Vorlesung

Theoretische Informatik II

Bernhard Beckert

Institut für Informatik

UNIVERSITÄT KOBLENZ · LANDAU

Wintersemester 2007/2008

Dank

Diese Vorlesungsmaterialien basieren zum Teil auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Christoph Kreitz (gehalten an der Universität Potsdam)

Ihnen gilt mein herzlicher Dank.

– Bernhard Beckert, Oktober 2007

Darstellung von Aussagenlogik im λ-Kalkül

Wahrheitswerte

$$\begin{aligned} \mathit{true} &=_{\text{def}} \quad \lambda x.\lambda y.\, x \\ \mathit{false} &=_{\text{def}} \quad \lambda x.\lambda y.\, y \end{aligned}$$

if-then-else

$$\mathit{if}\ C\ \mathit{then}\ U\ \mathit{else}\ V\ =_{\text{def}} \quad ???$$

Darstellung von Aussagenlogik im λ-Kalkül

Wahrheitswerte

$$\mathit{true} =_{\text{def}} \quad \lambda x.\lambda y.\, x$$
$$\mathit{false} =_{\text{def}} \quad \lambda x.\lambda y.\, y$$

if-then-else

$$\mathit{if\ C\ then\ U\ else\ V} =_{\text{def}} \quad \textbf{???}$$

Darstellung von Aussagenlogik im λ-Kalkül

Wahrheitswerte

$$\begin{aligned} \mathit{true} &=_{\text{def}} \quad \lambda x.\lambda y.\, x \\ \mathit{false} &=_{\text{def}} \quad \lambda x.\lambda y.\, y \end{aligned}$$

if-then-else

$$\mathit{if}\ C\ \mathit{then}\ U\ \mathit{else}\ V \ =_{\text{def}} \quad C\ U\ V$$

Darstellung von Aussagenlogik im λ-Kalkül

Mit true, false, if-then-else alles darstellbar

$$\begin{aligned} \neg x &\equiv \textit{if } x \textit{ then false else true} \\ x \wedge y &\equiv \textit{if } x \textit{ then } y \textit{ else false} \\ x \vee y &\equiv \textit{if } x \textit{ then true else } y \end{aligned}$$

Damit

$$\begin{aligned} \neg x &\equiv x \textit{ false true} \\ x \wedge y &\equiv x\ y \textit{ false} \\ x \vee y &\equiv x \textit{ true } y \end{aligned}$$

Darstellung von Aussagenlogik im λ-Kalkül

Mit true, false, if-then-else alles darstellbar

$$\begin{aligned} \neg x &\equiv \textit{if } x \textit{ then false else true} \\ x \wedge y &\equiv \textit{if } x \textit{ then } y \textit{ else false} \\ x \vee y &\equiv \textit{if } x \textit{ then true else } y \end{aligned}$$

Damit

$$\begin{aligned} \neg x &\equiv x \textit{ false true} \\ x \wedge y &\equiv x\ y \textit{ false} \\ x \vee y &\equiv x \textit{ true } y \end{aligned}$$

Darstellung Paaren im λ-Kalkül

Paare

$$\begin{aligned} mkpair(x,y) &=_{\text{def}} \quad \lambda b.\, b\ x\ y \\ fst(p) &=_{\text{def}} \quad p\ true \\ snd(p) &=_{\text{def}} \quad p\ false \end{aligned}$$

Ähnlich für

- Tupel
- Listen
- etc.

Darstellung Paaren im λ-Kalkül

Paare

$$
\begin{aligned}
\mathit{mkpair}(x, y) &=_{\text{def}} \quad \lambda b.\, b\ x\ y \\
\mathit{fst}(p) &=_{\text{def}} \quad p\ \mathit{true} \\
\mathit{snd}(p) &=_{\text{def}} \quad p\ \mathit{false}
\end{aligned}
$$

Ähnlich für

- Tupel
- Listen
- etc.

Darstellung Paaren im λ-Kalkül

Paare

$$
\begin{aligned}
\mathit{mkpair}(x, y) &=_{\text{def}} \quad \lambda b.\, b\ x\ y \\
\mathit{fst}(p) &=_{\text{def}} \quad p\ \mathit{true} \\
\mathit{snd}(p) &=_{\text{def}} \quad p\ \mathit{false}
\end{aligned}
$$

Ähnlich für

- Tupel
- Listen
- etc.

Darstellung natürlicher Zahlen im λ-Kalkül

Natürliche Zahlen

$$
\begin{aligned}
0 &=_{\text{def}} \quad \lambda f.\lambda x.\, x \\
1 &=_{\text{def}} \quad \lambda f.\lambda x.\, f\ x \\
2 &=_{\text{def}} \quad \lambda f.\lambda x.\, f\,(f\ x) \\
3 &=_{\text{def}} \quad \lambda f.\lambda x.\, f(\,f\,(f\ x)) \\
&\vdots \\
n &=_{\text{def}} \quad \lambda f.\lambda x.\, \underbrace{f(\ldots(f}_{n \text{ mal}}\ x)\ldots)
\end{aligned}
$$

Darstellung natürlicher Zahlen im λ-Kalkül

Operationen auf natürlichen Zahlen

$$
\begin{aligned}
\mathit{succ}(n) &=_{\text{def}} \quad \lambda g.\lambda y.\, n\ g\ (g\ y) \\
+(n,m) &=_{\text{def}} \quad n\ \mathit{succ}\ m \\
*(n,m) &=_{\text{def}} \quad n\ (m\ \mathit{succ})\ 0 \\
\mathit{iszero}(n) &=_{\text{def}} \quad n\ (\lambda b.\mathit{false})\ \mathit{true}
\end{aligned}
$$

Rekursion im λ-Kalkül

Der Y-Operator

$$Y =_{\text{def}} \quad \lambda F.(\lambda y.F(y\ y))\ (\lambda x.F(xx))$$

Y ist Fixpunkt-Operator

$$f(Y\,f) = Y\,f$$

für alle f

Rekursion im λ-Kalkül

Der Y-Operator

$$Y =_{\text{def}} \quad \lambda F.(\lambda y.F(y\ y))\ (\lambda x.F(xx))$$

Y ist Fixpunkt-Operator

$$f(Y\,f) = Y\,f$$

für alle f

Rekursion im λ-Kalkül

Beispiel

$$fak\ n =_{\text{def}} \quad (Y\ F)\ n$$

mit

$$F =_{\text{def}} \quad \lambda f.\lambda n.\ if\ iszero(n)\ then\ 1\ else\ n * (f(n-1))$$

Allgemein

Der Y-Operator erlaubt es, beliebige μ-rekursive Funktionen im λ-Kalkül zu definieren.

Rekursion im λ-Kalkül

Beispiel

$$fak\ n =_{\text{def}} \quad (Y\ F)\ n$$

mit

$$F =_{\text{def}} \quad \lambda f.\lambda n.\ if\ iszero(n)\ then\ 1\ else\ n * (f(n-1))$$

Allgemein

Der Y-Operator erlaubt es, beliebige μ-rekursive Funktionen im λ-Kalkül zu definieren.