Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Olaf Müller

WS 1998/99
Lösungsvorschläge zu Blatt 12
23. 2. 1999

Übungen zur Semantik von Programmiersprachen

Aufgabe 1 (Lösungsvorschlag)

(a) Wenn $x = 0$ ist, dann terminiert $ble(x, y)$ in call-by-name-Auswertung unmittelbar, und zwar auch dann, wenn $y = \perp$ ist, da y zur Auswertung von $ble(0, y)$ nicht benötigt wird. Wenn $x \neq 0$ ist, dann terminiert $ble(x, y)$ genau dann, wenn $ble(x - 1, ble(x - y, y))$ terminiert. Ist $x < 0$, so terminiert der äußere Aufruf $ble(x - 1, \ldots)$ nicht. Ist $x > 0$, so terminiert der Aufruf $ble(x - 1, ble(x - y, y))$ in call-by-name-Auswertung unabhängig davon, ob $ble(x - y, y)$ terminiert: wenn $x > 0$ ist, dann wird nach endlichmaliger Subtraktion von 1 von x der Wert 0 erreicht, und der Aufruf $ble(x - 1, ble(x - y, y))$ terminiert, ohne das zweite Argument zu benötigen.

Wir erhalten:

$ble(x, y)$ mit $x, y \in \mathbf{N}_\perp$ terminiert in call-by-name-Auswertung genau dann, wenn gilt:

$$x \geq 0$$

(b) Da wir hier nur eine zweistellige Funktionsdefinition haben, ist die Funktionsumgebung für call-by-name-Auswertung

$$\mathbf{Fenv} = [{\mathbf{N}_\perp}^2 \to \mathbf{N}_\perp] \ .$$

Da **Fenv** ein einstelliges Produkt ist, schreiben wir für $\varphi \in \mathbf{Fenv}$ die Projektion φ_1 auf die erste Komponente abkürzend als φ.

Die Variablenumgebung lautet für call-by-name-Auswertung

$$\mathbf{Env} = [\mathbf{Var} \to \mathbf{N}_\perp] \ .$$

Im folgenden ist $\rho \in \mathbf{Env}$.

Für $F : \mathbf{Fenv} \to \mathbf{Fenv}$ erhalten wir:

$$\begin{aligned}
& F(\varphi) \\
= \quad & \{\text{Definition von } F \text{ für call-by-name}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, [\![\mathbf{if}\ x\ \mathbf{then}\ 1\ \mathbf{else}\ ble(x - 1, ble(x - y, y))]\!]_{na}\ \varphi\ \rho[k/x, m/y] \\
= \quad & \{\text{denotationelle call-by-name-Semantik bedingter Terme}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, Cond([\![x]\!]_{na}\ \varphi\ \rho[k/x, m/y], [\![1]\!]_{na}\ \varphi\ \rho[k/x, m/y], \\
& [\![ble(x - 1, ble(x - y, y))]\!]_{na}\ \varphi\ \rho[k/x, m/y]) \\
= \quad & \{\text{denotationelle call-by-name-Semantik von Variablen, Konstanten und} \\
& \text{Funktionsapplikationen}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, Cond(k, \lfloor 1 \rfloor, \\
& \varphi([\![x - 1]\!]_{na}\ \varphi\ \rho[k/x, m/y], [\![ble(x - y, y)]\!]_{na}\ \varphi\ \rho[k/x, m/y])) \\
= \quad & \{\text{denotationelle call-by-name-Semantik von arithmetischen Termen und}
\end{aligned}$$

$$
\begin{aligned}
& \qquad \text{Funktionsapplikationen}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, Cond(k, \lfloor 1 \rfloor, \\
& \varphi([\![x]\!]_{na}\, \varphi\, \rho[k/x, m/y] -_\perp [\![1]\!]_{na}\, \varphi\, \rho[k/x, m/y], \\
& \varphi([\![x-y]\!]_{na}\, \varphi\, \rho[k/x, m/y], [\![y]\!]_{na}\, \varphi\, \rho[k/x, m/y]))) \\
= & \qquad \{\text{denotationelle call-by-name-Semantik von Variablen, Konstanten und} \\
& \qquad \text{arithmetischen Termen}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, Cond(k, \lfloor 1 \rfloor, \\
& \varphi(k -_\perp \lfloor 1 \rfloor, \varphi([\![x]\!]_{na}\, \varphi\, \rho[k/x, m/y] -_\perp [\![y]\!]_{na}\, \varphi\, \rho[k/x, m/y], m))) \\
= & \qquad \{\text{denotationelle call-by-name-Semantik von Variablen}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, Cond(k, \lfloor 1 \rfloor, \varphi(k -_\perp \lfloor 1 \rfloor, \varphi(k -_\perp m, m))) \\
= & \qquad \{\text{Definition von } Cond\} \\
& \lambda k, m \in \mathbf{N}_\perp \,. \\
& \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor 0 \rfloor \\ \varphi(k -_\perp \lfloor 1 \rfloor, \varphi(k -_\perp m, m)), & \text{falls } k = \lfloor l \rfloor \text{ für ein } l \in \mathbf{N} \text{ mit } l \neq 0 \\ \perp, & \text{falls } k = \perp \end{cases}
\end{aligned}
$$

(c) Die Teilaufgaben (a) und (b) lassen vermuten, daß gilt

$$
F^n(\perp) = \lambda k, m \in \mathbf{N}_\perp \,.\, \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor l \rfloor \text{ mit } l \in \mathbf{N} \wedge 0 \leq l < n \\ \perp & \text{sonst} \end{cases} .
$$

Beweis: Induktion über n

$n = 0$:

$$
\begin{aligned}
& F^0(\perp) \\
= & \qquad \{F^0 = id\} \\
& \perp \\
= & \qquad \{\text{kleinstes Element in } [\mathbf{N}_\perp{}^2 \to \mathbf{N}_\perp] \text{ ist } \lambda k, m \in \mathbf{N}_\perp \,.\, \perp\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, \perp \\
= & \qquad \{\text{Arithmetik}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,.\, \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor l \rfloor \text{ mit } l \in \mathbf{N} \wedge 0 \leq l < 0 \\ \perp & \text{sonst} \end{cases}
\end{aligned}
$$

$n \rightsquigarrow n+1$:

$$
\begin{aligned}
& F^{n+1}(\perp) \\
= & \qquad \{F^{n+1} = F \circ F^n\} \\
& F(F^n(\perp)) \\
= & \qquad \{F \text{ (Teilaufgabe (b))}\} \\
& \lambda k, m \in \mathbf{N}_\perp \,. \\
& \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor 0 \rfloor \\ (F^n(\perp))(k -_\perp \lfloor 1 \rfloor, (F^n(\perp))(k -_\perp m, m)), & \text{falls } k = \lfloor l \rfloor \text{ für ein } l \in \mathbf{N} \text{ mit} \\ & l \neq 0 \\ \perp, & \text{falls } k = \perp \end{cases} \\
= & \qquad \{\text{Induktionsvoraussetzung angewandt auf das äußere } F^n(\perp)\}
\end{aligned}
$$

$$\lambda k, m \in \mathbf{N}_\perp \,.\,
\begin{cases}
\lfloor 1 \rfloor, & \text{falls } k = \lfloor 0 \rfloor \\
\lfloor 1 \rfloor, & \text{falls } k -_\perp \lfloor 1 \rfloor = \lfloor h \rfloor \text{ mit } h \in \mathbf{N} \wedge 0 \leq h < n \wedge \\
& k = \lfloor l \rfloor \text{ für ein } l \in \mathbf{N} \text{ mit } l \neq 0 \\
\perp, & \text{falls } ((k -_\perp \lfloor 1 \rfloor = \lfloor h \rfloor \text{ mit } h \in \mathbf{N} \wedge (h < 0 \vee n \leq h)) \vee k -_\perp \lfloor 1 \rfloor = \perp) \wedge \\
& k = \lfloor l \rfloor \text{ für ein } l \in \mathbf{N} \text{ mit } l \neq 0 \\
\perp, & \text{falls } k = \perp
\end{cases}$$

$$= \quad \{\text{Rechnung in } \mathbf{N}_\perp\}$$

$$\lambda k, m \in \mathbf{N}_\perp \,.\, \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor l \rfloor \text{ mit } l \in \mathbf{N} \wedge 0 \leq l < n+1 \\ \perp & \text{sonst} \end{cases}$$

□

(d) Aus der expliziten Form von $F^n(\perp)$ aus Teilaufgabe (c) folgern wir:

$$\bigsqcup_{n \in \omega} F^n(\perp) = \lambda k, m \in \mathbf{N}_\perp \,.\, \begin{cases} \lfloor 1 \rfloor, & \text{falls } k = \lfloor l \rfloor \text{ mit } l \in \mathbf{N} \wedge 0 \leq l \\ \perp & \text{sonst} \end{cases}$$

$\bigsqcup_{n \in \omega} F^n(\perp)$ ist nach Definition die call-by-name Semantik der gegebenen Funktionsdefinition von *ble*.

Aufgabe 2 (Lösungsvorschlag)

(a) Da p nach Voraussetzung strikt ist und da "if _ then _ else _" im 1. Argument, d.h. in der Bedingung strikt ist, ist auch f strikt.

(b) *Zu zeigen:* $\forall x \in D. f(f(x)) = f(x)$

Beweis: Das im Hinweis vorgeschlagene Prädikat $P(g)$

$$\forall x \in D. f(g(x)) = g(x)$$

ist zulässig in g. Damit können wir $P(f)$ mittels Fixpunktinduktion über f zeigen. Nach Definition ist f der kleinste Fixpunkt der Funktion

$$\tau = \lambda g. \lambda x. \text{if } p(x) \text{ then } x \text{ else } g(g(h(x))) \,.$$

Induktionsanfang: Zu zeigen ist $P(\perp)$, wobei $\perp$ die überall undefinierte Funktion ist.

Es gilt für beliebiges $x \in D$:

$$\begin{aligned}
& f(\perp(x)) \\
= \quad & \{ \ \perp \text{ ist die überall undefinierte Funktion}\} \\
& f(\perp) \\
= \quad & \{ \ f \text{ ist strikt nach Teilaufgabe (a)}\} \\
& \perp \\
= \quad & \{ \ \perp \text{ ist die überall undefinierte Funktion}\} \\
& \perp(x)
\end{aligned}$$

Induktionsschritt: Gelte $P(g)$ (Induktionsvoraussetzung). Zu zeigen ist $P(\tau(g))$.

Es gilt für beliebiges $x \in D$:

$$
\begin{array}{ll}
 & f(\tau(g)(x)) \\
= & \{\ \text{Def. von } \tau\} \\
 & f(\text{if } p(x) \text{ then } x \text{ else } g(g(h(x)))) \\
= & \{\ f \text{ ist strikt und distributiert damit über "if \_ then \_ else \_"}\} \\
 & \text{if } p(x) \text{ then } f(x) \text{ else } f(g(g(h(x)))) \\
= & \{\ \text{Induktionsvoraussetzung}\} \\
 & \text{if } p(x) \text{ then } f(x) \text{ else } g(g(h(x)))
\end{array}
$$

$$
\begin{array}{ll}
= & \{\ \text{Def. von } f\} \\
 & \text{if } p(x) \text{ then } x \text{ else } g(g(h(x))) \\
= & \{\ \text{Def. von } \tau\} \\
 & \tau(g)(x)
\end{array}
$$

Damit ist die Behauptung gezeigt.

Aufgabe 3 (Lösungsvorschlag)

(a) **Zu zeigen:** $P(x)$ ist zulässig, d.h. für alle Ketten $(x_i)_{i\in\omega}$ in D gilt:

$$(\forall i \in \omega : P(x_i)) \Rightarrow P(\bigsqcup_{i\in\omega} x_i)$$

Beweis: Sei $(x_i)_{i\in\omega}$ eine beliebige Kette in D. Im folgenden steht $\sqsupset$ für "echt größer".

$$
\begin{array}{ll}
 & \forall i \in \omega : P(x_i) \\
= & \quad \{\text{Definition von } P\} \\
 & \forall i \in \omega : t(x_i) \neq \perp \\
\Rightarrow & \quad \{\text{Spezialisierung auf } i = 0\} \\
 & t(x_0) \neq \perp \\
\Rightarrow & \quad \{\perp \text{ kleinstes Element}\} \\
 & t(x_0) \sqsupset \perp \\
\Rightarrow & \quad \{t \text{ monoton}\} \\
 & t(\bigsqcup_{i\in\omega} x_i) \sqsupset \perp \\
\Rightarrow & \\
 & t(\bigsqcup_{i\in\omega} x_i) \neq \perp \\
= & \quad \{\text{Definition von } P\} \\
 & P(\bigsqcup_{i\in\omega} x_i)
\end{array}
$$

$\square$

(b) Wir konstruieren ein Gegenbeispiel:

Wähle $t_1(g) = g$ und $t_2(g) = \mathit{fix}(\tau)$ mit:

$$\tau = \lambda f.\lambda x.\ \textbf{if } x \textbf{ then } 0 \textbf{ else } 1 + f(x-1)$$

Damit gilt dann (ohne Beweis):

$$\forall i.\tau^i(\bot) \neq \bigsqcup_{i\in\omega} \tau^i(\bot)$$

Wenn wir nun die ω-Kette $(Y_i)_{i\in\omega}$ mit $Y_i = \tau^i(\bot)$ betrachten, so haben wir ein Beispiel für stetige Terme t_1, t_2 und eine Kette $(Y_i)_{i\in\omega}$ gefunden, für die gilt:

$$t_1(Y_i) = Y_i = \tau^i(\bot) \neq \bigsqcup_{i\in\omega} \tau^i(\bot) = \mathit{fix}(\tau) = t_2(Y_i) \qquad \text{für alle } i$$

aber

$$t_1(\bigsqcup_{i\in\omega} Y_i) = \bigsqcup_{i\in\omega} Y_i = \bigsqcup_{i\in\omega} \tau^i(\bot) = \mathit{fix}(\tau) = t_2(\bigsqcup_{i\in\omega} Y_i)$$

Bemerkung: Jedes Funktional, dessen Iteration eine Kette erzeugt, so daß die kleinste obere Schranke nicht schon Teil der Kette ist, liefert ein Gegenbeispiel nach obigem Schema. □

Aufgabe 4 (Lösungsvorschlag)

(a) Wir ordnen jedem Typ als Semantik die Summe aus den eigentlichen Werten und dem Wert ex zu. Mittels struktureller Induktion über Typen definieren wir deren Semantik VE:

$$\begin{aligned}
VE_{\mathbf{int}} &= \mathbf{N} + \{ex\} \\
VE_{\tau_0 * \tau_1} &= (D_{\tau_0} \times D_{\tau_1}) + \{ex\}\ , \\
&\quad \text{falls } VE_{\tau_0} = D_{\tau_0} + \{ex\} \text{ und } VE_{\tau_1} = D_{\tau_1} + \{ex\} \text{ ist} \\
VE_{\tau_0 -> \tau_1} &= [D_{\tau_0} \to (VE_{\tau_1})_\bot] + \{ex\}\ , \\
&\quad \text{falls } VE_{\tau_0} = D_{\tau_0} + \{ex\} \text{ ist}
\end{aligned}$$

Beim Produkt benötigen wir nicht, daß einzelne Komponenten ex sind. Funktionen brauchen kein ex abzubilden, da man dann sofort dem gesamten Term den Wert ex zuordnen kann (siehe (b)).

(b) Wie in der Vorlesung ordnen wir nur wohlgetypten Termen eine Semantik zu. Die Typisierungsregeln für wohlgetypte Terme werden aus der Vorlesung übernommen. Hinzu kommen Typisierungsregeln für die neuen Terme:

Der Term **raise** ex kann jeden beliebigen Typ τ annehmen. (Setzen wir voraus, daß der gewünschte Typ syntaktisch angegeben wird.)

Für den Term t_0 **handle** t_1 nehmen wir folgende Typisierungsregel hinzu:

$$\frac{t_0 : \tau \qquad t_1 : \tau}{t_0 \ \mathbf{handle}\ t_1 : \tau}$$

Variablenumgebungen ρ sind Funktionen

$$\rho : \mathbf{Var} \to \bigcup\{D_\tau \mid \tau \text{ Typ und } VE_\tau = D_\tau + \{ex\}\}\ ,$$

welche die Typen von Variablen respektieren. Die *Exception* ex wird als Variablenwert nicht benötigt. Wir schreiben **Env** für die cpo aller Variablenumgebungen.

Ist t Term eines Typs τ, dann ist die call-by-value- (d.h. eager-) Semantik $[\![t]\!]$ von t eine Funktion

$$[\![t]\!] : \mathbf{Env} \to (VE_\tau)_\perp \ .$$

Wir definieren die denotationelle call-by-value- (d.h. eager-) Semantik (wohlgetypter Terme) mittels Induktion über die Termstruktur:

$$[\![x]\!] = \lambda\rho \ . \ \lfloor in_1(\rho(x)) \rfloor$$

$$[\![n]\!] = \lambda\rho \ . \ \lfloor in_1(n) \rfloor$$

$$[\![t_0 \ op \ t_1]\!] = \lambda\rho \ . \ ([\![t_0]\!]\rho) \ opE_\perp([\![t_1]\!]\rho)$$

mit $opE : (\mathbf{N} + \{ex\}) \times (\mathbf{N} + \{ex\}) \to (\mathbf{N} + \{ex\})$

$$opE = \lambda k, l \ . \ \begin{cases} in_1(m \ op \ n), & \text{falls } k = in_1(m) \text{ und } l = in_1(n) \text{ für } m, n \in \mathbf{N} \\ in_2(ex), & \text{falls } k = in_2(ex) \text{ oder } l = in_2(ex) \end{cases}$$

$$[\![\mathbf{if} \ t_0 \ \mathbf{then} \ t_1 \ \mathbf{else} \ t_2]\!] = \lambda\rho \ . \ CondE([\![t_0]\!]\rho, [\![t_1]\!]\rho, [\![t_2]\!]\rho)$$

mit $CondE : (\mathbf{N} + \{ex\})_\perp \times (D + \{ex\})_\perp \times (D + \{ex\})_\perp \to (D + \{ex\})_\perp$ (für cpos D)

$$CondE = \lambda k, d_0, d_1 \ . \ \begin{cases} d_0, & \text{falls } k = \lfloor in_1(0) \rfloor \\ d_1, & \text{falls } k = \lfloor in_1(n) \rfloor \text{ mit } n \in \mathbf{N} \text{ und } n \neq 0 \\ \lfloor in_2(ex) \rfloor, & \text{falls } k = \lfloor in_2(ex) \rfloor \\ \perp, & \text{falls } k = \perp \end{cases}$$

$$[\![(t_0, t_1)]\!] = \lambda\rho \ . \ let \ v_0 \Leftarrow [\![t_0]\!]\rho, v_1 \Leftarrow [\![t_1]\!]\rho \ . \ \lfloor ProdE(v_0, v_1) \rfloor$$

mit $ProdE : (D_0 + \{ex\}) \times (D_1 + \{ex\}) \to ((D_0 \times D_1) + \{ex\})$

$$ProdE = \lambda k_0, k_1 \ . \ \begin{cases} in_1(d_0, d_1), & \text{falls } k_0 = in_1(d_0) \text{ und } k_1 = in_1(d_1) \\ in_2(ex), & \text{falls } k_0 = in_2(ex) \text{ oder } k_1 = in_2(ex) \end{cases}$$

$$[\![\mathbf{fst}(t)]\!] = \lambda\rho \ . \ let \ v \Leftarrow [\![t]\!]\rho \ . \ \lfloor \pi E_1(v) \rfloor$$

mit $\pi E_1 : ((D_0 \times D_1) + \{ex\}) \to (D_0 + \{ex\})$

$$\pi E_1 = \lambda p \ . \ \begin{cases} in_1(d_0), & \text{falls } p = in_1(d_0, d_1) \text{ mit } d_0 \in D_0, d_1 \in D_1 \\ in_2(ex), & \text{falls } p = in_2(ex) \end{cases}$$

$$[\![\mathbf{snd}(t)]\!] = \lambda\rho \ . \ let \ v \Leftarrow [\![t]\!]\rho \ . \ \lfloor \pi E_2(v) \rfloor$$

mit $\pi E_2 : ((D_0 \times D_1) + \{ex\}) \to (D_1 + \{ex\})$

$$\pi E_2 = \lambda p \ . \ \begin{cases} in_1(d_1), & \text{falls } p = in_1(d_0, d_1) \text{ mit } d_0 \in D_0, d_1 \in D_1 \\ in_2(ex), & \text{falls } p = in_2(ex) \end{cases}$$

$$[\![\lambda x \ . \ t]\!] = \lambda\rho \ . \ \lfloor in_1(\lambda v \in D_{\tau_0} \ . \ [\![t]\!]\rho[v/x]) \rfloor \ ,$$

falls $\lambda x \ . \ t : \tau_0 - > \tau_1$ und $VE_{\tau_0} = D_{\tau_0} + \{ex\}$

$$[\![(t_0 \ t_1)]\!] = \lambda\rho \ . \ let \ \varphi \Leftarrow [\![t_0]\!]\rho, v \Leftarrow [\![t_1]\!]\rho \ . \ AppE(\varphi, v)$$

mit $AppE : ([D_1 \to (D_0 + \{ex\})_\perp] + \{ex\}) \times (D_1 + \{ex\}) \to (D_0 + \{ex\})_\perp$ (für cpos D_0 und D_1)

$$AppE = \lambda\varphi, v \,.\, \begin{cases} \psi(d_1), & \text{falls } \varphi = in_1(\psi) \text{ und } v = in_1(d_1) \text{ mit } d_1 \in D_1 \\ \lfloor in_2(ex) \rfloor, & \text{falls } \varphi = in_2(ex) \text{ oder } v = in_2(ex) \end{cases}$$

$[\![\textbf{rec}\; y \,.\, (\lambda x \,.\, t)]\!] = \lambda\rho \,.\, \lfloor in_1(\mu\varphi \,.\, (\lambda v \in D_{\tau_0} \,.\, [\![t]\!]\rho[v/x, \varphi/y])) \rfloor$,
falls $x : \tau_0$, $t : \tau_1$ und $VE_{\tau_0} = D_{\tau_0} + \{ex\}$

$[\![\textbf{raise}\; ex]\!] = \lambda\rho \,.\, \lfloor in_2(ex) \rfloor$

$[\![t_0\; \textbf{handle}\; t_1]\!] = \lambda\rho \,.\, let\ v_0 \Leftarrow [\![t_0]\!]\rho \,.\, IfE(v_0, [\![t_1]\!]\rho)$
mit $IfE : (D + \{ex\}) \times (D + \{ex\})_\perp \to (D + \{ex\})_\perp$ (für cpos D)

$$IfE = \lambda k, l \,.\, \begin{cases} \lfloor k \rfloor, & \text{falls } k = in_1(d) \text{ mit } d \in D \\ l, & \text{falls } k = in_2(ex) \end{cases}$$