5. Übung Programmierung

Dominic Deckert

16. Mai 2017

Previously on ...

- Strukturelle Induktion
- Unifikation

Induktionsanfang

Behauptung: Für alle Typen a und $t :: \mathit{Tree}\ a$ gilt:
$\mathit{reverse}\,(\mathit{yield}\ t) = \mathit{yield}\,(\mathit{mirror}\ t)$

Induktionsanfang

Behauptung: Für alle Typen a und $t ::\ Tree\ a$ gilt:
$reverse\,(yield\ t) = yield\,(mirror\ t)$
Für $t = Leaf\ x$ mit $x ::\ a$ beliebig, aber fest gilt:

$$
\begin{aligned}
reverse\,(yield\ t) &= reverse\,(yield\,(Leaf\ x)) \\
&\overset{Z9}{=} reverse([x]) \\
&\overset{E1}{=} [x] \\
&\overset{Z9}{=} yield\,(Node\,x) \\
&\overset{Z5}{=} yield\,(mirror(Leaf\ x)) \\
&= yield\,(mirror\ t)
\end{aligned}
$$

Induktionsvoraussetzung

Sei a ein Typ und $t1, t2 :: \ Tree\, a$ beliebig aber fest so, dass gilt:
(IV1) $reverse\,(yield\; t1) = yield\,(mirror\; t1)$
(IV2) $reverse\,(yield\; t2) = yield\,(mirror\; t2)$

Induktionsschritt

Sei $x :: a$ beliebig aber fest und $t = Nodext1t2$. Dann gilt:

$$\begin{aligned}
reverse\,(yield\;t) &= reverse\,(yield\,(Node\,x\,t1\,t2)) \\
&\overset{Z8}{=} reverse\,(yield\,(t1){+}{+}yield\,(t2)) \\
&\overset{E2}{=} reverse\,(yield\,(t2)){+}{+}reverse\,(yield\,(t1)) \\
&\overset{IV1,IV2}{=} yield\,(mirror\;t2){+}{+}yield(mirror\;t1) \\
&\overset{Z8}{=} yield\,(Nodex(mirror\;t2)(mirror\;t1)) \\
&\overset{Z4}{=} yield\,(mirror\,(Node\,x\,t1\,t2)) \\
&= yield\,(mirror\;t)
\end{aligned}$$

a)

$$\begin{pmatrix} \sigma(\gamma(x_2), \sigma(\gamma(\alpha), x_3)) \\ \sigma(x_1, \sigma(\gamma(\alpha), \sigma(\alpha, x_1))) \end{pmatrix}$$

a)

$$\begin{pmatrix} \sigma(\gamma(x_2), \sigma(\gamma(\alpha), x_3)) \\ \sigma(x_1, \sigma(\gamma(\alpha), \sigma(\alpha, x_1))) \end{pmatrix}$$

$$\overset{Dek}{\Rightarrow} \left\{ \begin{pmatrix} \gamma(x_2) \\ x_1 \end{pmatrix}, \begin{pmatrix} \sigma(\gamma(\alpha), x_3) \\ \sigma(\gamma(\alpha), \sigma(\alpha, x_1)) \end{pmatrix} \right\}$$

$$\overset{Dek}{\Rightarrow} \left\{ \begin{pmatrix} \gamma(x_2) \\ x_1 \end{pmatrix}, \begin{pmatrix} \gamma(\alpha) \\ \gamma(\alpha) \end{pmatrix}, \begin{pmatrix} x_3 \\ \sigma(\alpha, x_1) \end{pmatrix} \right\}$$

$$\overset{Dek^2}{\Rightarrow} \left\{ \begin{pmatrix} \gamma(x_2) \\ x_1 \end{pmatrix}, \begin{pmatrix} x_3 \\ \sigma(\alpha, x_1) \end{pmatrix} \right\}$$

$$\overset{Vert}{\Rightarrow} \left\{ \begin{pmatrix} x_1 \\ \gamma(x_2) \end{pmatrix}, \begin{pmatrix} x_3 \\ \sigma(\alpha, x_1) \end{pmatrix} \right\}$$

$$\overset{Sub}{\Rightarrow} \left\{ \begin{pmatrix} x_1 \\ \gamma(x_2) \end{pmatrix}, \begin{pmatrix} x_3 \\ \sigma(\alpha, \gamma(x_2)) \end{pmatrix} \right\}$$

a), b)

Allgemeinster Unifikator:
$x_1 \mapsto \gamma(x_2), x_3 \mapsto \sigma(\alpha, \gamma(x_2))$

a), b)

Allgemeinster Unifikator:
$x_1 \mapsto \gamma(x_2), x_3 \mapsto \sigma(\alpha, \gamma(x_2))$

Weitere Unifikatoren (Bsp):
-$x_1 \mapsto \gamma(\alpha), x_2 \mapsto \alpha, x_3 \mapsto \sigma(\alpha, \gamma(\alpha))$
-$x_1 \mapsto \gamma(\gamma(\alpha)), x_2 \mapsto \gamma(\alpha), x_3 \mapsto \sigma(\alpha, \gamma(\gamma(\alpha)))$

Struktur

Lambda-Terme stellen Funktionen dar
Eingabewerte werden mit λx an Variable x gebunden
Verinfachungen:

- Klammern normal von links - können weggelassen werden
- $\lambda x.(\lambda y.(\lambda z.(t))) \to \lambda xyz.t$

Gebundene Variable: x, falls λx vorkommt
Freie Variable: x, falls x in einem Teilterm vorkommt,
und dieser nicht mit λx gebunden ist

Aufgabe 3 a)

t	GV(t)	FV(t)
$(\lambda x.xy)(\lambda y.y)$	$\{x, y\}$	$\{y\}$
$(\lambda x.(\lambda y.z(\lambda z.z(\lambda x.z))))$	$\{x, y, z\}$	$\{z\}$
$(\lambda x.(\lambda y.xz(yz)))(\lambda x.y(\lambda y.y))$	$\{x, y\}$	$\{y, z\}$

Reduktion

Umformungsregeln:

- α-Konversion: Ersetze λx (und alle daran gebundenen x) durch ein neues y
- β-Reduktion: (im Fall $(\lambda x.s)(t)$ und $GV(s) \cap FV(t) = \emptyset$)
  ersetze alle x in s durch t (und gib das entstehende s' zurück)

Aufgabe 3 b), 4

siehe Nachtrag