Leftrightarrow & F\ \lceil M \rceil =_\beta c_1. \end{array}$$

Let $M_0 \in A$, $M_1 \in \Lambda \backslash A$, and let

$$G = \lambda x.\mathbf{if}\ (\mathbf{zero?}\ (F\ x))\ \mathbf{then}\ M_1\ \mathbf{else}\ M_0$$

Then

$$\begin{array}{lcl} M \in B & \Leftrightarrow & G\ \lceil M \rceil =_\beta M_1 \\ M \notin B & \Leftrightarrow & G\ \lceil M \rceil =_\beta M_0 \end{array}$$

so

$$\begin{array}{lcllcl} G \in B & \Leftrightarrow & G\ \lceil G \rceil =_\beta M_1 & \Rightarrow & G\ \lceil G \rceil \notin A & \Rightarrow & G \notin B \\ G \notin B & \Leftrightarrow & G\ \lceil G \rceil =_\beta M_0 & \Rightarrow & G\ \lceil G \rceil \in A & \Rightarrow & G \in B \end{array}$$

a contradiction. □

Undecidability of lambda calculus

1.5.24. REMARK. The above theorem is analogous to *Rice's theorem* known in recursion theory.

The following is a variant of the halting problem. Informally it states that the formal theory of β-equality mentioned in Remark 1.4.12 is undecidable.

1.5.25. COROLLARY (Church). $\{M \in \Lambda \mid M =_\beta \mathbf{true}\}$ *is not recursive.*

1.5.26. COROLLARY. *The following set is not recursive:*

$$\{M \in \Lambda \mid \exists N \in \Lambda : M \twoheadrightarrow_\beta N \ \&\ N \text{ is a } \beta\text{-normal form } \}.$$

One can also infer from these results the well-known theorem due to Church stating that first-order predicate calculus is undecidable.