5. Übung (14.–18. Mai 2018)

Programmierung

Beachten Sie die Vorlesungsverlegung vom 11.05. auf den 18.05., 1. DS.

Aufgabe 1

```
1 data IntTree = Node Int [IntTree]
2
3 yield :: IntTree -> [Int]
4 yield (Node i []) = [i]
5 yield (Node i ts) = concat (map yield ts)
6
7 yieldProd :: IntTree -> Int
8 yieldProd (Node i []) = i
9 yieldProd (Node i ts) = product (map yieldProd ts)
```

Zeigen Sie unter Verwendung der obigen Definitionen durch strukturelle Induktion die Gültigkeit der Gleichung

$$\texttt{product (yield t)} = \texttt{yieldProd t} \tag{A}$$

für jeden Baum `t :: IntTree`. Sie dürfen dabei nutzen, dass für alle Typen `a`, `b`, positive Integer `k` > 0, Integer `i :: Int`, Funktionen `f :: a -> b`, Werte `a1, ..., ak :: a` und Listen `l1, ..., lk :: [Int]` folgende Gleichungen gelten:

$$\texttt{product [i]} = \texttt{i} \tag{L1}$$

$$\texttt{map f [a1, ..., ak]} = \texttt{[f a1, ..., f ak]} \tag{L2}$$

$$\texttt{product (concat [l1, ..., lk])} = \texttt{product [product l1, ..., product lk]} \tag{L3}$$

Aufgabe 2 (AGS 12.4.36)

Gegeben sei der λ-Term $\langle\text{pow}\rangle = (\lambda n f z.\ n\ (\lambda g x.\ g\,(g\,x))\ f\ z)$.

(a) Berechnen Sie die Normalform von $\langle\text{pow}\rangle\ \langle 2\rangle$. Sie dürfen Zwischenschritte weglassen.

(b) Welche mathematische Funktion berechnet $\langle\text{pow}\rangle$?

(c) *Zusatzaufgabe:* Verallgemeinern Sie $\langle\text{pow}\rangle$.

Aufgabe 3 (AGS 12.4.29 $\star$)

(a) Geben Sie einen Kombinator A an, so dass $A\ t\ s\ u \Rightarrow^* s$ für alle Lambdaterme t, s und u.

(b) Geben Sie einen Kombinator B an, so dass $B\ t\ s \Rightarrow^* s\ t$ für alle Lambdaterme t und s.

(c) Geben Sie einen Kombinator C an, so dass $C\ C \Rightarrow_\beta C\ C$.

(d) Geben Sie einen Kombinator D an, so dass $D \Rightarrow_\beta D$.

(e) Geben Sie einen Kombinator E an, so dass $E\ E\ t \Rightarrow^* E\ t\ E$ für jeden Lambdaterm t.

Zusatzaufgabe 1 (AGS 12.3.19)

```
1 data Tree a = Node a (Tree a) (Tree a) | Leaf a
2
3 mirror :: Tree a -> Tree a
4 mirror (Node x t1 t2) = Node x (mirror t2) (mirror t1)
5 mirror (Leaf x) = Leaf x
6
7 yield :: Tree a -> [a]
8 yield (Node _ t1 t2) = yield t1 ++ yield t2
9 yield (Leaf x) = [x]
```

Zeigen Sie unter Verwendung der obigen Definitionen durch strukturelle Induktion die Gültigkeit der Gleichung

$$\texttt{reverse (yield t)} = \texttt{yield (mirror t)} \tag{A}$$

für jeden Typ `a` und jeden Baum `t :: Tree a`. Sie dürfen dabei nutzen, dass für alle Typen `a`, `x :: a` und `xs, ys :: [a]` gilt:

$$\texttt{reverse [x]} = \texttt{[x]} \tag{E1}$$

$$\texttt{reverse (xs ++ ys)} = \texttt{reverse ys ++ reverse xs} \tag{E2}$$

Zusatzaufgabe 2 (AGS 12.3.22 ⋆)

```
1 data Tree a = Branch a (Tree a) (Tree a) | Leaf a
2
3 p :: Tree a -> [a]
4 p (Leaf x)       = [x]
5 p (Branch x s t) = [x] ++ (p s ++ p t)
6
7 d :: Tree a -> Tree a -> Tree a
8 d (Leaf x) u       = Branch x u u
9 d (Branch x s t) u = Branch x s (d t u)
```

Zeigen Sie unter Verwendung der obigen Definitionen durch strukturelle Induktion die Gültigkeit der Gleichung

$$\texttt{p (d t u)} = \texttt{p t ++ (p u ++ p u)} \tag{A}$$

für jeden Typ `a` und alle Bäume `t, u :: Tree a`. Sie dürfen dabei nutzen, dass für alle Typen `a` und alle Listen `xs, ys, zs :: [a]` gilt:

$$\texttt{xs ++ (ys ++ zs)} = \texttt{(xs ++ ys) ++ zs} \tag{B}$$

Hinweis: Für den Beweis von (A) genügt Induktion über der Struktur des Baumes `t`.