Karlsruher Institut für Technologie

Lehrstuhl für Programmierparadigmen

Denis Lohner denis.lohner@kit.edu

Sebastian Ullrich sebastian.ullrich@kit.edu

Semantik von Programmiersprachen – SS 2017

http://pp.ipd.kit.edu/lehre/SS2017/semantik

Blatt 8: Typsicherheit Besprechung: 19.06.2017

1. Welche der folgenden Aussagen sind richtig, welche falsch? (H)

(a) $[\mathtt{x} \mapsto \mathsf{int},\ \mathtt{y} \mapsto \mathsf{bool}] \vdash \texttt{(x + 3 <= 17) || (not y)} :: \mathsf{bool}$

(b) Im Kontext $[\mathtt{x} \mapsto \mathsf{bool}]$ ist `true == x` ist ein typkorrekter Ausdruck.

(c) $\mathcal{E}\,[\![\texttt{0 * (x - y)}]\!]\,[\mathtt{x} \mapsto 5,\ \mathtt{y} \mapsto \mathbf{tt}] = 0$

(d) Jeder Ausdruck e hat in jedem Kontext Γ höchstens einen Typ τ.

(e) Der Typ eines Wertes v ließe sich auch so definieren:

$$\text{type}(v) = \tau \quad \text{gdw.} \quad \Gamma \vdash v :: \tau$$

(f) Es gibt keinen Kontext Γ mit

$$\Gamma \vdash \texttt{\{ var x = x; z := x \}; \{ var y = x; if (y) then z := 5 else skip \}} \surd$$

(g) Wenn $\Gamma \vdash c \surd$ und $\langle c,\ \sigma\rangle \to_1 \langle c',\ \sigma'\rangle$ mit $\sigma' :: \Gamma$ und $\Gamma \vdash c' \surd$, dann auch $\sigma :: \Gamma$.

(h) Wenn $\Gamma \vdash c \surd$ und $\sigma :: \Gamma$, dann gibt es c' und σ' mit $\langle c,\ \sigma\rangle \to_1 \langle c',\ \sigma'\rangle$.

2. Ausgaben (H)

Erweitern Sie die Sprache While_T um Ausgaben:

(a) Ergänzen Sie die Syntax um eine neue Anweisung `print` e für Ausgaben.

(b) Passen Sie die Typ-Regeln an. Sowohl Zahlen als auch Wahrheitswerte sind als Ausgabe erlaubt.

(c) Ändern Sie die Big-Step-Semantik, sodass die neue Auswertungsrelation $\langle c,\ \sigma\rangle \Downarrow_o \sigma'$ die Anweisung c im Anfangszustand σ zum Endzustand σ' auswertet, wobei die Liste der Ausgaben o entsteht.

(d) Passen Sie auch die Small-Step-Semantik an Ausgaben an. Unterscheidet sie sich in der Ausdrucksmächtigkeit von der Big-Step-Semantik?

(e) Ist die erweiterte Sprache typsicher? Begründen Sie!

3. Typsicherheit mit der Big-Step-Semantik (Ü)

Typsicherheit lässt sich auch für eine Big-Step-Semantik zeigen, in dieser Aufgabe für While_T:

(a) Ändern Sie die Auswertungsrelation so, dass eine Anweisung mit einem Zustand statt zu einem Endzustand auch zu einem speziellen Fehlerwert Err auswerten kann. Fügen Sie nun für jede Regel, die einen Typcheck (implizit) durchführt, eine neue Regel für den Fehlerfall hinzu. Vergessen Sie auch nicht Propagationsregeln für den Typfehler Err.

(b) Zeigen Sie, dass typkorrekte Programme von konformanten Zuständen aus nie Typfehler erzeugen: Wenn $\Gamma \vdash c \surd$, $\sigma :: \Gamma$ und $\langle c,\ \sigma\rangle \Downarrow \bar{\sigma}'$, dann $\bar{\sigma}' \neq \mathsf{Err}$ und $\bar{\sigma}' :: \Gamma$.

(c) Vergleichen Sie diesen Ansatz, Typsicherheit einer Sprache zu zeigen, mit dem Small-Step-Ansatz aus der Vorlesung.