Übungen zur "Theorie der Programmierung" - SoSe2015

Freitag 12:15-13:45, Martenstr. 3, Raum 02.133-113

Daniel Hausmann

daniel.hausmann@fau.de

Friedrich-Alexander-Universität Erlangen
Department Informatik

Lehrstuhl 8

June 10, 2015

Übungsblatt 1 - Häufige Fehler

- Polynom für einige Operationen nicht angegeben
- nicht-monotone Polynome gewählt
- Domäne der polynomiellen Interpretation nicht angegeben
- Falsche kritische Paare (kritisches Paar muss aus "Überlappung" von Regeln hervorgehen)
- Kritische Paare übersehen (um das zu verhindern: systematisch alle Regel-Kombinationen durchgehen)
- Falsche Beschreibungen der Normalformen

Sei Γ Kontext, s und t λ-Terme, α Typ.

Satz (Subjekt-Reduktion)

Wenn $\Gamma \vdash t : \alpha$ und $t \to^* s$, dann $\Gamma \vdash s : \alpha$.

$IPL^{\to}$

Implikationsfragment der intuitionistischen propositionalen Logik:

$$\varphi, \psi ::= a \mid \varphi \to \psi \qquad\qquad a \in \underline{V}$$

Sequentenkalkül:

$$(\to_E)\ \frac{\Gamma \vdash \varphi \to \psi \qquad \Gamma \vdash \varphi}{\Gamma \vdash \psi} \qquad (\to_I)\ \frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \varphi \to \psi} \qquad (\mathrm{Ax})\ \frac{}{\Gamma \vdash \varphi}\ (\varphi \in \Gamma)$$

Curry-Howard-Isomorphismus

$$\vdash \varphi \Leftrightarrow \varphi \text{ inhabited}$$

Übung 11 - Programme sind Beweise?!

Problem der **type-inhabitation**: λ-Term eines gegebenen Typs α finden.
Im Folgenden bezeichnen p, q und r Typvariablen.

1. Finden Sie für jeden folgenden Typ α einen Term s mit $\vdash s : \alpha$.
   1. $p \to p$
   2. $p \to (q \to p)$
   3. $(p \to (q \to r)) \to (p \to q) \to p \to r$
   4. $((p \to q) \to r) \to q \to r$
2. Prüfen Sie (beispielsweise mittels Wahrheitstafeln), ob die Typen der vorherigen Teilaufgabe – als **aussagenlogische Formeln** interpretiert – aussagenlogische Tautologien sind. Ist das Ergebnis Ihrer Prüfung angesichts der Curry-Howard-Korrespondenz eine Überraschung?
3. Verwenden Sie die Curry-Howard-Korrespondenz, um zu zeigen, dass es keinen λ-Term *coerce* gibt, sodass $\vdash \mathit{coerce} : a \to b$ (für Typvariablen a und b).

Sei $\Sigma_{Nat} = \{0_{/0}, Suc_{/1}\}$ Signatur mit den **Konstruktoren** $0 : () \to Nat$ und $Suc : Nat \to Nat$. Weiter sei $[\![Nat]\!] = T_{\Sigma_{Nat}}(\emptyset)$.

Σ-Algebra

Σ-Algebra $\mathcal{M}$ besteht aus Trägermenge M und einer Funktion $\mathcal{M}[\![f]\!] : M^n \to M$ für jedes $f_{/n} \in \Sigma$.

$[\![Nat]\!]$ als Σ_{Nat}-Algebra: $[\![Nat]\!][\![0]\!] = 0$, $[\![Nat]\!][\![Suc]\!](t) = Suc(t)$.

Σ_{Nat}-Homomorphismus

Ein Algebra-Homomorphismus $f : \mathcal{M} \to \mathcal{N}$ zwischen zwei Σ_{Nat}-Algebren $\mathcal{M}$ und $\mathcal{N}$ ist eine Funktion $f : M \to N$ mit $f(\mathcal{M}[\![0]\!]) = \mathcal{N}[\![0]\!]$ und für alle t, $f(\mathcal{M}[\![Suc]\!](t)) = \mathcal{N}[\![Suc]\!](f(t))$.

Initiale Algebra

Σ-Algebra $\mathcal{M}$ heisst **initial**, wenn für jede Σ-Algebra $\mathcal{N}$ genau ein Σ-Homomorphismus $\mathcal{M} \to \mathcal{N}$ existiert.

Fakt

$\mathcal{M} = T_\Sigma(\emptyset)$ mit $\mathcal{M}[\![f]\!](t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$ ist initiale Σ-Algebra.

Beispiel: $[\![Nat]\!]$ ist initiale Σ_{Nat}-Algebra.

Fakt

Die initiale Σ-Algebra ist bis auf Isomorphie eindeutig.

Übung 1 - Listen natürlicher Zahlen

data NatList where

$NNil : () \rightarrow \textbf{NatList}$

$NCons : \textbf{Nat} \rightarrow \textbf{NatList} \rightarrow \textbf{NatList}$

1. Beschreiben Sie die folgenden Listen natürlicher Zahlen:
   - $NNil$
   - $NCons\ 5\ NNil$
   - $NCons\ 5\ (NCons\ 5\ NNil)$
   - $NCons\ 1\ (NCons\ 2\ (NCons\ 3\ (NCons\ 4\ NNil)))$

   $sum\ NNil = 0$

   $sum\ (NCons\ x\ xs) = x + sum\ xs$

2. 1. Welchen Typ hat *sum*?
   2. Werten Sie den Term $sum\ (NCons\ 4\ (NCons\ 89\ (NCons\ 21\ NNil)))$ aus.

3. Schreiben Sie eine Funktion $element$: **Nat** $\rightarrow$ **NatList** $\rightarrow$ **Bool**, mit $element\ a\ xs = True$ wenn a in xs vorkommt, und $element\ a\ xs = False$ sonst.

Übung 2 - Allgemeine Listen

data List a **where**

$Nil : () \rightarrow$ **List** a

$Cons : a \rightarrow$ **List** $a \rightarrow$ **List** a

1. Entscheiden Sie für jeden der folgenden Terme, ob er typisierbar ist, und geben Sie gegebenenfalls den zugehörigen Prinzipaltyp an.
   - $Cons\ True\ (Cons\ True\ Nil)$
   - $Cons\ True\ (Cons\ False\ Nil)$
   - $Cons\ True\ (Cons\ 35\ Nil)$
   - $Cons\ True$
   - $Cons\ Nil\ (Cons\ (Cons\ 35\ Nil)\ Nil)$
   - $Cons\ Nil\ (Cons\ 35\ Nil)$
   - $Cons\ Nil$

   $[a,b,c,d]$ anstelle von $Cons\ a\ (Cons\ b\ (Cons\ c\ (Cons\ d\ Nil)))$.

   Insbesondere: [] anstelle Nil.

Übung 2 - Allgemeine Listen

2. Definieren Sie induktiv die folgenden Funktionen über Listen:
   1. $length$: **List** $a \rightarrow$ **Nat**, so dass:

      $length\ [] = 0$, $length\ [a] = 1$, $length\ [a,b] = 2$, ...
   2. $snoc$: **List** $a \rightarrow a \rightarrow$ **List** a, so dass:

      $snoc\ []\ x = [x]$, $snoc\ [a]\ x = [a,x]$, $snoc\ [a,b]\ x = [a,b,x]$
   3. $reverse$: **List** $a \rightarrow$ **List** a, so dass:

      $reverse\ [] = []$, $reverse\ [a,b,c] = [c,b,a]$, ...
   4. $drop : a \rightarrow$ **List** $a \rightarrow$ **List** a, so dass:

      $drop\ x\ [] = []$, $drop\ x\ [x,y,z,x] = [y,z]$, ...
   5. $elem$: $a \rightarrow$ **List** $a \rightarrow$ **Bool**, so dass:

      $elem\ x\ [y,z,q,v] = False$, $elem\ x\ [y,z,z,q,x,z] = True$, ...
   6. $maximum$: **List Nat** $\rightarrow$ **Nat**, so dass:

      $maximum\ [] = 0$, $maximum\ [3] = 3$, $maximum\ [3,5,2,3] = 5$

Übung 3 - Beweise mittels Struktureller Induktion

Beweisen Sie die folgenden Eigenschaften jeweils durch Induktion über der Struktur der Argumentliste. Rechtfertigen Sie hierbei Ihre Schritte und geben Sie jeweils Ihre Induktions-Hypothese klar an.

1. $\forall x\ xs,\ \mathit{length}\ (\mathit{snoc}\ xs\ x) = 1 + \mathit{length}\ xs$
2. $\forall xs,\ \mathit{length}\ (\mathit{reverse}\ xs) = \mathit{length}\ xs$
3. $\forall x\ xs,\ \mathit{reverse}\ (\mathit{snoc}\ xs\ x) = \mathit{Cons}\ x\ (\mathit{reverse}\ xs)$
4. $\forall xs,\ \mathit{reverse}\ (\mathit{reverse}\ xs) = xs$

Hinweis: Wir erinnern daran, dass $s = t$ als $s =_{\beta\delta} t$ zu lesen ist. Ausserdem können Sie jederzeit zuvor bereits bewiesene Eigenschaften verwenden.