technische universität dortmund

FLDIT

Lehrstuhl 1
Logik in der Informatik

Prof. Dr. Peter Padawitz

Pascal Hof (*pascal.hof@tu-dortmund.de*)
Jens Lechner (*jens.lechner@tu-dortmund.de*)

Wintersemester 2015/2016

Übungen zu Funktionaler Programmierung
Präsenzblatt 11

Aufgabe 11.1 Markieren von Bäumen

In dieser Aufgabe ist eine Funktion `numbering :: Int -> Bintree a -> Bintree Int` zu definieren, die die k Knoten eines Baumes beginnend bei einem Wert n mit den Werten von n bis $n + k - 1$ markiert. Die Knotenmarkierungen vom Typ `a` werden dabei verworfen, wobei die Struktur des Baumes aber erhalten bleiben soll.

Der Datentyp `Bintree a` ist dabei wie folgt definiert:

```
data Bintree a = Leaf a | Branch a (Bintree a) (Bintree a)
```

In dieser Aufgabe soll die Lösung dieser Aufgabe mit der Zustandstransitionsmonade geschehen. Sie finden im EWS eine Datei `Blatt11Vorlage.hs`, die Ihnen als Vorlage dienen kann. In dem Modul finden Sie zudem eine Lösung der Aufgabe, die ohne die Transitionsmonade auskommt.

1. Zunächst wird eine Hilfsfunktion `fresh :: Trans Int Int` benötigt, die den aktuellen Zustand als Resultat zurückgibt und den Zustand inkrementiert. Definieren Sie die Funktion `fresh`, indem Sie eine Zustandstransition mit dem Konstruktor `T` erzeugen.

   Beispiele:

   ```
   runT fresh 3 ⇝ (3,4)
   runT fresh 10 ⇝ (10,11)
   ```

2. Definieren Sie nun die Funktion

   ```
   numberTree :: Tree a -> Trans Int (Bintree Int)
   ```

   zur Markierung eines Baumes als Zustandstransition, wobei in dem Zustand stets die als nächstes zu vergebene Markierung steht. Vergeben Sie die Markierungen in der Preorder-Reihenfolge.

   Natürlich sollten sie in dieser Funktion die Hilfsfunktion `fresh` aus dem ersten Aufgabenteil nutzen. Definieren Sie diese Funktionen in »=- oder `do`-Notation.

   Sie können die Funktion `numberT` aus der Vorlage zur Überprüfung Ihrer Implementierung nutzen. Für alle `t :: Bintree a` und `n :: Int` sollte folgende Bedingung gelten:

   ```
   runT (numberTree t) n == numberT t n
   ```