Übung zur Vorlesung

Automatisierte Logik und Programmierung

Prof. Chr. Kreitz

Universität Potsdam, Theoretische Informatik — Wintersemester 2008/09

Blatt 4 — Abgabetermin: 9.12.08 nach der Übung

Aufgabe 4.1 (Typisierung)

Geben Sie, wo möglich, eine Typisierung für die folgenden Terme an.

4.1–a $\lambda \mathtt{t}.\lambda \mathtt{y}.\ \mathtt{t\,y\,y}$

4.1–b $(\lambda \mathtt{x}.\lambda \mathtt{y}.\ \mathtt{x\,y})\ (\lambda \mathtt{z}.\ \mathtt{z})$

4.1–c $\lambda \mathtt{f}.\ (\lambda \mathtt{x}.\ \mathtt{f}\,(\mathtt{x\,x}))\ (\lambda \mathtt{x}.\ \mathtt{f}\,(\mathtt{x\,x}))$

4.1–d $(\lambda \mathtt{x}.\ \mathtt{x}^3\,\mathtt{y})\ (\lambda \mathtt{z}.\ \mathtt{g\,z\,z})$

Welches Ergebnis würde eine Anwendung des Hindley–Milner–Algorithmus auf diese Terme liefern?

4.1-e Zeigen Sie durch Induktion, daß die Church–Numerals ($\overline{n} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}$) für alle $n \in \mathbb{N}$ mit $(\mathtt{X}\to\mathtt{X})\to\mathtt{X}\to\mathtt{X}$ typisiert werden können.

Aufgabe 4.2 (Hindley–Milner Algorithmus in der Praxis)

In vielen funktionalen Programmiersprachen wie ML oder Haskell wird eine erweiterte Version des Hindley–Milner Typechecking Algorithmus eingesetzt um den Datentyp eines gegebenen Ausdrucks der Sprache zu bestimmen.

Wie müsste der Hindley–Milner Algorithmus erweitert werden, wenn neben dem einfachen Funktionenraum auch die folgenden Datenstrukturen zum Typsystem der Sprache gehören.

- Den Typ $\mathbb{B}$ der booleschen Werte zusammen mit den Elementen **T** und **F** und der Analyseoperation **if** b **then** s **else** t.
- Das Produkt $S{\times}T$ zweier Datentypen S und T zusammen mit dem Element $\langle s,t\rangle$ (Paarbildung) und der Analyseoperation **let** $\langle x,y\rangle = \langle s,t\rangle$ **in** e.
- Den Typ $\mathbb{N}$ der natürlichen Zahlen mit den Element 0, der Nachfolgeroperation $\mathsf{s}(i)$, den arithmetischen Ausdrücken $i{+}j$, $i{-}j$, $i{*}j$, i/j, i mod j, und einem induktiven Analyseoperator $\mathsf{PR}[base;h](i)$, oft geschrieben als letrec f(x) = if x=0 then $base$ else h(x,f(x-1)) in f(i).
- Den Typ T list der Listen über dem Datentyp T zusammen mit den Element [], der Operation $t{::}l$ (t wird an den Anfang der Liste l gehängt) und einem induktiven Analyseoperator $\mathsf{list\_ind}[base;h](l)$ (auch letrec f(x) = if x=[] then $base$ else let x=hd::tl in h(hd,tl,f(tl)) in f(l)).

Aufgabe 4.3 (Definitorische Erweiterung des Typsystems

Zeigen Sie, daß in der Typentheorie mit abhängigen Datentypen die folgenden Datentypen als definitorische Erweiterung erklärt werden können

4.3–a Die Summe $S{+}T$ (erzwungen disjunkte Vereinigung) zweier Datentypen S und T zusammen mit den Elementen inl (s) ("linksseitige" Einbettung eines $s \in S$) inr (t) und der Analyseoperation case e of inl $(a) \mapsto u$ | inr $(b) \mapsto v$.

4.3–b Das abhängige Produkt $x{:}S{\times}T[x]$ (vgl. Einheit 7, Folie 4)

4.3–c Ein leerer Datentyp ohne Elemente

Warum müsste ein leerer Datentyp eine Analyseoperation haben? Welchen Datentyp müsste diese sinnvollerweise haben?