Max Wisniewski , Alexander Steen

Tutor: Ansgar Schneider

Aufgabe 1 *Fortsetzungssemantik*

Schreiben Sie ein WHILE-Programm C zur Berechnung des Quotienten zweier ganzer Zahlen. Beweisen Sie, dass die Fortsetzungssemantik $\mathcal{P}[C] < 3, 2 >=< 1 >$ gilt.

Beweis:
Unser Programm sieht wie folgt aus $C \equiv output(read \div read)$

Nun beweisen wir, dass dieses Programm die Spezifikation erfüllt:

$$
\begin{aligned}
& \mathcal{P}[output(read \div read)] < 3, 2 > \\
= \; & \mathcal{C}[output(read \div read)]id \star \pi_3(s_0, < 3, 2 >, \varepsilon) \\
= \; & \mathcal{T}[read \div read](\lambda n(s, e, a).id(s, e, a.n)) \star \pi_3(s_0, < 3, 2 >, \varepsilon) \\
= \; & \mathcal{T}[read](\lambda t_1.\mathcal{T}[read](\lambda t_2.(\lambda n(s, e, a).(s, e, a.n))(t_1 \div t_2))) \star \pi_3(s_0, < 3, 2 >, \varepsilon) \\
= \; & \mathcal{T}[read](\lambda t_1.\mathcal{T}[read](\lambda t_2(\lambda(s, e, a).(s, e, a.(t_1 \div t_2)))))(s_0, < 3, 2 >, \varepsilon) \\
= \; & (\lambda t_1.\mathcal{T}[read](\lambda t_2.(\lambda(s, e, a).(s, e, a.(t_1 \div t_2))))) \star \pi_3 3(s_0, < 2 >, \varepsilon) \\
= \; & \mathcal{T}[read](\lambda t_2.(\lambda(s, e, a).(s, e, a.(3 \div t_2)))) \star \pi_3(s_0, < 2 >, \varepsilon) \\
= \; & (\lambda t_2.(\lambda(s, e, a).(s, e, a.(3 \div t_2)))) \star \pi_3 2(s_0, \varepsilon, \varepsilon) \\
= \; & (\lambda(s, e, a).(s, e, a.(3 \div 2))) \star \pi_3(s_0, \varepsilon, \varepsilon) \\
= \; & \pi_3(s_0, \varepsilon, \varepsilon.(3 \div 2)) \\
= \; & \pi_3(s_0, \varepsilon, < 1 >) \\
= \; & < 1 >
\end{aligned}
$$

Damit haben wir die Spezifikation bewiesen und unser Programm stimmt daher.

Aufgabe 2 *FOR-Schleifen*

Erweitern Sie die Sprache WHILE um FOR - Schleifen. Erklären Sie den deren Fortsetzungssemantik.

Lösung:
Zunächst erweitern wir C um das Kontrollkonstrukt FOR I:=T_1 TO T_2 DO C.

Die Semantik sieht wie folgt aus:
$\mathcal{C}[FOR\, I := T_1\, TO\, T_2\, DO\, C] :=$
$\mathcal{T}[T_1](\lambda s.\mathcal{T}[T_2](\lambda e.\mathcal{C}[I := s]\mathcal{B}[s < e]\text{cond} < \mathcal{C}[C] \circ \mathcal{C}[FOR\, I := s + 1\, TO\, e\, DOC], \lambda z.z >))$
Wir werten also zunächst T_1 aus und geben den Wert in unsere restliche Funktion. Nach dem ersten Auswerten ist dieser Wert s nur noch inkrementiert und nicht mehr berechnet. Als nächstes Werten wir das Ende aus, dass in jeder Iteration auch fix bleibt. Danach prüfen wir jedes mal, ob $s < e$ gilt, wenn es gilt, führen wir das ganze noch einmal aus und wenn nicht, dann geben wir den Zustand einfach zurück.

Aufgabe 3 $\star$ *Operator*

Erläutern Sie, warum der $\star$ Operator in der Fortsetzungssemantik kaum noch vor kommt.

Lösung:
Dies liegt daran, dass die Fortsetzungssemantik schon das ganze als erweiterte Konkatenation auswertet. Bei $[C_1; C_2]$ wird zunächst C_1 ausgewertet und danach C_2 auf das Ergebnis angewandt. Das schöne daran ist, das die Fortsetzung C_2 gar nicht ausführen wird, wann in C_1 schon ein Fehler aufgetreten ist. Daher ist der Sternoperator schon durch die Fortsetzungssemantik nativ gegeben.

Aufgabe 4 *Fortsetzungssemantik II*

Jemand hat bei der Definition der Fortsetzungssemantik einen Fehler gemacht:

$$\mathcal{P}[C]e = \mathcal{C}[C]((\lambda z.z) \star \pi_3) < s_0, e, \varepsilon >$$

Wo liegt der Fehler genau?

Lösung:
Der Fehler besteht darin, dass der Ausdruck falsch geklammert ist.
$((\lambda z.z) \star \pi_3)$ wird dafür sorgen, dass der Zustand aus der Eingabe einfach genommen wird und sofort danach die Ausgabe ε heraus getrennt wird. Die Fortsetzungssemantik bekommt also einen Falschen Typ geliefert, da $\mathcal{C}[C]$ einen Zustand erwartet, aber nur die Ausgabe bekommt.

Klammert man es indes anders herum, ist alles korrekt. Da das λ Kalkül ohnehin linksassoziativ geklammert ist, aber leider auch unnötig.