

## Übungen zu Theoretische Informatik III

Abgabe bis zum 28.06.2005

Auf den Folien 111–115 werden Semantik-Funktionen für arithmetische und boolesche Ausdrücke definiert, nämlich

$$\mathcal{A} : \mathbf{Aexp} \rightarrow (\Sigma \rightarrow \mathbb{N}) \quad \text{und} \quad \mathcal{B} : \mathbf{Bexp} \rightarrow (\Sigma \rightarrow \mathbb{N}),$$

wobei  $\Sigma = \mathbb{N}^{\mathbf{Loc}} = \{\sigma : \mathbf{Loc} \rightarrow \mathbb{N}\}$  die Menge aller Speicherzustände modelliert: jeder Variable (Adresse) wird eine natürliche Zahl (Speicherinhalt) zugewiesen.

Die erste Aufgabe zeigt wie man zwei Programme auf Äquivalenz prüft; die zweite führt an einem Beispiel die Äquivalenz zweier Semantiken operationalen Charakters vor. Beide Semantiken verwenden diesselbe Definition für die Veränderungen, die das kleinste/atomare Programm bewirken kann (nämlich keine).

$$\overline{(\mathbf{skip}, \sigma) \rightarrow \sigma} \quad (\text{skip})$$

Sie unterscheiden sich jedoch darin, wie sie zusammengesetzte Programme behandeln.

$\mathcal{C}_{\text{BIG}}$ : Die Semantik  $\mathcal{C}_{\text{BIG}}$  bestimmt die Bedeutung zusammengesetzter Programme rekursiv, d.h. unter Zuhilfenahme der bereits bestimmten Bedeutungen der Teilprogramme; da in gewisser Weise ganze Teilprogramme auf einmal abgearbeitet werden heißt sie *Big Step*-Semantik.

$$\frac{\mathcal{A}[a]\sigma = m}{(X := a, \sigma) \rightarrow \sigma[m/X]} \quad (:=)$$

$$\frac{(c_1, \sigma) \rightarrow \sigma' \quad (c_2, \sigma') \rightarrow \sigma''}{(c_1; c_2, \sigma) \rightarrow \sigma''} \quad (;)$$

$$\frac{(c_1, \sigma) \rightarrow \sigma' \quad \mathcal{B}[b]\sigma = \mathbf{true}}{(\mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2 \mathbf{ fi}, \sigma) \rightarrow \sigma'} \quad (\mathbf{if} \dots \mathbf{fi}_{\text{true}})$$

$$\frac{(c_2, \sigma) \rightarrow \sigma' \quad \mathcal{B}[b]\sigma = \mathbf{false}}{(\mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2 \mathbf{ fi}, \sigma) \rightarrow \sigma'} \quad (\mathbf{if} \dots \mathbf{fi}_{\text{false}})$$

$$\frac{\mathcal{B}[b]\sigma = \mathbf{true} \quad (c_1, \sigma) \rightarrow \sigma' \quad (\mathbf{while } b \mathbf{ do } c \mathbf{ od}, \sigma') \rightarrow \sigma''}{(\mathbf{while } b \mathbf{ do } c \mathbf{ od}, \sigma) \rightarrow \sigma''} \quad (\mathbf{while}_{\text{true}})$$

$$\frac{\mathcal{B}[b]\sigma = \mathbf{false}}{(\mathbf{while } b \mathbf{ do } c \mathbf{ od}, \sigma) \rightarrow \sigma} \quad (\mathbf{while}_{\text{false}})$$

Die Semantik  $\mathcal{C}_{\text{BIG}}[a]$  des Programms  $a \equiv Y := Y + 1; X := Y + X$  mit  $X \neq Y$  wird dann für die 0-Speicherbelegung  $\mathbf{0}$  ( $\mathbf{0}$  sei die konstante 0-Funktion, also  $\forall n \in \mathbf{Loc}. \mathbf{0}(n) = 0$ ) folgendermaßen bestimmt (hier sind auch die Schritte von  $\mathcal{A}$  mit aufgeführt und zusätzlich der “Syntaxbaum”):

$$\frac{\overline{\mathbf{0}} \quad \overline{\mathcal{A}[Y]\mathbf{0}} \quad \overline{\mathbf{1}} \quad \overline{\mathcal{A}[\mathbf{1}]\mathbf{0}}}{\overline{\mathbf{0}[1/Y]}} \quad + \quad \frac{\overline{\mathbf{0}} \quad \overline{\mathcal{A}[X]\mathbf{0}[1/Y]} \quad \overline{\mathbf{1}} \quad \overline{\mathcal{A}[Y]\mathbf{0}[1/Y]}}{\overline{\mathbf{0}[1/Y][1/X]}} \quad + \quad \frac{\overline{Y} \quad \overline{v} \quad \overline{\mathbf{1}} \quad \overline{n}}{\overline{Y := Y + 1}} \quad + \quad \frac{\overline{Y} \quad \overline{v} \quad \overline{X} \quad \overline{v}}{\overline{X := Y + X}} \quad +$$

$$\frac{\overline{\mathbf{0}[1/Y][1/X]}}{\overline{\mathbf{0}[1/Y][1/X]}} \quad ; \quad \frac{\overline{Y := Y + 1} \quad \overline{X := Y + X}}{\overline{Y := Y + 1; X := Y + X}} \quad ;$$

also  $\mathcal{C}_{\text{BIG}}[a]\mathbf{0} = \{n \mapsto 1 \mid n \in \{0, 1\}\} \cup \{n \mapsto 0 \mid 0 \in \mathbb{N} \setminus \{0, 1\}\}$  ( $v$  und  $n$  stehen für *Variable* und *natürliche Zahl*). Für allgemeines  $\sigma : \mathbf{Loc} \rightarrow \mathbb{N}$  sieht der Beweis so aus:

$$\frac{\overline{\sigma(Y)} \quad \overline{\mathcal{A}[Y]\sigma} \quad \overline{\mathbf{1}} \quad \overline{\mathcal{A}[\mathbf{1}]\sigma}}{\overline{\sigma(Y) + 1}} \quad + \quad \frac{\overline{\sigma(X)} \quad \overline{\mathcal{A}[X]\sigma[\sigma(Y)+1/Y]} \quad \overline{\sigma(Y) + 1} \quad \overline{\mathcal{A}[Y]\sigma[\sigma(Y)+1/Y]}}{\overline{\sigma(X) + \sigma(Y) + 1}} \quad +$$

$$\frac{\overline{\sigma[\sigma(Y) + 1/Y]}}{\overline{\sigma[\sigma(Y) + 1/Y]}} \quad := \quad \frac{\overline{\sigma[\sigma(Y) + 1/Y][\sigma(X) + \sigma(Y) + 1/X]}}{\overline{\sigma[\sigma(Y) + 1/Y][\sigma(X) + \sigma(Y) + 1/X]}} \quad ;$$

$\mathcal{C}_{\text{SOS}}$ : Die Semantik  $\mathcal{C}_{\text{SOS}}$  definiert die *einzelnen* Arbeitsschritte, die eine abstrakten Maschine auf dem Speicher ausführt; deshalb heißt  $\mathcal{C}_{\text{SOS}}$  auch *Einzel-schrittsemantik*.

$$\overline{(X := a, \sigma) \rightarrow \sigma[\mathcal{A}[a]\sigma/X]} \quad (:=)$$

$$\overline{(c_1; c_2, \sigma) \rightarrow (c_2, \mathcal{C}_{\text{SOS}}[c_1]\sigma')} \quad (;)$$

$$\overline{(\mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2 \mathbf{ fi}, \sigma) \rightarrow \begin{cases} (c_1, \sigma) & \text{if } \mathcal{B}[b]\sigma = \mathbf{true} \\ (c_2, \sigma) & \text{if } \mathcal{B}[b]\sigma = \mathbf{false} \end{cases}} \quad (\mathbf{if} \dots \mathbf{fi})$$

$$\overline{(\mathbf{while } b \mathbf{ do } c \mathbf{ od}, \sigma) \rightarrow \begin{cases} (\mathbf{while } b \mathbf{ do } c \mathbf{ od}, \mathcal{C}_{\text{SOS}}[c]\sigma') & \text{if } \mathcal{B}[b]\sigma = \mathbf{true} \\ (\mathbf{skip}, \sigma) & \text{if } \mathcal{B}[b]\sigma = \mathbf{false} \end{cases}} \quad (\mathbf{while})$$

Die Berechnung der Einzelschritt-Semantik von  $a \equiv Y := Y + 1; X := Y + X$  mit  $X \neq Y$  sieht für  $\sigma : \mathbf{Loc} \rightarrow \mathbb{N}$  wie folgt aus.

$$\begin{aligned}
\mathcal{C}_{\text{sos}}[[Y := Y + 1; X := Y + X]]\sigma & \\
= \mathcal{C}_{\text{sos}}[[X := Y + X]](\mathcal{C}_{\text{sos}}[[Y := Y + 1]]\sigma) & \quad (;) \\
= \mathcal{C}_{\text{sos}}[[X := Y + X]](\sigma[\mathcal{A}[[Y + 1]]\sigma/Y]) & \quad (:=) \\
= \mathcal{C}_{\text{sos}}[[X := Y + X]](\sigma[\mathcal{A}[[Y]]\sigma + \mathcal{A}[[1]]\sigma/Y]) & \quad (+) \\
= \mathcal{C}_{\text{sos}}[[X := Y + X]](\sigma[\sigma(Y) + 1/Y]) & \quad (\mathcal{A}) \\
= (\sigma[\sigma(Y) + 1/Y])[\mathcal{A}[[Y + X]]\sigma[\sigma(Y) + 1/Y]/X] & \quad (:=) \\
= (\sigma[\sigma(Y) + 1/Y])[\mathcal{A}[[Y]]\sigma[\sigma(Y) + 1/Y] + \mathcal{A}[[X]]\sigma[\sigma(Y) + 1/Y]/X] & \quad (+) \\
= (\sigma[\sigma(Y) + 1/Y])[(\sigma(Y) + 1) + \sigma(X)/X] & \quad (\mathcal{A})
\end{aligned}$$

### Aufgabe 6.1

4 Punkte

Es sei  $b \in \mathbf{Bexp}$  und  $c \in \mathbf{Cmd}$ ,  $w \equiv \mathbf{while } b \mathbf{ do } c$ ; und  $w' \equiv \mathbf{if } b \mathbf{ then } c; w \mathbf{ else skip}$ ;

Zeigen Sie

$$\mathcal{C}_{\text{BIG}}[[w]] = \mathcal{C}_{\text{BIG}}[[w']]$$

unter Verwendung obiger Regeln.

### Aufgabe 6.2

4 Punkte

Es sei

$c \equiv \mathbf{while } \mathbf{true} \mathbf{ do } \mathbf{skip}$ ;

Zeigen Sie  $\mathcal{C}_{\text{BIG}}[[c]] = \perp$  und  $\mathcal{C}_{\text{sos}}[[c]] = \perp$  direkt unter Benutzung obiger Regeln. Die Lösung für die erste Gleichung besteht in einem unendlichen Beweisbaum bzw. dessen Beschreibung, während die der zweiten eine unendliche Sequenz von Umformungsschritten ist.

### Aufgabe 6.3

4 Punkte

Ideale (Server-)Betriebssysteme *sollten* nie abstürzen. Falls also  $b, b' \in \mathbf{Cmd}$  ideale Betriebssysteme sind, so muss  $\mathcal{C}_{\text{BIG}}[[b]]\sigma = \perp = \mathcal{C}_{\text{BIG}}[[b']]\sigma$  gelten. Sind deshalb alle Betriebssysteme gleich? Erklären Sie ganz kurz, in welchem Sinne die Semantik  $\mathcal{C}_{\text{BIG}}$  total ist, auch wenn  $\mathcal{C}_{\text{BIG}}[[b]]\sigma = \perp = \mathcal{C}_{\text{BIG}}[[b']]\sigma$  gilt (Aufgabe 2 !!!), und was man heranziehen könnte um  $b$  von  $b'$  zu unterscheiden (1 Punkt).

Nun wollen wir den Benutzer mit in die Semantik einbeziehen. Deshalb nehmen wir neben dem Speicher noch einen Strom von Benutzereingaben  $\pi$  als zusätzlichen Parameter für unsere Semantik an, wobei  $\pi : \mathbb{N} \rightarrow \mathbb{N}$ . Außerdem sei  $\mathbf{next} \in \mathbf{Aexp}$  ein Ausdruck, der eine Benutzereingabe liest, nämlich die nächste. Erweitern Sie  $\mathcal{C}_{\text{BIG}}$  entsprechend zu einer Funktion

$$\mathcal{C}_{\text{BIG}} : \mathbf{Cmd} \rightarrow (\Sigma \rightarrow (\mathbb{N}^{\mathbb{N}} \rightarrow \Sigma))$$

wobei auch eine angemessene neue Regel für  $\mathbf{next}$  anzugeben ist.