Material zur Vorlesung "Semantik von Programmiersprachen"

Prof. Gert Smolka und Martin Müller

1 Eine idealisierte funktionale Programmiersprache

Wir definieren die idealisierte funktionale Programmiersprache F zunächst ungetypt. Im weiteren Verlauf der Vorlesung werden wir verschiedene getypte und ungetypte Teilsprachen betrachten.

Datentypen

$$\mathbf{int}, \mathbf{bool}, \mathbf{unit}, \tau_1 \times \tau_2 \text{ (Produkt) }, \tau_1 + \tau_2 \text{ (Summe) }, \tau_1 \to \tau_2 \text{ (Pfeil)}$$

Terme. Wir nehmen eine unendliche Menge **Var** von Variablen an. Variablen werden als x, y, z geschrieben. Die Menge **Ter** der Terme von F ist durch die folgende Grammatik definiert.

$$\begin{array}{rcll}
t & ::= & 0 \mid\sim 1 \mid 1 \mid\sim 2 \mid 2 \mid \dots \mid t_1 + t_2 \mid t_1 - t_2 \mid t_1 * t_2 & \text{(int)} \\
& \mid & \mathbf{true} \mid \mathbf{false} \mid t_1 \leq t_2 \mid \mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 & \text{(bool)} \\
& \mid & () & \text{(unit)} \\
& \mid & t_1, t_2 \mid \mathbf{fst}\ t \mid \mathbf{snd}\ t & (\times) \\
& \mid & \mathbf{inl}\ t \mid \mathbf{inr}\ t \mid \mathbf{case}\ t\ \mathbf{of}\ \mathbf{inl}\ x_1.t_1\ []\ \mathbf{inr}\ x_2.t_2 & (+) \\
& \mid & \lambda x.t \mid \mathbf{rec}\ x_1.\lambda x_2.t \mid x \mid t_1\ t_2 & (\to)
\end{array}$$

Alle *Variablenbinder* sind als $x.t$ geschrieben und binden x mit Skopus t. Die Menge der geschlossenen Terme von F wird als **GTer** geschrieben.

Operationale Semantik

Kanonische Terme. Die Menge **CTer** der kanonischen Terme von F ist durch die Grammatik

$$c \quad ::= \quad 0 \mid\sim 1 \mid 1 \mid\sim 2 \mid 2 \mid \dots \mid \mathbf{true} \mid \mathbf{false} \mid () \mid c_1, c_2 \mid \mathbf{inl}\ c \mid \mathbf{inr}\ c \mid \lambda x.t$$

definiert, wobei $\lambda x.t$ nur dann ein kanonischer Term ist, falls $\lambda x.t$ geschlossen ist.

Evaluierung.

1. Geschlossene Terme können zu kanonischen Termen evaluiert werden.

2. Kanonische Terme werden zu sich selbst evaluiert.

3. Evaluierung kann divergieren oder an einem Typfehler scheitern.

4. Evaluierung ist eine binäre Relation $\to\ \subseteq \mathbf{GTer} \times \mathbf{CTer}$ auf geschlossenen Termen (Ergebnissemantik).

Regelname	Regel	Analoge Bildung
(kan)	$c \to c$	
(+)	$\dfrac{t_1 \to n_1 \qquad t_2 \to n_2}{t_1 + t_2 \to n}\ \ n = n_1 + n_2$	$(*, -)$
($\leq$-false)	$\dfrac{t_1 \to n_1 \qquad t_2 \to n_2}{t_1 \leq t_2 \to \mathbf{false}}\ \ n_1 > n_2$	($\leq$-true)
(if-true)	$\dfrac{t_1 \to \mathbf{true} \qquad t_2 \to c}{\mathbf{if}\ t_1\ \mathbf{then}\ t_2\ \mathbf{else}\ t_3 \to c}$	(if-false)
(,)	$\dfrac{t_1 \to c_1 \qquad t_2 \to c_2}{t_1, t_2 \to c_2, c_2}$	
(fst)	$\dfrac{t \to c_1, c_2}{\mathbf{fst}\ t \to c_1}$	(snd)
(inl)	$\dfrac{t \to c}{\mathbf{inl}\ t \to \mathbf{inl}\ c}$	(inr)
(case-l)	$\dfrac{t \to \mathbf{inl}\ c \qquad t_1[c/x_1] \to c'}{\mathbf{case}\ t\ \mathbf{of}\ \mathbf{inl}\ x_1.t_1\ []\ \mathbf{inr}\ x_2.t_2 \to c'}$	(case-r)
(appl)	$\dfrac{t_1 \to \lambda x.t \qquad t_2 \to c_2 \qquad t[c_2/x] \to c}{t_1\ t_2 \to c}$	
(rec)	$\dfrac{}{\mathbf{rec}\ x.t \to t[\mathbf{rec}\ x.t/x]}$	

2 Die Teilsprache $\mathbf{F}_0$

F_0 ist die Teilsprache von F, die dem call-by-value λ-Kalkül entspricht.

Terme. Die Teilsprache F_0 von F ist durch die Grammatik

$$t \ ::= \ x \mid \lambda x.t \mid t_1\ t_2$$

gegeben. Mit $\mathbf{Ter}_0$ bezeichnen wir die Menge der Terme von F_0. Die Menge der geschlossenen (bzw. kanonischen) Terme von F_0 bezeichnen wir mit $\mathbf{GTer}_0$ (bzw. $\mathbf{CTer}_0$).

Operationale Semantik. Für F_0 definieren wir zusätzlich zu der von F induzierten Ergebnissemantik $\to\ \subseteq\ \mathbf{GTer}_0 \times \mathbf{CTer}_0$ eine sogenannte Einschrittsemantik $\to_1\ \subseteq \mathbf{GTer}_0 \times \mathbf{GTer}_0$ durch die folgenden Regeln.

$$(\lambda x.t_1)\ t_2 \quad \to_1 \quad t_1[t_2/x] \qquad\qquad \text{falls } t_2 \text{ kanonisch}$$

$$\frac{t_1 \to_1 t_1'}{t_1\ t_2 \to_1 t_1'\ t_2} \qquad\qquad \frac{t_2 \to_1 t_2'}{t_1\ t_2 \to_1 t_1\ t_2'}$$