Tutorium 06: λ-Kalkül

Paul Brinkmeier

25. November 2019

Tutorium Programmierparadigmen am KIT

Heutiges Programm

- Church-Zahlen
- Übungsblatt 5
- Altklausuraufgaben zum λ-Kalkül

Wiederholung

Ein Term im λ-Kalkül hat eine der drei folgenden Formen:

Notation	Besteht aus	Bezeichnung
x	x : Variablenname	Variable
$\lambda p.b$	p : Variablenname	Abstraktion
	b : λ-Term	
$f\ a$	f, a : λ-Terme	Funktionsanwendung

- „λ-Term": rekursive Datenstruktur

Begriffe im λ-Kalkül

Begriff	Formel	Bedeutung
α-Äquivalenz	$t_1 \overset{\alpha}{=} t_2$	t_1, t_2 sind gleicher Struktur
η-Äquivalenz	$\lambda x.f\ x \overset{\eta}{=} f$	„Unterversorgung"
Freie Variablen	$fv(\lambda p.b) = b$	Menge der nicht durch λs gebundenen Variablen
Substitution	$(\lambda p.b)\,[b \to c] = \lambda p.c$	Ersetzung nicht-freier Variablen
Redex	$(\lambda p.b)\ t$	„Reducible expression"
β-Reduktion	$(\lambda p.b)\ t \Rightarrow b\,[p \to t]$	„Funktionsanwendung"

Church-Zahlen im λ-Kalkül

Peano-Axiome

$$c_0 = \quad ?$$
$$c_1 = \quad s(c_0)$$
$$c_2 = \quad s(s(c_0))$$
$$c_3 = \quad s(s(s(c_0)))$$
$$c_8 = \quad s(s(s(s(s(s(s(s(c_0))))))))$$

1. Die 0 ist Teil der natürlichen Zahlen
2. Wenn n Teil der natürlichen Zahlen ist,
   ist auch $s(n) = n + 1$ Teil der natürlichen Zahlen

Church-Zahlen

- „Zahlen" im λ-Kalkül werden durch Funktionen in Normalform dargestellt
- n f $x = f$ n-mal angewendet auf x
- Bspw. $(3\ g\ y) = g\ (g\ (g\ y)) = g^3\ y$
  Mit $3 = \lambda f.\lambda x.f\ (f\ (f\ x))$
- Schreibt eine λ-Funktion *succ*, die eine Church-Zahl nimmt und zu deren Nachfolger auswertet

Church-Zahlen

- „Zahlen" im λ-Kalkül werden durch Funktionen in Normalform dargestellt
- $n\ f\ x = f$ n-mal angewendet auf x
- Bspw. $(3\ g\ y) = g\ (g\ (g\ y)) = g^3\ y$
  Mit $3 = \lambda f.\lambda x.f\ (f\ (f\ x))$
- Schreibt eine λ-Funktion *succ*, die eine Church-Zahl nimmt und zu deren Nachfolger auswertet
- Übertragt die Funktion in euren Haskell-Code vom letzten Mal und wertet *succ* c_0 durch wiederholtes Anwenden von `normalBeta` aus
- Vergleicht euer Ergebnis mit dem von Wavelength
  - //pp.ipd.kit.edu/lehre/misc/lambda-ide/Wavelength.html

6

Übungsblatt 5

$$c_0\ c_1\ (c_2\ c_3\ c_4)c_5 =$$

$$c_0 \; c_1 \; (c_2 \; c_3 \; c_4) c_5 = \qquad \underline{(}(c_0 \; c_1\underline{)} \; \underline{(}(c_2 \; c_3\underline{)} \; c_4)\underline{)} \; c_5 \qquad (1)$$

$$(c_0 \; c_1 \; c_2) \; (c_3 \; c_4 \; c_5) =$$

$$c_0 \, c_1 \, (c_2 \, c_3 \, c_4) c_5 = \quad ((c_0 \, c_1) \, ((c_2 \, c_3) \, c_4)) \, c_5 \tag{1}$$

$$(c_0 \, c_1 \, c_2) \, (c_3 \, c_4 \, c_5) = \quad ((c_0 \, c_1) \, c_2) \, ((c_3 \, c_4) \, c_5) \tag{2}$$

$$c_0 \, c_1 \, (c_2 \, c_3 \, c_4) \, (c_5 \, c_6) =$$

$$c_0 \, c_1 \, (c_2 \, c_3 \, c_4) c_5 = \quad \underline{(}(c_0 \, c_1\underline{)} \, (\underline{(}c_2 \, c_3\underline{)} \, c_4)\underline{)} \, c_5 \tag{1}$$

$$(c_0 \, c_1 \, c_2) \, (c_3 \, c_4 \, c_5) = \quad (\underline{(}c_0 \, c_1\underline{)} \, c_2) \, (\underline{(}c_3 \, c_4\underline{)} \, c_5) \tag{2}$$

$$c_0 \, c_1 \, (c_2 \, c_3 \, c_4) \, (c_5 \, c_6) = \quad \underline{(}(c_0 \, c_1\underline{)} \, (\underline{(}c_2 \, c_3\underline{)} \, c_4)\underline{)} \, (c_5 \, c_6) \tag{3}$$

$$c_0 \, c_1 \, (c_2 \, c_3 \, c_4) \, c_5 \, c_6 =$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4) c_5 = \quad ((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ c_5 \tag{1}$$

$$(c_0 \ c_1 \ c_2) \ (c_3 \ c_4 \ c_5) = \quad ((c_0 \ c_1) \ c_2) \ ((c_3 \ c_4) \ c_5) \tag{2}$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4) \ (c_5 \ c_6) = \quad ((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ (c_5 \ c_6) \tag{3}$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4) \ c_5 \ c_6 = \quad (((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ c_5) \ c_6 \tag{4}$$

$$c_0 \ (c_1 \ (c_2 \ c_3 \ c_4)) \ c_5 \ c_6 =$$

$$c_0\ c_1\ (c_2\ c_3\ c_4)c_5 = \underline{(}\underline{(}c_0\ c_1\underline{)}\ (\underline{(}c_2\ c_3\underline{)}\ c_4)\underline{)}\ c_5 \tag{1}$$

$$(c_0\ c_1\ c_2)\ (c_3\ c_4\ c_5) = (\underline{(}c_0\ c_1\underline{)}\ c_2)\ (\underline{(}c_3\ c_4\underline{)}\ c_5) \tag{2}$$

$$c_0\ c_1\ (c_2\ c_3\ c_4)\ (c_5\ c_6) = \underline{(}\underline{(}c_0\ c_1\underline{)}\ (\underline{(}c_2\ c_3\underline{)}\ c_4)\underline{)}\ (c_5\ c_6) \tag{3}$$

$$c_0\ c_1\ (c_2\ c_3\ c_4)\ c_5\ c_6 = \underline{(}\underline{(}\underline{(}c_0\ c_1\underline{)}\ (\underline{(}c_2\ c_3\underline{)}\ c_4)\underline{)}\ c_5\underline{)}\ c_6 \tag{4}$$

$$c_0\ (c_1\ (c_2\ c_3\ c_4))\ c_5\ c_6 = \underline{(}\underline{(}c_0\ (c_1\ (\underline{(}c_2\ c_3\underline{)}\ c_4))\underline{)}\ c_5\underline{)}\ c_6 \tag{5}$$

$$(\lambda y.c_0\ c_1\ c_2)\ (c_3\ c_4\ c_5) =$$

$$c_0 \; c_1 \; (c_2 \; c_3 \; c_4) c_5 = \qquad ((c_0 \; c_1) \; ((c_2 \; c_3) \; c_4)) \; c_5 \qquad (1)$$

$$(c_0 \; c_1 \; c_2) \; (c_3 \; c_4 \; c_5) = \qquad ((c_0 \; c_1) \; c_2) \; ((c_3 \; c_4) \; c_5) \qquad (2)$$

$$c_0 \; c_1 \; (c_2 \; c_3 \; c_4) \; (c_5 \; c_6) = \qquad ((c_0 \; c_1) \; ((c_2 \; c_3) \; c_4)) \; (c_5 \; c_6) \qquad (3)$$

$$c_0 \; c_1 \; (c_2 \; c_3 \; c_4) \; c_5 \; c_6 = \qquad (((c_0 \; c_1) \; ((c_2 \; c_3) \; c_4)) \; c_5) \; c_6 \qquad (4)$$

$$c_0 \; (c_1 \; (c_2 \; c_3 \; c_4)) \; c_5 \; c_6 = \qquad ((c_0 \; (c_1 \; ((c_2 \; c_3) \; c_4))) \; c_5) \; c_6 \qquad (5)$$

$$(\lambda y.c_0 \; c_1 \; c_2) \; (c_3 \; c_4 \; c_5) = \qquad (\lambda y.(c_0 \; c_1) \; c_2) \; ((c_3 \; c_4) \; c_5) \qquad (6)$$

$$(\lambda y.(c_0 \; (\lambda z.(c_1 \; c_2)))) \; (c_3 \; c_4 \; c_5) =$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4)c_5 = \quad ((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ c_5 \tag{1}$$

$$(c_0 \ c_1 \ c_2) \ (c_3 \ c_4 \ c_5) = \quad ((c_0 \ c_1) \ c_2) \ ((c_3 \ c_4) \ c_5) \tag{2}$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4) \ (c_5 \ c_6) = \quad ((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ (c_5 \ c_6) \tag{3}$$

$$c_0 \ c_1 \ (c_2 \ c_3 \ c_4) \ c_5 \ c_6 = \quad (((c_0 \ c_1) \ ((c_2 \ c_3) \ c_4)) \ c_5) \ c_6 \tag{4}$$

$$c_0 \ (c_1 \ (c_2 \ c_3 \ c_4)) \ c_5 \ c_6 = \quad ((c_0 \ (c_1 \ ((c_2 \ c_3) \ c_4))) \ c_5) \ c_6 \tag{5}$$

$$(\lambda y.c_0 \ c_1 \ c_2) \ (c_3 \ c_4 \ c_5) = \quad (\lambda y.(c_0 \ c_1) \ c_2) \ ((c_3 \ c_4) \ c_5) \tag{6}$$

$$(\lambda y.(c_0 \ (\lambda z.(c_1 \ c_2)))) \ (c_3 \ c_4 \ c_5) = \quad (\lambda y.(c_0 \ (\lambda z.(c_1 \ c_2)))) \ ((c_3 \ c_4) \ c_5) \tag{7}$$

- Funktionsaufrufe sind linksassoziativ, wie in Haskell
- Bzw. in Haskell sind FA linksassoz., wie im λ-Kalkül

$$(\lambda y.y)\ c_0 \overset{?}{=} \lambda y.y\ c_0 \qquad (1)$$
$$\lambda y.(y\ c_0) \overset{?}{=} \lambda y.y\ c_0 \qquad (2)$$

- Term $1 \approx$ `App (Abs "y"(Var "y")) (Var "c0")`
- Term $2 \approx$ `Abs "y"(App (Var "y") (Var "c0"))`

$$(\lambda y.y)\ c_0 \stackrel{?}{=} \lambda y.y\ c_0 \tag{1}$$

$$\lambda y.(y\ c_0) \stackrel{?}{=} \lambda y.y\ c_0 \tag{2}$$

- Term $1 \approx$ `App (Abs "y"(Var "y")) (Var "c0")`
- Term $2 \approx$ `Abs "y"(App (Var "y") (Var "c0"))`
- $\rightsquigarrow$ zweite Gleichung stimmt

$$((x)\, c_0)\, [x \rightarrow \lambda y.y] = (\lambda y.y)\, c_0 \qquad (1)$$

$$(x\, c_0)\, [x \rightarrow (\lambda y.y)] = (\lambda y.y)\, c_0 \qquad (2)$$

$$(x\, c_0)\, [x \rightarrow \lambda y.y] = (\lambda y.y)\, c_0 \qquad (3)$$

- Alle drei Substitutionen führen zum selben Ergebnis
- „Für beliebiges t repräsentieren t und (t) den gleichen λ-Term" stimmt

1.4 — Klammerung im λ-Kalkül

Angenommen, $x = c_0 \; c_1$.
Welche der folgenden Aussagen gelten?

$$c_0 \; c_1 \; c_2 = x \; c_2 \tag{1}$$

$$c_2 \; c_0 \; c_1 = c_2 \; x \tag{2}$$

$$c_2 \; (c_3 \; c_4) \; c_0 \; c_1 = c_2 \; (c_3 \; c_4) \; x \tag{3}$$

$$c_2 \; (c_0 \; c_1 \; c_3) c_4 = c_2 \; (x \; c_3) \; c_4 \tag{4}$$

1.4 — Klammerung im λ-Kalkül

Angenommen, $x = c_0\ c_1$.

Welche der folgenden Aussagen gelten?

$$c_0\ c_1\ c_2 = x\ c_2 \tag{1}$$

$$c_2\ c_0\ c_1 = c_2\ x \tag{2}$$

$$c_2\ (c_3\ c_4)\ c_0\ c_1 = c_2\ (c_3\ c_4)\ x \tag{3}$$

$$c_2\ (c_0\ c_1\ c_3)c_4 = c_2\ (x\ c_3)\ c_4 \tag{4}$$

- 1 und 4 gelten
- $c_2\ c_0\ c_1 = \underline{(c_2\ c_0)}\ c_1 \neq c_2\ \underline{(c_0\ c_1)} = c_2\ x$
- $c_2\ (c_3\ c_4)\ c_0\ c_1 \overset{*}{\neq} c_2\ (c_3\ c_4)\ x$

2 — Church-Zahlen in Haskell

```
module ChurchNumbers where

type Church t = (t -> t) -> t -> t

int2church n f t = iterate f t !! n

-- Rekursiv:
int2church' 0 f t = t
int2church' n f t = f $ int2church (n - 1) f t

church2int cn = cn (+ 1) 0
```

Klausuraufgaben zum λ-Kalkül

$$S = \lambda x.\lambda y.\lambda z.x\ z\ (y\ z)$$
$$K = \lambda x.\lambda y.x$$
$$I = \lambda x.x$$

- SKI-Kalkül kann alles, was λ-Kalkül auch kann, mit 3 „Kombinatoren"
- Definiere $U = \lambda x.x\ S\ K$
- Aufgabe: Beweise, dass man S, K und I durch U darstellen kann

$$S = \lambda x.\lambda y.\lambda z.x\ z\ (y\ z)$$
$$K = \lambda x.\lambda y.x$$
$$I = \lambda x.x$$

- SKI-Kalkül kann alles, was λ-Kalkül auch kann, mit 3 „Kombinatoren"
- Definiere $U = \lambda x.x\ S\ K$
- Aufgabe: Beweise, dass man S, K und I durch U darstellen kann
  - $U\ U\ x \stackrel{?}{\Longrightarrow} x = I\ x$
  - $U\ (U\ (U\ U)) = U\ (U\ I) \stackrel{?}{\Longrightarrow} K$
  - $U\ (U\ (U\ (U\ U))) = U\ K \stackrel{?}{\Longrightarrow} S$

$$nil = \lambda n.\lambda c.n$$
$$cons = \lambda x.\lambda xs.\lambda n.\lambda c.c \; x \; xs$$

- Schreibe *head* und *tail*, sodass:
  - *head* (*cons A B*) $\overset{*}{\Longrightarrow}$ *A*
  - *tail* (*cons A B*) $\overset{*}{\Longrightarrow}$ *B*

$$nil = \lambda n.\lambda c.n$$
$$cons = \lambda x.\lambda xs.\lambda n.\lambda c.c \; x \; xs$$

- Schreibe *head* und *tail*, sodass:
    - *head* (*cons A B*) $\overset{*}{\Longrightarrow}$ *A*
    - *tail* (*cons A B*) $\overset{*}{\Longrightarrow}$ *B*
- Schreibe *replicate*, sodass:
    - *replicate* c_n *A* = $\underbrace{cons\;A\;(cons\;A\;...\;(cons\;A\;nil)\;...)}_{n\;\text{mal}}$
    - Erinnerung: $c_n \; f \; x = \underbrace{f \; (f \; ... \; (f \; x) \; ...)}_{n\;\text{mal}}$

$$nil = \lambda n.\lambda c.n$$
$$cons = \lambda x.\lambda xs.\lambda n.\lambda c.c\ x\ xs$$

- Schreibe *head* und *tail*, sodass:
  - *head* (*cons A B*) $\stackrel{*}{\Longrightarrow}$ *A*
  - *tail* (*cons A B*) $\stackrel{*}{\Longrightarrow}$ *B*
- Schreibe *replicate*, sodass:
  - *replicate* c_n *A* = $\underbrace{cons\ A\ (cons\ A\ ...\ (cons\ A\ nil)\ ...)}_{n\ \text{mal}}$
  - Erinnerung: $c_n\ f\ x = \underbrace{f\ (f\ ...\ (f\ x)\ ...)}_{n\ \text{mal}}$
- Werte aus: *replicate* c_3 *A* $\stackrel{*}{\Longrightarrow}$?