Übung zur Vorlesung

Automatisierte Logik und Programmierung

Prof. Chr. Kreitz

Universität Potsdam, Theoretische Informatik — Wintersemester 2008/09

Blatt 5 — Abgabetermin: 13.01.09 nach der Übung

Achtung: Am 6. und 7. Januar 2009 finden keine Vorlesungen statt.
Die erste Veranstaltung im neuen Jahr ist die Übung am 13. Januar

Aufgabe 5.1 (Substitution in CTT)

Führen Sie die folgenden Substitutionen aus:

5.1–a $(\lambda\texttt{x.y})[\texttt{z/y}]$

5.1–b $(\lambda\texttt{x.y})[\texttt{x/y}]$

5.1–c $(\lambda\texttt{x.}\lambda\texttt{y.f s(0)})[\ \lambda\texttt{e.PR[0, n,x}\mapsto\texttt{h](e)/f}]$

Substituieren Sie zunächst die Terme werden auf Basis des üblichen Verständnisses direkt in der angegebenen Darstellung. Untersuchen Sie anschließend, wie die Substitution mit der schematischen Methode durchgeführt wird, d.h. wenn die Terme in die jeweilige Abstraktionsform umgewandelt, die allgemeinen Substitutionsregeln von Folie 19 angewandt und die Ergebnisse mit der zugehörigen Display Form ausgegeben werden.

Aufgabe 5.2 (Lazy Evaluation)

Werten Sie die folgenden Terme mit Hilfe des Lazy Evaluation Algorithmus “EVAL” aus:

5.2–a $(\lambda\texttt{x.}\lambda\texttt{y.y})\ \texttt{t}$

5.2–b $\texttt{PR[s(0), n,x}\mapsto\texttt{(s(x))](s(0))}$

Aufgabe 5.3 (Urteile semantisch analysieren)

Zeigen Sie durch Verwendung der formalen Semantikdefinition, daß gilt

5.3–a $\lambda\texttt{x.s(x)} \in \mathbb{N}\to\mathbb{N}$

5.3–b $\lambda\texttt{x.}(\lambda\texttt{y.s(y)})\texttt{(s(0))} = \lambda\texttt{z.PR[s(0), n,x}\mapsto\texttt{(s(x))](s(0))} \in \mathbb{N}\to\mathbb{N}$

Aufgabe 5.4 (Konservative Erweiterung)

Zeigen Sie, daß das Produkt $S{\times}T$ zweier Datentypen S und T als konservative Erweiterung definiert werden kann, wenn neben dem abhängigen Funktionenraum $(x{:}S{\to}T)$ auch ein boolescher Datentyp $\mathbb{B}$ (siehe Details im Anhang) zur0.7 Verfügung stehen würde.

5.4–a Gehen Sie davon aus, daß der Typ $\mathbb{B}$ mit den kanonischen Elementen **T** und **F** und dem nicht–kanonischen Element if b then s else t wie im Anhang beschrieben vordefiniert ist und definieren Sie $S{\times}T$, $\langle s,t\rangle$ und let $\langle x,y\rangle = p$ in t durch entsprechende typentheoretische Ausdrücke.

5.4–b Zeigen Sie, daß die so entstehende Semantik der Ausdrücke genau mit den bisherigen Einträgen der Redex–Kontraka–Tabelle bzw. der Semantiktabellen übereinstimmt.

5.4–c Simulieren Sie die Regel `independent_pairEquality` mit den Regeln von $\mathbb{B}$ und $x{:}S{\to}T$.

Aufgabe 5.5 (Bleibt starke Normalisierbarkeit bei Erweiterungen der einfachen Typentheorie erhalten?)

Wenn man die einfache Typentheorie um die Datentypen $\mathbb{B}$, $\mathbb{N}$, $S{\times}T$, $S{+}T$, T list erweitert, bleiben nach wie vor alle typisierbaren Terme stark normalisierbar. Erklären Sie, auf welche Art man den Beweis für die starke Normalisierbarkeit der einfachen Typentheorie erweitern muß, um dies zu zeigen?

Was würde passieren, wenn man einen leeren Datentyp hinzunimmt?

Was müssten Sie tun um zu zeigen, daß die Terme dieser erweiterten Typentheorie konfluent sind?

Anhang 5.1: Formalisierung eines Booleschen Datentyps

1. **Operatorentabelle:**

Operator und Termstruktur	*Display Form*
bool{}()	$\mathbb{B}$
true{}()	**T**
false{}()	**F**
cond{}($\boxed{b}$;s;t)	if $\boxed{b}$ then s else t

2. **Redex/Kontraktum Verhalten:**

Redex		**Kontraktum**
if **T** then s else t	$\xrightarrow{\beta}$	s
if **F** then s else t	$\xrightarrow{\beta}$	t

2 **Semantik/Urteile:**

Typsemantik
$\mathbb{B}$= $\mathbb{B}$

Elementsemantik
T= **T** $\in$ $\mathbb{B}$
F= **F** $\in$ $\mathbb{B}$
$\mathbb{B}$= $\mathbb{B}$ $\in$ $\mathbb{U}_j$ falls j eine natürliche Zahl ist

3. **Inferenzsystem/Regeln:**

$\Gamma \vdash \mathbb{B} = \mathbb{B} \in \mathbb{U}_j$ ⌊Ax⌋
by `boolEq`

$\Gamma \vdash$ **T** = **T** $\in \mathbb{B}$ ⌊Ax⌋
by `trueEq`

$\Gamma \vdash \mathbb{B}$ ⌊**ext T**⌋
by `trueI`

$\Gamma \vdash$ **F** = **F** $\in \mathbb{B}$ ⌊Ax⌋
by `falseEq`

$\Gamma \vdash \mathbb{B}$ ⌊**ext F**⌋
by `falseI`

$\Gamma \vdash$ if b_1 then s_1 else t_1
  = if b_2 then s_2 else t_2 $\in C[b_1/z]$ ⌊Ax⌋
by `condEq` z C $\mathbb{B}$
  $\Gamma \vdash b_1 = b_2 \in \mathbb{B}$ ⌊Ax⌋
  Γ, y: b_1=**T** $\in \mathbb{B} \vdash s_1 = s_2 \in C[$**T**$/z]$ ⌊Ax⌋
  Γ, y: b_1=**F** $\in \mathbb{B} \vdash t_1 = t_2 \in C[$**F**$/z]$ ⌊Ax⌋

Γ, z:$\mathbb{B}$, $\Delta \vdash C$ ⌊**ext** if z then s else t⌋
by `boolE` i
  Γ, z:$\mathbb{B}$, $\Delta[$**T**$/z] \vdash C[$**T**$/z]$ ⌊**ext** s⌋
  Γ, z:$\mathbb{B}$, $\Delta[$**F**$/z] \vdash C[$**F**$/z]$ ⌊**ext** t⌋

$\Gamma \vdash$ if **T** then s else t = $s_1 \in T$ ⌊Ax⌋
by `condRedT`
  $\Gamma \vdash s = s_1 \in T$ ⌊Ax⌋

$\Gamma \vdash$ if **F** then s else t = $t_1 \in T$ ⌊Ax⌋
by `condRedF`
  $\Gamma \vdash t = t_1 \in T$ ⌊Ax⌋