Übungen zur "Theorie der Programmierung" - SoSe2015

Freitag 12:15-13:45, Martenstr. 3, Raum 02.133-113

Daniel Hausmann

daniel.hausmann@fau.de

Friedrich-Alexander-Universität Erlangen
Department Informatik

Lehrstuhl 8

May 25, 2015

Übung 4 - Church-Kodierung

2 Paare:

$$\begin{aligned} pair\ a\ b &= \lambda select\ .\ \ select\ \ a\ \ b \\ fst\ \ p &= p\ (\lambda x\ y\ .\ \ x) \\ snd\ p &= p\ (\lambda x\ y\ .\ \ y) \end{aligned}$$

Schreiben Sie eine Funktion *swap*, welche die beiden Komponenten eines gegebenen Paares austauscht, sodaß für alle λ-Terme s und t folgendes gilt:

$$fst\,(swap\ (pair\ \ s\ \ t)) \rightarrow^*_{\beta\delta} t$$

$$snd(swap\ (pair\ \ s\ \ t)) \rightarrow^*_{\beta\delta} s$$

Übung 4 - Church-Kodierung

3 Church-Numeral: $\lceil n \rceil := \lambda f a.\, \underbrace{f(f(f(\ldots f}_{n}\, a))))$

$$
\begin{aligned}
\mathit{zero} &= \lambda f\, a\ .\ a \\
\mathit{succ}\ n &= \lambda f\, a\ .\ f\ (n\ f\ a) \\
\mathit{one} &= \mathit{succ}\ \mathit{zero} \\
\mathit{two} &= \mathit{succ}\ \mathit{one} \\
\mathit{three} &= \mathit{succ}\ \mathit{two} \\
\mathit{four} &= \mathit{succ}\ \mathit{three}
\end{aligned}
$$

1. Zeigen Sie für alle n: $\mathit{succ}\ \lceil n \rceil \to^*_{\beta\delta} \lceil n+1 \rceil$.
2. Definieren Sie *add*, *mult*, *exp* sodaß für alle x und y gilt:

$$\mathit{add}\ \lceil x \rceil \lceil y \rceil \to^*_{\beta\delta} \lceil x+y \rceil \qquad \mathit{mult}\ \lceil x \rceil \lceil y \rceil \to^*_{\beta\delta} \lceil x \cdot y \rceil$$

$$\mathit{exp}\ \lceil x \rceil \lceil y \rceil \to^*_{\beta\delta} \lceil x^y \rceil$$

3. Definieren Sie eine Funktion *isZero*, so daß:

$$\mathit{isZero}\ \lceil 0 \rceil \to^*_{\beta\delta} \mathit{true} \qquad \mathit{isZero}\ \lceil n+1 \rceil \to^*_{\beta\delta} \mathit{false}$$

Übung 5 - Subtraktion

Dekrementieren von Church-Numeralen:

$pred\ n = fst\ (n\ (\lambda p\ .\ \ pair\ (snd\ p)\ (succ\ (snd\ p)))\ (pair\ \ zero\ \ zero))$

1. Zeigen Sie, dass $pred\ \lceil 3 \rceil \rightarrow^*_{\beta\delta} \lceil 2 \rceil$.
2. Erläutern Sie in eigenen Worten die Funktionsweise des durch *pred* implementierten Algorithmus.
3. Vervollständigen Sie die folgenden Funktionsdefinitionen:

$$
\begin{array}{lll}
sub\ n\ m = \ldots & // & n - m \\
eq\ \ n\ m = \ldots & // & n == m\ ? \\
le\ \ \ n\ m = \ldots & // & n \leq m\ ? \\
lt\ \ \ n\ m = \ldots & // & n < \ \ m\ ?
\end{array}
$$

Wiederholung aus der Vorlesung

Fixpunkt

Seien s und t λ-Terme. Term s ist Fixpunkt von t, wenn $s \rightarrow^*_{\beta\delta} t\ s$.

Fixpunktkombinator

Ein Fixpunktkombinator ist ein Term *fix* für den für alle Terme t gilt:

$$fix\ t \rightarrow_\beta t\ (fix\ t)$$

Beispiel:

$$Y = \lambda f.(\lambda x.f(xx))\lambda x.f(xx)$$

Es gilt für alle Terme t:

$$Y\ t \rightarrow_\beta t\ (Y\ t)$$

Übung 6 - Rekursive Definitionen

1. Wir betrachten die folgende rekursive Funktion:

$$fact\ \ n = \textbf{if}\ \ n \leq \lceil 1 \rceil\ \textbf{then}\ \lceil 1 \rceil\ \textbf{else}\ n * (\,fact\ \ (n - \lceil 1 \rceil))$$

   Zeigen Sie, dass $fact\ \ \lceil 4 \rceil \rightarrow^*_{\beta\delta} \lceil 24 \rceil$. Verwenden Sie, um die Derivationen abzukürzen, die Tatsache, dass $\lceil n \rceil * \lceil m \rceil \rightarrow^*_{\beta\delta} \lceil nm \rceil$ etc.

2. Schreiben Sie eine rekursive Funktion $odd\ \lceil n \rceil$, welche $true$ als Ergebnis liefert, wenn n ungerade ist, und $false$ andernfalls.

3. Schreiben Sie eine rekursive Funktion $halve$, so dass:
   $(\lceil 2 \rceil * halve\ \ \lceil n \rceil) + \textbf{if}\ odd\ \lceil n \rceil\ \textbf{then}\ \lceil 1 \rceil\ \textbf{else}\ \lceil 0 \rceil\ \ \rightarrow^*_{\beta\delta} \lceil n \rceil$.

Übung 7 - Wer braucht schon Rekursion?

Rekursive Definitionen sind komfortabel, aber nicht notwendig: Bereits im reinen λ-Kalkül ist es möglich, Fixpunkt-Kombinatoren zu definieren (wie z.B. den *Y*-Kombinator). Es sei *fix* ein beliebiger Fixpunkt-Kombinator, d.h. es gelte $\mathit{fix}\ f \rightarrow^*_{\beta\delta} f\,(\mathit{fix}\ f)$.

1. Die folgende Definition entspricht der Funktion *fact* aus Übung 6, jedoch wurde die Rekursion durch einen Fixpunkt ersetzt:
   $\mathit{fact'} = \mathit{fix}\ (\lambda \mathit{myself}\ n\ .\ \textbf{if}\ n \leq \lceil 1 \rceil\ \textbf{then}\ \lceil 1 \rceil$
   $\textbf{else}\ n * (\mathit{myself}\ (n - \lceil 1 \rceil)))$
   Zeigen Sie, dass $\mathit{fact'}\ \lceil 3 \rceil \rightarrow^*_{\beta\delta} \lceil 6 \rceil$
2. Schreiben Sie unter Verwendung von *fix* nun nicht-rekursive Versionen der übrigen Definitionen aus Übung 6 (*odd*, *halve*).