Karlsruher Institut für Technologie

Lehrstuhl für Programmierparadigmen
Andreas Lochbihler andreas.lochbihler@kit.edu
Joachim Breitner breitner@kit.edu

Semantik von Programmiersprachen – SS 2012

http://pp.info.uni-karlsruhe.de/lehre/SS2012/semantik

Lösungen zu Blatt 3: Small-Step-Semantik Besprechung: 8.05.2012

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

(a) b_1 `&&` b_2 verhält sich semantisch wie `if (`b_1`) then true else` b_2.

(b) $\langle$`if (`b`) then` c_1 `else` c_2, $\sigma\rangle$ und $\langle$`if (not` b`) then` c_2 `else` c_1, $\sigma\rangle$ haben die gleichen Ableitungsfolgen.

(c) Wenn $\langle c_1\texttt{;}\ c_2,\ \sigma\rangle \xrightarrow{*}_1 \langle c_1'\texttt{;}\ c_2,\ \sigma'\rangle$, dann auch $\langle c_1,\ \sigma\rangle \xrightarrow{*}_1 \langle c_1',\ \sigma'\rangle$.

(d) Wenn $\langle c,\ \sigma\rangle \xrightarrow{n}_1 \langle c,\ \sigma\rangle$, dann $n = 0$.

(e) $\langle$`x := 1; while (x <= 2) do x := x * 2`, $\sigma\rangle \xrightarrow{12}_1 \langle$`skip`, $\sigma[\texttt{x} \mapsto 4]\rangle$.

(f) Wenn $\langle c,\ \sigma\rangle \xrightarrow{*}_1 \langle$`skip`, $\sigma'\rangle$, dann enthält c syntaktisch keine Schleife der Form `while (true) do` c'.

(g) Wenn $(\gamma_i)_i$ und $(\delta_j)_j$ Ableitungsfolgen für $\langle c,\ \sigma\rangle$ sind, dann $(\gamma_i)_i = (\delta_j)_j$.

(h) Wenn $\langle c,\ \sigma\rangle \xrightarrow{*}_1 \langle c',\ \sigma'\rangle$, dann enthält c' höchstens dreimal so viele AST-Knoten wie c.

Lösung:

(1a) Falsch. Das würde funktionieren, wenn man ein „funktionales“ `if` hätte, allerdings ist `if (`b_1`) then true else` b_2 gar kein While-Programm.

(1b) Falsch. Nach dem ersten Reduktionsschritt sind die Ableitungsfolgen alle gleich. Das erste Element enthält jedoch auch die Anweisung, die bei beiden unterschiedlich ist.

(1c) Falsch. Gegenbeispiel: $c_1 =$ `skip`, $c_2 =$ `while (true) do (skip; skip)`, $c_1' =$ `skip; skip` und $\sigma = \sigma'$.

$$\langle c_1\texttt{;}\ c_2,\ \sigma\rangle \to_1 \langle c_2,\ \sigma\rangle \to_1 \langle\texttt{if (true) then } c_1'\texttt{;}\ c_2\texttt{ else skip},\ \sigma\rangle \to_1 \langle c_1'\texttt{;}\ c_2,\ \sigma\rangle$$

aber $\langle c_1,\ \sigma\rangle \not\to_1$

(1d) Falsch. Gegenbeispiel: $c =$ `while (true) do skip`, σ beliebig.

$$\begin{aligned}&\langle c,\ \sigma\rangle \to_1 \langle\texttt{if (true) then skip; while (true) do skip else skip},\ \sigma\rangle\\ \to_1\ &\langle\texttt{skip; while (true) do skip},\ \sigma\rangle \to_1 \langle c,\ \sigma\rangle\end{aligned}$$

Somit: $\langle c,\ \sigma\rangle \xrightarrow{3}_1 \langle c,\ \sigma\rangle$.

(1e) Richtig. Sei $c =$ `x := x * 2`, $w =$ `while (x <= 2) do` c, $w' =$ `if (x <= 2) then` c`;` w `else skip` und $\sigma_n = \sigma[\texttt{x} \mapsto n]$.

$$\begin{aligned}&\langle\texttt{x := 0;}\ w,\ \sigma\rangle \to_1 \langle\texttt{skip;}\ w,\ \sigma_1\rangle\\ \to_1\ &\langle w,\ \sigma_1\rangle \to_1 \langle w',\ \sigma_1\rangle \to_1 \langle c\texttt{;}\ w,\ \sigma_1\rangle \to_1 \langle\texttt{skip;}\ w,\ \sigma_2\rangle\\ \to_1\ &\langle w,\ \sigma_2\rangle \to_1 \langle w',\ \sigma_2\rangle \to_1 \langle c\texttt{;}\ w,\ \sigma_2\rangle \to_1 \langle\texttt{skip;}\ w,\ \sigma_4\rangle\\ \to_1\ &\langle w,\ \sigma_4\rangle \to_1 \langle w',\ \sigma_4\rangle \to_1 \langle\texttt{skip},\ \sigma_4\rangle\end{aligned}$$

(1f) Falsch. Gegenbeispiel:

$$\langle \texttt{if (false) then while (true) do skip else skip}, \sigma\rangle \to_1 \langle \texttt{skip}, \sigma\rangle$$

(1g) Falsch, dies gilt nur für maximale Ableitungsfolgen. Bei normalen Ableitungsfolgen kann die eine ein Präfix der anderen sein.

(1h) Falsch. Für jedes Programm gibt es zwar eine maximal erreichbare Größe, das Verhältnis ist aber nicht beschränkt. Gegenbeispiel für 3:

```
while (true) do while (true) do while (true) do (skip)
```

2. Small-Step simuliert Big-Step (H)

Beweisen Sie durch Induktion über die Regeln der Big-Step-Semantik, dass jede Ausführung in der Big-Step-Semantik eine äquivalente Ausführung in der Small-Step-Semantik besitzt, d.h.: Aus $\langle c,\ \sigma\rangle \Downarrow \sigma'$ folgt $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$.

Lösung:

Beweis. Beweis per Induktion über $\langle c,\ \sigma\rangle \Downarrow \sigma'$:

- Fall $\textsc{Skip}_{\text{BS}}$: Wir haben $c = \texttt{skip}$ und $\sigma' = \sigma$.
  Zu zeigen: $\langle \texttt{skip},\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma\rangle$. Trivial ($* = 0$).
- Fall $\textsc{Ass}_{\text{BS}}$: Wir haben $c = x\ \texttt{:=}\ a$ und $\sigma' = \sigma[\mathtt{x} \mapsto \mathcal{A}[\![a]\!]\sigma]$.
  Zu zeigen: $\langle \mathtt{x}\ \texttt{:=}\ a,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma[\mathtt{x} \mapsto \mathcal{A}[\![a]\!]\sigma]\rangle$. Dies folgt aus Regel $\textsc{Ass}_{\text{SS}}$ mit $* = 1$.
- Fall $\textsc{Seq}_{\text{BS}}$: Induktionsannahmen: (i) $\langle c_1,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ und (ii) $\langle c_2,\ \sigma'\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma''\rangle$.
  Zu zeigen: $\langle c_1\texttt{;}\ c_2,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma''\rangle$.
  Aus (i) folgt nach dem Liftinglemma 6 für Sequenz, dass auch $\langle c_1\texttt{;}\ c_2,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip;}\ c_2,\ \sigma'\rangle$ gilt. Weiterhin gilt $\langle \texttt{skip;}\ c_2,\ \sigma'\rangle \to_1 \langle c_2,\ \sigma'\rangle$ nach Regel $\textsc{Seq2}_{\text{SS}}$. Zusammen mit (ii) folgt die Behauptung aus der Transitivität von $\overset{*}{\to}_1$.
- Fall $\textsc{IfTT}_{\text{BS}}$: Induktionsannahmen: (i) $\mathcal{B}[\![b]\!]\sigma = \mathbf{tt}$ und (ii) $\langle c_1,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$.
  Zu zeigen: $\langle \texttt{if (}b\texttt{) then}\ c_1\ \texttt{else}\ c_2,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$.
  Wegen (i) gilt $\langle \texttt{if (}b\texttt{) then}\ c_1\ \texttt{else}\ c_2,\ \sigma\rangle \to_1 \langle c_1,\ \sigma\rangle$ nach Regel $\textsc{IfTT}_{\text{SS}}$. Mit (ii) folgt die Behauptung.
- Fall $\textsc{IfFF}_{\text{BS}}$: Analog.
- Fall $\textsc{WhileFF}_{\text{BS}}$: Induktionsannahme: $\mathcal{B}[\![b]\!]\sigma = \mathbf{ff}$.
  Zu zeigen: $\langle \texttt{while (}b\texttt{) do}\ c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma\rangle$.
  Es gilt: $\langle \texttt{while (}b\texttt{) do}\ c,\ \sigma\rangle \to_1 \langle \texttt{if (}b\texttt{) then}\ c\texttt{; while (}b\texttt{) do}\ c\ \texttt{else skip},\ \sigma\rangle$ nach Regel $\textsc{While}_{\text{SS}}$ und $\cdots \to_1 \langle \texttt{skip},\ \sigma\rangle$ nach Regel $\textsc{IfFF}_{\text{SS}}$ mit der Induktionsannahme. Damit folgt die Behauptung.
- Fall $\textsc{WhileTT}_{\text{BS}}$: Induktionsannahmen: (i) $\mathcal{B}[\![b]\!]\sigma = \mathbf{tt}$, (ii) $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ und (iii) $\langle \texttt{while (}b\texttt{) do}\ c,\ \sigma'\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma''\rangle$. Zu zeigen: $\langle \texttt{while (}b\texttt{) do}\ c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma''\rangle$.
  Beweis mit dem Liftinglemma für Sequenz (Lem. 6):

$$\begin{aligned}&\langle \texttt{while (}b\texttt{) do}\ c,\ \sigma\rangle \to_1 \langle \texttt{if (}b\texttt{) then}\ c\texttt{; while (}b\texttt{) do}\ c\ \texttt{else skip},\ \sigma\rangle\\ \to_1\ &\langle c\texttt{; while (}b\texttt{) do}\ c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip; while (}b\texttt{) do}\ c,\ \sigma'\rangle \to_1 \langle \texttt{while (}b\texttt{) do}\ c,\ \sigma'\rangle\\ \overset{*}{\to}_1\ &\langle \texttt{skip},\ \sigma''\rangle\end{aligned}$$

□

3. Semantische Äquivalenz bei Small-Step (Ü)

Zwei Programme c und c' sind *äquivalent* bezüglich der Small-Step-Semantik, falls für alle Zustände σ, σ' gilt:

(i) $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ genau dann, wenn auch $\langle c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ und

(ii) $\langle c,\ \sigma\rangle \overset{\infty}{\to}_1$ genau dann, wenn auch $\langle c',\ \sigma\rangle \overset{\infty}{\to}_1$.

Vergleichen Sie diesen Äquivalenzbegriff mit dem semantischen Äquivalenzbegriff für die Big-Step-Semantik. Sie können dazu davon ausgehen, dass für alle c, σ und σ' gilt:

$$\langle c,\ \sigma\rangle \Downarrow \sigma' \qquad \text{gdw.} \qquad \langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle.$$

Beweisen Sie, dass die folgenden Programme im Small-Step-Sinn äquivalent sind:

(a) `while (`b`) do` c und `if (`b`) then` c`; while (`b`) do` c `else skip`

(b) c`; while (true) do` c' und `while (true) do` c''

(c) Wenn c und c' äquivalent sind, dann auch `while (`b`) do` c und `while (`b`) do` c'.

Überlegen Sie sich einen sinnvollen Äquivalenzbegriff, für den (c) nicht gilt.

Lösung: Für unsere spezielle Small-Step-Semantik folgt (ii) bereits aus (i): Angenommen, dass $\langle c,\ \sigma\rangle \overset{\infty}{\to}_1$. Wegen der Eindeutigkeit maximaler Ableitungen (Kor. 5) gibt es dann kein σ' mit $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$. Damit gilt wegen (i) auch $\langle c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ für kein σ'.

Wegen der Existenz und Eindeutigkeit maximaler Ableitung (Kor. 5) gibt es aber entweder σ' und ein blockierendes c'' mit $\langle c',\ \sigma\rangle \overset{*}{\to}_1 \langle c'',\ \sigma'\rangle$ oder $\langle c',\ \sigma\rangle \overset{\infty}{\to}_1$. Im ersten Fall ist aber $c'' = \texttt{skip}$, weil `skip` das einzige blockierende Programm ist (Fortschrittslemma 3), also gilt der zweite Fall, $\langle c',\ \sigma\rangle \overset{\infty}{\to}_1$.

Die Äquivalenz von c und c' bezüglich der Big-Step-Semantik ist definiert als: $\langle c,\ \sigma\rangle \Downarrow \sigma'$ genau dann, wenn auch $\langle c',\ \sigma\rangle \Downarrow \sigma'$ für alle σ, σ'.

Da aber $\langle c,\ \sigma\rangle \Downarrow \sigma'$ und $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ äquivalent sind (Kor. 10), sind somit diese beiden Äquivalenzbegriffe die gleichen.

(a) Da Big-Step- und Small-Step-Äquivalenz gleich sind, folgt dies direkt aus dem Schleifenabwicklungslemma (Lem. 1). Der direkte Beweis ist aber auch einfach: Falls $\mathcal{B}[\![b]\!]\,\sigma = \mathbf{tt}$, dann $\langle$`while (`b`) do` c`,` $\sigma\rangle \to_1 \langle$`if (`$b$`) then` c`; while (`b`) do` c `else skip`, $\sigma\rangle$ und damit die Behauptung. Falls $\mathcal{B}[\![b]\!]\,\sigma = \mathbf{ff}$, werten beide in einem Schritt zu `skip` aus.

(b) Beide Programme terminieren nie, damit sind sie äquivalent. Nichtterminationsbeweise:

i. Ein Programm der Form `while (true) do` c terminiert nie, also für ein beliebiges n und alle σ, σ' gilt $\langle$`while (true) do` c`,` $\sigma\rangle \overset{n}{\to}_1 \langle$`skip`, $\sigma'\rangle$ nicht.
Vollständige Induktion über n: Angenommen, für alle $m < n$ und σ, σ' gilt nicht, dass $\langle$`while (true) do` c`,` $\sigma\rangle \overset{m}{\to}_1 \langle$`skip`, $\sigma'\rangle$.
Zu zeigen: $\langle$`while (true) do` c`,` $\sigma\rangle \overset{n}{\to}_1 \langle$`skip`, $\sigma'\rangle$ gilt nicht.
Einzige Ableitungen für `while (true) do` c:

$$\begin{aligned} &\langle \texttt{while (true) do } c,\ \sigma\rangle \\ \to_1\ &\langle \texttt{if (true) then } c\texttt{; while (true) do } c \texttt{ else skip},\ \sigma\rangle \\ \to_1\ &\langle c\texttt{; while (true) do } c,\ \sigma\rangle \end{aligned}$$

Somit $n \geq 2$. Angenommen, $\langle c$`; while (true) do` c`,` $\sigma\rangle \overset{n-2}{\to}_1 \langle$`skip`, $\sigma'\rangle$. Nach dem Zerlegungslemma für Sequenz (Lem. 7) gibt es i, j und σ^* mit $\langle c,\ \sigma\rangle \overset{i}{\to}_1 \langle$`skip`, $\sigma^*\rangle$, $\langle$`while (true) do` c`,` $\sigma^*\rangle \overset{j}{\to}_1 \langle$`skip`, $\sigma'\rangle$ und $i + j + 1 = n - 2$. Damit gilt $j < n$, im Widerspruch zur Induktionsannahme.

ii. Die Hintereinanderausführung von Programmen, von denen eines nicht terminiert, terminiert auch nicht. Dies folgt aus dem Zerlegungslemma für Sequenz (Lem. 7) mit Widerspruchsbeweis.

(c) Es genügt aus Symmetriegründen, (i) nur in eine Richtung zu zeigen.
Wir zeigen durch vollständige Induktion nach n dass für beliebiges σ aus der Ableitung $\langle \texttt{while (}b\texttt{) do } c,\ \sigma\rangle \overset{n}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ die Ableitung $\langle \texttt{while (}b\texttt{) do } c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ folgt.
Damit erhalten wir als Induktionsannahme dass für alle $m < n$ und σ aus der Ableitung $\langle \texttt{while (}b\texttt{) do } c,\ \sigma\rangle \overset{m}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ folgt, dass $\langle \texttt{while (}b\texttt{) do } c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ gilt.
Falls $\mathcal{B}\,[\![b]\!]\,\sigma = \mathbf{ff}$, folgt die Behauptung trivial mit den Regeln $\textsc{While}_{\textsc{ss}}$ und $\textsc{IfFF}_{\textsc{ss}}$ in zwei Schritten. Sei also $\mathcal{B}\,[\![b]\!]\,\sigma = \mathbf{tt}$. Durch Regelinversion erhält man:

$$\begin{aligned}&\langle \texttt{while (}b\texttt{) do } c,\ \sigma\rangle \to_1 \langle \texttt{if (}b\texttt{) then } c\texttt{; while (}b\texttt{) do } c \texttt{ else skip},\ \sigma\rangle\\ \to_1\ &\langle c\texttt{; while (}b\texttt{) do } c,\ \sigma\rangle \overset{n-2}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle\end{aligned}$$

Nach dem Zerlegungslemma für Sequenz (Lem. 7) gibt es damit i, j und σ^* mit $\langle c,\ \sigma\rangle \overset{i}{\to}_1 \langle \texttt{skip},\ \sigma^*\rangle$, $\langle \texttt{while (}b\texttt{) do } c,\ \sigma^*\rangle \overset{j}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$ und $i + j + 1 = n - 2$. Da $j < n$ folgt nach der Induktionsannahme, dass $\langle \texttt{while (}b\texttt{) do } c',\ \sigma^*\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma'\rangle$. Wegen $\langle c,\ \sigma\rangle \overset{i}{\to}_1 \langle \texttt{skip},\ \sigma^*\rangle$ gilt auch $\langle c,\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma^*\rangle$ und deswegen mit der semantischen Äquivalenz von c und c' auch $\langle c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip},\ \sigma^*\rangle$. Somit ergibt sich folgende maximale Ableitungfolge (unter Verwendung des Liftinglemma 6):

$$\begin{aligned}&\langle \texttt{while (}b\texttt{) do } c',\ \sigma\rangle \to_1 \langle \texttt{if (}b\texttt{) then } c'\texttt{; while (}b\texttt{) do } c' \texttt{ else skip},\ \sigma\rangle\\ \to_1&\langle c'\texttt{; while (}b\texttt{) do } c',\ \sigma\rangle \overset{*}{\to}_1 \langle \texttt{skip; while (}b\texttt{) do } c',\ \sigma^*\rangle \to_1 \langle \texttt{while (}b\texttt{) do } c',\ \sigma^*\rangle\\ \overset{*}{\to}_1&\langle \texttt{skip},\ \sigma'\rangle\end{aligned}$$

und damit die Behauptung.

In der Vorlesung wurde die Äquivalenz von Programmen bezüglich einer Menge V von Variablen eingeführt (Definition 8). Wenn $V \subseteq \mathsf{Var}$ jene Variablen enthält, die nicht mit `tmp` beginnen, so sind die beiden Programme $c =$ `a := a - 1; tmp1 := 0` und $c' =$ `a := a - 1; tmp1 := a` bezüglich V äquivalent, nicht aber `while (0 <= tmp1) do` c und `while (0 <= tmp1) do` c'.