technische universität dortmund

FLDIT

Lehrstuhl 1
Logik in der Informatik

Prof. Dr. Peter Padawitz

Jos Kusiek (*jos.kusiek@tu-dortmund.de*)
Lukasz Czajka (*lukasz.czajka@tu-dortmund.de*)

Wintersemester 2018/2019

Übungen zu Funktionaler Programmierung
Übungsblatt 10

Ausgabe: 14.12.2018, **Abgabe:** 21.12.2018 – 16:00 Uhr, **Block:** 4

Das Übungsblatt behandelt Themen bis einschließlich Folie 175.

Aufgabe 10.1 (8 Punkte) *Gültige Gleichungen*
Zeigen Sie durch strukturelle Induktion die Korrektheit der Gleichungen auf Folie 158.

a) `foldList(listT)` = `id`

b) Für alle `alg` ∈ `List(x)(val)`,
`foldList(alg)` = `foldr (cons(alg))(nil(alg))`

Aufgabe 10.2 (6 Punkte) *Fixpunkte*
Gegeben sei folgender Datentyp für den Restklassenring $\mathbb{Z}_{10}$:

```
data Mod10 = Z0 | Z1 | Z2 | Z3 | Z4 | Z5 | Z6 | Z7 | Z8 | Z9
```

Der Datentyp ist instanziiert für die Klassen `Show`, `Eq`, `Ord`, `Read`, `Num`, `Enum` und `Bounded`. Durch die Instanzen der Klassen `Ord` und `Bounded` wird `Mod10` ein Verband und ist damit sowohl ein CPO als auch ein co-CPO.

a) Implementieren Sie die sowohl stetige und als auch co-stetige Funktion f in Haskell.

$$f : \mathbb{Z}_{10} \to \mathbb{Z}_{10}$$
$$f(x) = \begin{cases} x+1 & \text{falls } x < 5 \\ x-1 & \text{falls } x > 7 \\ x & \text{sonst} \end{cases}$$

b) Berechnen Sie den kleinsten Fixpunkt (`lfp`) von f mithilfe der Funktion `fixpt` (Folie 162).

c) Berechnen Sie den größten Fixpunkt (`gfp`) von f mithilfe der Funktion `fixpt`.

Aufgabe 10.3 (6 Punkte) *Semantik rekursiver Gleichungen*
Die Haskell-Funktion

```
fix :: (a -> a) -> a
fix f = f (fix f)
```

generiert den kleinsten Fixpunkt zu einer Funktion `f`, indem es die Funktion unendlich häufig auf sich selbst anwendet.

a) Zeigen Sie, dass die Rekursionsgleichung `length` eine Funktion definiert. Dazu wird eine Schrittfunktion Φ benötigt (analog zu Folie 163). Mithilfe der Funktion `fix` lässt sich dann der kleinste Fixpunkt für die Schrittfunktion generieren. Definieren Sie in Haskell die Schrittfunktion `phi` so, dass sich

```
lengthF :: [a] -> Int
lengthF = fix phi
```

wie die Funktion `length` verhält. (2 Punkte)

b) Beweisen Sie durch strukturelle Induktion, dass lfp(Φ) (`fix phi`) keine endliche Liste auf $\perp$ abbildet. (4 Punkte)

Aufgabe 10.4 (4 Punkte) *Graphen*
Definieren Sie folgende Haskell-Funktionen.

a) `reverseGraph :: Eq a => Graph a -> Graph a` – Dreht alle Kanten in einem Graph um.
Beispiel: `reverseGraph graph1` $\rightsquigarrow$

```
2 -> [1,6]
3 -> [1,5]
1 -> [3,4]
4 -> [3,6]
6 -> [3]
5 -> [5,6]
```

b) `isReachableFrom :: Eq a => a -> a -> Graph a -> Bool` – Gibt aus, ob der erste Knoten von dem zweiten Knoten aus erreichbar ist.
Beispiele:
`(6 `isReachableFrom` 4) graph1` $\rightsquigarrow$ `True`
`(6 `isReachableFrom` 2) graph1` $\rightsquigarrow$ `False`