Praktische Informatik 3: Einführung in die Funktionale Programmierung
Christoph Lüth · Universität Bremen, WS 2010/11

Begleitunterlagen zur Vorlesung vom 10.11.2010

Eigenschaften von funktionalen Programmen werden als Gleichungen formuliert, und durch Gleichungsumformung bewiesen. Die Gleichungsumformung wird dabei durch eine Kette von Gleichungen notiert. Als einfaches Beispiel, um die Notation einzuführen, sei folgendes Haskell-Programm gegeben

```
λbegin{code}
addTwice :: Int→ Int→ Int
addTwice x y = 2∗(x+ y)
```

Hierüber beweisen wir jetzt eine triviale Eigenschaft (ein Art Distributivregel über +):

Lemma: $\mathsf{AddTwice}\ x\ (y+z) = \mathsf{AddTwice}\ (x+y)\ z$

$$
\begin{array}{lll}
 & \mathsf{AddTwice}\ x\ (y+z) & \\
= & 2*(x+(y+z)) & \text{— Def. addTwice} \\
= & 2*((x+y)+z) & \text{— Assoziativität von } + \\
= & \mathsf{AddTwice}\ (x+y)\ z & \text{— Def. addTwice} \\
 & \square &
\end{array}
$$

Die Umformung beginnt mit der linken Seite der zu beweisenden Gleichung, und endet mit der rechten Seite. Hinter jeden Schritt schreiben wir rechts eine Rechtfertigung der Umformung von der vorhergehenden auf diese Zeile.

Fallunterscheidung. Funktionen, die über Fallunterscheidung definiert sind, erfordern Fallunterscheidung in den Beweisen. Auch hier ein Beispiel, welches die Notation einführt:

```
max, min :: Int→ Int→ Int
max x y = if x < y then y else x
min x y = if x < y then x else y
```

Lemma: $\mathsf{max}\ x\ y - \mathsf{min}\ x\ y = |x - y|$

$\mathsf{max}\ x\ y - \mathsf{min}\ x\ y$

- Fall: $x < y$

$= \quad y - \mathsf{min}\ x\ y \quad$ — Def. max

$= \quad y - x \quad$ — Def. min

$= \quad |x - y| \quad$ — Wenn $x < y$, dann $y - x = |x - y|$

- Fall: $x \geq y$

$= \quad x - \mathsf{min}\ x\ y \quad$ — Def. max

$= \quad x - y \quad$ — Def. min

$= \quad |x - y| \quad$ — Wenn $x \geq y$, dann $x - y = |x - y|$

$= \quad |x - y|$

□

Bei einer Fallunterscheidung werden die Fälle gegebenüber dem Hauptbeweis eingerückt. Die Disjunktion der Bedingungen der Fälle muss True ergeben, oder mit anderen Worten, ein Fall muss immer zutreffen (hier $x < y$ oder $x \geq y$). Jeder Fall startet eine eigene Umformungskette, die aber alle zu demselben Ergebnis führen müssen (hier $|x - y|$), danach wird der Hauptbeweis mit diesem Ergebnis fortgeführt.

Induktion. Eigenschaften rekursiv definierter Funktionen werden durch Induktion bewiesen. Wir zeigen hier die Beweise für die Behauptungen aus der Vorlesung; die Behauptungen beziehen sich auf die in der Vorlesung gezeigten Datentyp $\mathsf{MyString}$ und die darauf definierten Funktionen len, rev und cat. Der Induktionsbeweis besteht aus der Induktionsbasis und dem Induktionsschritt; beides sind vollständige Gleichungsumformungen. Es muss jeweils angegeben werden, über welcher Variablen der Behauptung die Induktion stattfindet.

(1): $\mathsf{len}\ s \geq 0$

Induktion über s

- Induktionsbasis

$\mathsf{len}\ \mathsf{Mt}$

$= 0$ — Def. len

≥ 0 — Trivial

- Induktionsschritt

$\mathsf{len}\ \mathsf{Cons}\ c\ s$

$= 1 + \mathsf{len}\ s$ — Def. len

$\geq 1 + 0$ — Induktionsannahme

$= 0$ — Grundregeln der Arithmetik

□

(2): $\mathsf{len}\ (\mathsf{cat}\ s\ t) = \mathsf{len}\ s + \mathsf{len}\ t$

Induktion über s

- Induktionsbasis

$\mathsf{len}\ (\mathsf{cat}\ \mathsf{Mt}\ t)$

$= \mathsf{len}\ t$ — Def. cat

$= 0$ — Def. len

- Induktionsschritt

$\mathsf{len}\ (\mathsf{cat}\ (\mathsf{Cons}\ c\ s)\ t)$

$= \mathsf{len}\ (\mathsf{Cons}\ c\ (\mathsf{cat}\ s\ t))$ — Def. cat

$= 1 + \mathsf{len}\ (\mathsf{cat}\ s\ t)$ — Def. len

$= 1 + (\mathsf{len}\ s + \mathsf{len}\ t)$ — Induktionsannahme

$= (1 + \mathsf{len}\ s) + \mathsf{len}\ t$ — Assoziativität von +

$= \mathsf{len}\ (\mathsf{Cons}\ c\ s) + \mathsf{len}\ t$ — Def. len

□

(3): $\mathsf{len}\ (\mathsf{rev}\ s) = \mathsf{len}\ s$

Induktion über s

- Induktionsbasis

$\mathsf{len}\ \mathsf{rev}\ \mathsf{Mt}$

$= \mathsf{len}\ \mathsf{Mt}$ — Def. rev

- Induktionsschritt

$\mathsf{len}\ (\mathsf{rev}\ (\mathsf{Cons}\ c\ s))$

$= \mathsf{len}\ (\mathsf{cat}\ (\mathsf{rev}\ s)\ (\mathsf{Cons}\ c\ \mathsf{Mt}))$ — Def. rev

$= \mathsf{len}\ (\mathsf{rev}\ s) + \mathsf{len}\ (\mathsf{Cons}\ c\ \mathsf{Mt})$ — Lemma (2)

$= \mathsf{len}\ s + (1 + \mathsf{len}\ \mathsf{Mt})$ — Induktionsannahme, Def. len

$= \mathsf{len}\ s + 1$ — Def. len , $1 + 0 = 1$

$= 1 + \mathsf{len}\ s$ — Kommutativität +

$= \mathsf{len}\ (\mathsf{Cons}\ c\ s)$ — Def. len

□

Die folgenden Beispiele kombinieren notwendigerweise Induktion und Fallunterscheidung, weil die Funktionen takeN und dropN sowohl rekursiv definiert sind als auch Fallunterscheidungen beinhalten.

(4): $n \geq 0 \longrightarrow \mathsf{len}\ (\mathsf{takeN}\ n\ s) \leq n$

Induktion über s

- Induktionsbasis

  $\mathsf{len}\ (\mathsf{takeN}\ n\ \mathsf{Mt})$

  $= \mathsf{len}\ \mathsf{Mt}$ — Def. takeN

  $= 0$ — Def. len

  $\leq n$ — nach Voraussetzung

- Induktionsschritt

  $\mathsf{len}\ (\mathsf{takeN}\ n\ (\mathsf{Cons}\ c\ s))$

  - Fall: $n = 0$

    $= \mathsf{len}\ \mathsf{Mt}$ — Def. takeN

    $= 0$ — Def. len

    $\leq n$ — nach Voraussetzung

  - Fall: $n \neq 0$

    $= \mathsf{len}\ (\mathsf{Cons}\ c\ (\mathsf{takeN}\ (n-1)\ s))$ — Def. takeN

    $= 1 + \mathsf{len}\ (\mathsf{takeN}\ (n-1)\ s)$ — Def. len

    $\leq 1 + (n-1)$ — Induktionsannahme (1)

    $= n$

  $\leq n$

□

(1) Hier muss sichergestellt sein, dass $n - 1 \geq 0$ gilt (Voraussetzung in der Induktionssannahme). Dieses ist der Fall, weil aus $n \geq 0$ (Annahme von oben) und $n \neq 0$ (Fallunterscheidung) $n > 0$ folgt, damit $n - 1 \geq 0$

(6): $\mathsf{cat}\ (\mathsf{takeN}\ n\ s)\ (\mathsf{dropN}\ n\ s) = s$

Induktion über s

- Induktionsbasis

  $\mathsf{cat}\ (\mathsf{takeN}\ n\ \mathsf{Mt})\ (\mathsf{dropN}\ n\ \mathsf{Mt})$

  $= \mathsf{cat}\ \mathsf{Mt}\ \mathsf{Mt}$ — Def. takeN , dropN

  $= \mathsf{Mt}$ — Def. cat

- Induktionsschritt

  $\mathsf{cat}\ (\mathsf{takeN}\ n\ (\mathsf{Cons}\ c\ \mathsf{Mt}))\ (\mathsf{dropN}\ n\ (\mathsf{Cons}\ c\ \mathsf{Mt}))$

  - Fall: $n = 0$

  $= \mathsf{cat}\ \mathsf{Mt}\ (\mathsf{Cons}\ c\ s)$ — Def. takeN , dropN

  $= \mathsf{Cons}\ c\ s$ — Def. cat

  - Fall: $n \neq 0$

  $= \mathsf{cat}\ (\mathsf{Cons}\ c\ (\mathsf{takeN}\ (n-1)\ s))\ (\mathsf{dropN}\ (n-1)\ s)$ — Def. takeN , dropN

  $= \mathsf{Cons}\ c\ (\mathsf{cat}\ (\mathsf{takeN}\ (n-1)\ s)\ (\mathsf{dropN}\ (n-1)\ s))$ — Def. cat

  $= \mathsf{Cons}\ c\ s$ — Induktionsannahme (1)

$= \mathsf{Cons}\ c\ s$

□

(1) Auch hier muss sichergestellt sein, dass $n-1 \geq 0$ gilt, weil sonst $\mathsf{takeN}\ (n-1)\ s$ und $\mathsf{dropN}\ (n-1)\ s$ nicht definiert sind; es folgt aus $n \geq 0$ und $n \neq 0$.