Übung zur Vorlesung

Automatisierte Logik und Programmierung I

Prof. Chr. Kreitz

Universität Potsdam, Theoretische Informatik — Wintersemester 2016/17

Blatt 6 — Abgabetermin: 26.01.2017

Das sechste Übungsblatt soll dazu dienen, sich mit formalen Beweisen in der Typentheorie und Programmentwicklung durch Beweisführung auseinanderzusetzen.
Wir werden mögliche Lösungen in der Veranstaltung am 26.01.2017 besprechen.

Aufgabe 6.1 (Für Tüftler: Lehre der Leere)

Zeigen Sie, das sich der Datentyp $\{\}$ konservativ durch den Ausdruck $\mathsf{T} = \mathsf{F} \in \mathbb{B}$ simulieren läßt. Dabei sei $\mathbb{B}$ inklusive seiner kanonischen und nicht–kanonischen Elemente wie in Übung 5.3-a definiert.

Zeigen Sie die Korrektheit der Simulation, indem Sie die Inferenzregeln für $\{\}$ durch Taktiken beschreiben, welche die Definition $\{\} \equiv \mathsf{T} = \mathsf{F} \in \mathbb{B}$ auffalten und die gleichen Unterziele generieren wie die ursprünglichen Regeln. Betrachten Sie zunächst nur die Regeln `voidEquality` und `voidElimination`. Welche Extraktterme werden dabei erzeugt? Wie könnte man den Term `any(x)` simulieren?

Aufgabe 6.2 (Programmsynthese: Bestimmung des Minimums zweier Zahlen)

Gegeben sei die Spezifikation: $\forall \mathtt{i,j}{:}\mathbb{Z}.\exists \mathtt{min}{:}\mathbb{Z}.\ \mathtt{min} \le \mathtt{i} \wedge \mathtt{min} \le \mathtt{j} \wedge (\mathtt{min{=}i} \in \mathbb{Z} \vee \mathtt{min{=}j} \in \mathbb{Z})$.

Beweisen Sie dieses Spezifikationstheorem und extrahieren Sie das zugehörige Programm.

Sie müssen im Beweis eine Fallanalyse $\mathtt{i{<}j} \vee \mathtt{i} \ge \mathtt{j}$ *durchführen. Die Disjunktion kann mit der Regel* `cut` *eingefügt und mit* `arith` *bewiesen werden. Letztere erzeugt passend zur Struktur der Disjunktion und der Definition* $x \ge y \equiv \neg(y{<}x)$ *den Extrakt-Term* $\mathsf{if}\ \mathtt{i{<}j}\ \mathsf{then}\ \mathsf{inl}(\mathsf{Ax})\ \mathsf{else}\ \mathsf{inr}(\lambda \mathtt{z}.\mathsf{Ax})$.
Beweisziele mit Größenvergleichen zwischen `i` *und* `j` *können ebenfalls mit* `arith` *bewiesen werden, wobei zur Vereinfachung angenommen werden kann, daß als Extrakt-Term* Ax *entsteht.*

Aufgabe 6.3 (Programmsynthese: Berechnung von $\lfloor \log_k n \rfloor$)

Synthetisieren Sie ein Programm zur Berechnung von ganzzahligen Logarithmen.

Geben Sie dazu eine formale (induktive) Definition der Potenz n^k an, formulieren Sie das erforderliche Spezifikationstheorem, skizzieren Sie einen Beweis, extrahieren Sie daraus den Algorithmus für $\lfloor \log_k n \rfloor$ und schätzen Sie seine Komplexität ab. Falls erforderlich, formulieren Sie ein Induktionslemma.