Universität Bremen

Praktische Informatik 3 WS 14/15

Christoph Lüth
Sandor Herms
Daniel Müller
Jan Radtke
Henrik Reichmann
Sören Schulze
Felix Thielke

6. Übungsblatt

Ausgabe: 25.11.14
Abgabe: 05.12.14

6.1 *Welcher Typ ist das?* *6 Punkte*

Leiten Sie mit dem Kontext

$$\begin{aligned} C = \{ \quad & map :: \forall \alpha\, \beta.\, (\alpha \to \beta) \to [\alpha] \to [\beta] \\ , \quad & len :: \forall \alpha.\, [\alpha] \to Int \\ , \quad & (\cdot) :: \forall \alpha\, \beta\, \gamma.\, (\beta \to \gamma) \to (\alpha \to \beta) \to \alpha \to \gamma \\ , \quad & (:) :: \forall \alpha.\alpha \to [\alpha] \to [\alpha] \\ , \quad & head :: \forall \alpha.\, [\alpha] \to \alpha \\ \} & \end{aligned}$$

die Typen folgender Terme ab:

$$C \vdash \lambda x\, y.\, head\ x : head\ y :: ? \tag{1}$$
$$C \vdash \lambda x.\, map\ len\ x :: ? \tag{2}$$
$$C \vdash \lambda f.\, len \cdot (map\ f) :: ? \tag{3}$$

Verwenden Sie die in der Vorlesung vorgestellte lineare Form der Typableitung, und führen Sie wie dort alle Schritte explizit auf.

6.2 *Huffmankodierung* *14 Punkte*

Im Laufe Ihres Studiums sollte Ihnen die Huffmankodierung[1] mindestens einmal begegnet sein. Wenn nicht, dann wird es allerhöchste Zeit.

In einer Huffmann-Kodierung werden Elemente einer Sequenz durch die Pfade in einem Häufigkeitsbaum kodiert. Indem oft vorkommende Elemente durch kürzere Pfade kodiert werden, wird die Gesamtlänge der Sequenz reduziert. Ihre Aufgabe ist es, eine Kodierung und Dekodierung von beliebigen Listen nach dem Verfahren von Huffman zu realisieren.

Als ersten Schritte schreiben Sie eine Funktion freq, welche für jedes Element einer Liste die relative Häufigkeit berechnet, und als Tupelliste zurückgibt:

```
freq :: Eq α ⇒ [α] → [(α, Double)]
```

Der Huffmanbaum für die Kodierung wird durch den Datentypen HuffTree realisiert, welcher einen binären Baum darstellt, der Elemente und ihre dazugehörige Häufigkeit in seinen Blättern speichert:

```
data HuffTree α = Node (HuffTree α) (HuffTree α) | Leaf α Double
```

Schreiben Sie nun eine Funktion fromList, welche aus einer durch freq erstellten Häufigkeitstabelle einen Huffmanbaum aufbaut, indem Sie die Liste zuerst in eine Liste von Bäumen konvertieren, und dann solange das Paar mit der geringsten Häufigkeit zusammenführen, bis nur noch eine Liste aus einem Baum übrig ist.

```
fromList :: Eq α ⇒ [(α, Double)] → HuffTree α
```

Hierfür ist eine Funktion sumTree hilfreich, welche die Summe der Häufigkeiten der Elemente in einem Baum ausgibt.

```
sumTree :: HuffTree α → Double
```

[1] http://de.wikipedia.org/wiki/Huffman-Kodierung

Nun soll mithilfe des Huffmanbaums eine gegebene Liste in eine Bitliste konvertiert werden. Schreiben Sie hierfür eine Funktion encode, welche Nothing zurückgibt, wenn die Kodierung fehlschlägt, und die kodierte Bitliste andernfalls. Verwenden Sie hierfür den Datentyp Bit aus Übungsblatt 3:

```
data Bit = I | O

encode :: Eq α ⇒ HuffTree α → [α] → Maybe [Bit]
```

Hier ist eine Hilfsfunktion tree2Table nützlich, welche aus dem Baum eine Kodierungstabelle erstellt, in der jeweils das erste Element der Tupel ein Element und das zweite die dazugehörige Kodierung darstellt.

```
tree2Table :: HuffTree α → [(α,[Bit])]
```

Für die Dekodierung implementieren Sie die Funktion decode, welche mithilfe eines Huffmanbaumes eine Bitliste dekodiert und Nothing zurückgibt, wenn die Dekodierung fehlschlägt.

```
decode :: HuffTree α → [Bit] → Maybe [α]
```

Schreiben Sie hierfür eine Hilfsfunktion followPaths, welche mithilfe einer Bitliste einen Huffmanbaum traversiert, bis ein Leaf gefunden wurde, und sowohl das gefundene Element als auch den Rest der Bitliste zurückgibt.

```
followPaths :: HuffTree α → [Bit] → (Maybe α, [Bit])
```

? *Verständnisfragen*

1. Warum ist es hilfreich, Typen abzuleiten und nicht nur die gegebene Typisierung zu überprüfen?
2. Welches sind drei charakteristische Eigenschaften von Haskells Typsystem (Hindley-Milner)?
3. Was ist ein Typschema, und wozu wird es im Hindley-Milner-Typsystem benötigt?