Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Clemens Ballarin

Vorlesung Semantik
Sommersemester 2005
Übungsblatt 3
Besprechung am 6. Mai 2005

Aufgabe 1 (H) (*Regelinduktion*)

Zeigen Sie mit Regelinduktion, dass ein Programm s, das den Anfangszustand σ in den Endzustand σ' überführt, also terminiert, nicht von der Form

$$\textbf{while}\ \text{True}\ \textbf{do}\ c$$

für ein beliebiges Programmstück c sein kann.

Hinweis: Formulieren Sie zunächst die per Regelinduktion zu zeigende Aussage!

Aufgabe 2 (H) (*Regelinduktion*)

Permutationen von Listen (über einem beliebigen Elementtyp) werden durch die folgenden vier Regeln definiert.

$$([],[]) \in \textbf{Perm} \quad \text{(Nil)} \qquad (x\#y\#l, y\#x\#l) \in \textbf{Perm} \quad \text{(Swap)}$$

$$\frac{(xs,ys) \in \textbf{Perm}}{(z\#xs, z\#ys) \in \textbf{Perm}}\ \text{(Cons)} \qquad \frac{(xs,ys) \in \textbf{Perm} \qquad (ys,zs) \in \textbf{Perm}}{(xs,zs) \in \textbf{Perm}}\ \text{(Trans)}$$

Dabei enthält die definierte Menge **Perm** Paare von Listen, die sich nur durch die Reihenfolge der Elemente unterscheiden. Mit # wird der *Cons*-Operator bezeichnet, d.h. $x\#xs$ ist die Liste bestehend aus dem ersten Element x und der Restliste xs. Formulieren Sie die dazugehörige Induktionsregel und beweisen Sie damit die folgenden Aussagen:

a) Für $(xs, ys) \in \textbf{Perm}$ gilt: xs und ys haben die gleiche Länge.

b) Für $(xs, ys) \in \textbf{Perm}$ gilt: xs und ys enthalten die gleichen Elemente.

Aufgabe 3 (Ü) (*Äquivalenz von Big-Step- und Small-Step-Semantik*)

Für die Sprache WHILE wurde in der Vorlesung eine alternative operationelle Semantik definiert. Hier soll nun eine Richtung der Äquivalenz von Big-Step-Semantik (Auswertungssemantik) $\rightarrow$ und Small-Step-Semantik (Transitionssemantik) $\rightarrow_1$ gezeigt werden. Beweisen Sie:

$$\forall s, \sigma, \sigma'.\, [\langle s, \sigma\rangle \rightarrow \sigma' \Rightarrow \langle s, \sigma\rangle \rightarrow_1^* \sigma']$$

Aufgabe 4 (P) (*Small-Step-Semantik in PROLOG*)

a) Neben *while* und *for* gibt es in imperativen Programmiersprachen oft auch ein *repeat*-Konstrukt, bei dem die Abbruchbedingung erst nach Ausführung des Schleifenkörpers getestet wird.

Die Syntax der Sprache WHILE aus der Vorlesung sei nun um

repeat s **until** b

erweitert.

Geben Sie die entsprechenden Regeln für die Small-Step-Semantik an. Die neuen Regeln sollen dabei nicht auf das **while** der Sprache zurückgreifen.

b) Die Small-Step-Semantik der erweiterten WHILE-Sprache soll in PROLOG programmiert werden. Es gelten die gleichen Vereinfachungsregeln für die abstrakte Syntax wie in Aufgabe 5, Blatt 2. Geben Sie Ihr Programm und ein Protokoll der Beispiele in der Datei `dialog.txt` ab.

Einen Rahmen zur Lösung dieser Programmieraufgabe finden Sie im Verzeichnis

`/usr/proj/semantik/prog/prolog/small-step/`