

Tutorium 06: λ -Kalkül

Paul Brinkmeier

26. 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 b : λ -Term	Abstraktion
$f a$	f, a : λ -Terme	Funktionsanwendung

- „ λ -Term“: rekursive Datenstruktur

Begriffe im λ -Kalkül

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

Church-Zahlen im λ -Kalkül

$$\begin{aligned}c_0 &= ? \\c_1 &= s(c_0) \\c_2 &= s(s(c_0)) \\c_3 &= s(s(s(c_0))) \\c_8 &= s(s(s(s(s(s(s(s(c_0))))))))\end{aligned}$$

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
- $n\ f\ x = f\ n\text{-mal angewendet auf } x$
- 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
- $n\ f\ x = f\ n\text{-mal angewendet auf } x$
- 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
- Überträgt 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](http://pp.ipd.kit.edu/lehre/misc/lambda-ide/Wavelength.html)

Übungsblatt 5

1.1 — Klammerung im λ -Kalkül

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

1.1 — Klammerung im λ -Kalkül

$$\begin{aligned} c_0 \ c_1 \ (c_2 \ c_3 \ c_4) c_5 &= \underline{\underline{((c_0 \ c_1) ((c_2 \ c_3) c_4))}} \ c_5 & (1) \\ (c_0 \ c_1 \ c_2) \ (c_3 \ c_4 \ c_5) &= \end{aligned}$$

1.1 — Klammerung im λ -Kalkül

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

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

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

1.1 — Klammerung im λ -Kalkül

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

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

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

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

1.1 — Klammerung im λ -Kalkül

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

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

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

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

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

1.1 — Klammerung im λ -Kalkül

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

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

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

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

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

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

1.1 — Klammerung im λ -Kalkül

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

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

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

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

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

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

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

1.1 — Klammerung im λ -Kalkül

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

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

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

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

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

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

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

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

1.2 — Klammerung im λ -Kalkül

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

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

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

1.2 — Klammerung im λ -Kalkül

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

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

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

1.3 — Klammerung im λ -Kalkül

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

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

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

- 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?

$$c_0 \ c_1 \ c_2 = \quad x \ c_2 \quad (1)$$

$$c_2 \ c_0 \ c_1 = \quad c_2 \ x \quad (2)$$

$$c_2 \ (c_3 \ c_4) \ c_0 \ c_1 = \quad c_2 \ (c_3 \ c_4) \ x \quad (3)$$

$$c_2 \ (c_0 \ c_1 \ c_3) c_4 = \quad c_2 \ (x \ c_3) \ c_4 \quad (4)$$

1.4 — Klammerung im λ -Kalkül

Angenommen, $x = c_0 \ c_1$.

Welche der folgenden Aussagen gelten?

$$c_0 \ c_1 \ c_2 = \quad x \ c_2 \quad (1)$$

$$c_2 \ c_0 \ c_1 = \quad c_2 \ x \quad (2)$$

$$c_2 \ (c_3 \ c_4) \ c_0 \ c_1 = \quad c_2 \ (c_3 \ c_4) \ x \quad (3)$$

$$c_2 \ (c_0 \ c_1 \ c_3) c_4 = \quad c_2 \ (x \ c_3) \ c_4 \quad (4)$$

- 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

- $U \ U \ x \xRightarrow{?} x = I \ x$
- $U \ (U \ (U \ U)) = U \ (U \ I) \xRightarrow{?} K$
- $U \ (U \ (U \ (U \ U))) = U \ K \xRightarrow{?} S$

$$\text{nil} = \lambda n. \lambda c. n$$

$$\text{cons} = \lambda x. \lambda xs. \lambda n. \lambda c. c \ x \ xs$$

- Schreibe *head* und *tail*, sodass:

- $\text{head} (\text{cons } A \ B) \xRightarrow{*} A$
- $\text{tail} (\text{cons } A \ B) \xRightarrow{*} B$

$$nil = \lambda n. \lambda c. n$$

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

- Schreibe *head* und *tail*, sodass:

- $head (cons \ A \ B) \xRightarrow{*} A$

- $tail (cons \ A \ B) \xRightarrow{*} B$

- Schreibe *replicate*, sodass:

- $replicate \ c_n \ A = \underbrace{cons \ A \ (cons \ A \ \dots (cons \ A \ nil) \ \dots)}_{n \text{ mal}}$

- Erinnerung: $c_n \ f \ x = \underbrace{f \ (f \ \dots (f \ x) \ \dots)}_{n \text{ mal}}$

$$nil = \lambda n. \lambda c. n$$

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

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

$$\text{pair} = \lambda a. \lambda b. \lambda f. f \ a \ b$$

$$\text{fst} = \lambda x. \lambda y. x$$

$$\text{snd} = \lambda x. \lambda y. y$$

$$\text{fst} (\text{pair} \ a \ b) = a$$

$$\text{snd} (\text{pair} \ a \ b) = b$$

- Schreibe *curry* und *uncurry*, sodass:
 - $(\text{curry } f) \ a \ b = f \ (\text{pair} \ a \ b)$
 - $(\text{uncurry } g) \ (\text{pair} \ a \ b) = g \ a \ b$