

# 4.8 Operationale Semantik

## 4.8.1 Transitionssysteme

- Ein Transitionssystem ist gegeben durch einen gerichteten (nicht notwendig endlichen) Graph. Die Knoten eines Transitionssystems nennen wir Zustände, die Kanten Transitionen.
- Gibt es in einem Transitionssystem mit Knotenmenge  $Q$  und Kanten gegeben durch die zweistellige Relation  $\rightarrow$  einen ausgezeichneten Startzustand  $q_0$ , so sprechen wir von dem initialen Transitionssystem  $(Q, \rightarrow, q_0)$ .
- Ist im initialen Transitionssystem  $(Q, \rightarrow, q_0)$  jeder Zustand  $q$  auf einem gerichteten Weg von  $q_0$  aus erreichbar, so heißt das System initial zusammenhängend.
- Endzustände des Systems sind Knoten mit Ausgangsgrad 0.
- Ein System heißt deterministisch, wenn für alle  $u, v_1, v_2 \in Q$  aus  $u \rightarrow v_1$  und  $u \rightarrow v_2$  folgt  $v_1 = v_2$ .
- Es sei ein deterministisches Transitionssystem mit Knotenmenge  $Q$  und Kantenmenge  $\rightarrow$  gegeben. Es seien  $X, Y \subseteq Q$  gegeben, so dass der Ausgangsgrad von  $y$  0 ist für alle  $y \in Y$ . Dann erhalten wir eine partielle Abbildung  $f: X \rightarrow^p Y$  durch die Festlegung  $f(x) = y$ , falls es einen gerichteten Weg von  $x$  nach  $y$  gibt,  $f(x)$  undefiniert, sonst.  
(Man überprüfe, dass diese Abbildung wohldefiniert ist, d.h. dass es maximal einen Knoten  $y \in Y$  geben kann, der von  $x$  aus erreichbar ist.)

## 4.8.2 Einzelschrittsemantik

Bei der Definition der Mengen **Aexp** und **Bexp** hatten wir schon eine Auswertungsstrategie angegeben, d.h. eine Abbildung

$$\mathcal{A}: \mathbf{Aexp} \rightarrow (\Sigma \rightarrow \mathbb{N}).$$

Zur Erinnerung: Wir schreiben  $\mathcal{A}[[a]]\sigma = n$ , falls der Ausdruck  $a$  unter der Belegung  $\sigma$  zu  $n$  ausgewertet wird.

Entsprechend definieren wir

$$\mathcal{B}: \mathbf{Bexp} \rightarrow (\Sigma \rightarrow \mathbb{B}).$$

Bei der Einzelschritt-Semantik gehen wir davon aus, dass für alle arithmetischen oder booleschen Ausdrücke diese Abbildung in einem Schritt ausgeführt werden kann.

Dagegen müssen wir die Ausführung der Anweisungen (d.h. der Elemente von **Cmd**) durch Regeln definieren.

Wir benutzen ein Transitionssystem, dessen Knoten (Zustände) durch die Menge

$$(\mathbf{Cmd} \times \Sigma) \cup \Sigma$$

gegeben wird. Die Kanten werden wir so bilden, dass ein deterministisches Transitionssystem entsteht, in dem alle Knoten aus  $(\mathbf{Cmd} \times \Sigma)$  Ausgangsgrad 1 haben, alle Knoten aus  $\Sigma$  dagegen Ausgangsgrad 0.

Es wird also durch die oben definierte partielle Abbildung  $f$ , die jedem Knoten das eindeutig bestimmte Ende eines bei diesem Knoten startenden Weges zuordnet (bzw. undefiniert, falls der Weg nicht endet), eine partielle Funktion von  $(\mathbf{Cmd} \times \Sigma)$  nach  $\Sigma$  definiert. Ordnet diese Funktion dem Paar  $(c, \sigma)$  das Element  $\sigma'$  zu, so schreiben wir

$$\mathcal{C}[[c]]\sigma = \sigma'.$$

Die so definierte Funktion  $\mathcal{C}$  nennen wir auch  $\mathcal{C}_{\text{SOS}}$ .

Wir definieren nun induktiv die Kantenmenge, die wir mit  $\rightarrow_1$  bezeichnen, da es sich um die Einzelschrittsemantik handelt:

**skip**-Regel:

$$\frac{}{(\mathbf{skip}, \sigma) \rightarrow_1 \sigma}$$

Zuweisungs-Regel:

$$\frac{\mathcal{A}[[a]]\sigma = m}{(X := a, \sigma) \rightarrow_1 \sigma[m/X]}$$

(Dabei ist  $\sigma[m/X](X) = m$  und für alle  $Y \neq X$  gilt  $\sigma[m/X](Y) = \sigma(Y)$ .)

Hintereinanderausführungs-Regeln:

$$\frac{(c_1, \sigma) \rightarrow_1 \sigma'}{(c_1; c_2, \sigma) \rightarrow_1 (c_2, \sigma')}$$

$$\frac{(c_1, \sigma) \rightarrow_1 (c'_1, \sigma')}{(c_1; c_2, \sigma) \rightarrow_1 (c'_1; c_2, \sigma')}$$

**if-then-else**-Regeln:

$$\frac{\mathcal{B}[[b]]\sigma = \text{true}}{(\text{if } b \text{ then } c_1 \text{ else } c_2 \text{ fi}, \sigma) \rightarrow_1 (c_1, \sigma)}$$

$$\frac{\mathcal{B}[[b]]\sigma = \text{false}}{(\text{if } b \text{ then } c_1 \text{ else } c_2 \text{ fi}, \sigma) \rightarrow_1 (c_2, \sigma)}$$

**while**-Regeln:

$$\frac{\mathcal{B}[[b]]\sigma = \text{true}}{(\text{while } b \text{ do } c_1 \text{ od}, \sigma) \rightarrow_1 (c_1; \text{while } b \text{ do } c_1 \text{ od}, \sigma)}$$

$$\frac{\mathcal{B}[[b]]\sigma = \text{false}}{(\text{while } b \text{ do } c_1 \text{ od}, \sigma) \rightarrow_1 (\text{skip}, \sigma)}$$

Wir können nun zeigen:

1. Alle Elemente  $\sigma \in \Sigma$  haben als Knoten unseres Transitionssystems Ausgangsgrad 0, und
2. Alle Elemente  $(c, \sigma) \in \mathbf{Cmd} \times \Sigma$  haben als Knoten des Transitionssystems Ausgangsgrad 1.

Daraus folgt dann, dass das Transitionssystem deterministisch ist und die Abbildung  $\mathcal{C}_{\text{SOS}}$  eine wohldefinierte partielle Funktion vom Typ  $\mathbf{Cmd} \times \Sigma \rightarrow^p \Sigma$  darstellt.

Beweis, dass im Transitionssystem für die Einzelschrittsemantik alle  $\sigma \in \Sigma$  Ausgangsgrad 0 haben, alle anderen Knoten Ausgangsgrad 1:

Zu  $\sigma \in \Sigma$ : Betrachte alle Regeln. Man sieht, dass niemals eine Transition der Form  $\sigma \rightarrow_1 \dots$  definiert wird.

Also ist der Ausgangsgrad von  $\sigma$  immer 0 (für alle  $\sigma \in \Sigma$ ).

Zu  $(c, \sigma) \in \mathbf{Cmd} \times \Sigma$ :

1. Fall:  $c = \mathbf{skip}$ ; die einzigen Regeln, die eine Transition

$$(\mathbf{skip}, \sigma) \rightarrow_1 \dots$$

erzeugen, sind die **skip**-Regeln. Diese erzeugen für jedes  $\sigma \in \Sigma$  genau eine Transition

$$(\mathbf{skip}, \sigma) \rightarrow_1 \sigma.$$

Für  $c = \mathbf{skip}$  hat also  $(c, \sigma)$  den Ausgangsgrad 1 (für alle  $\sigma \in \Sigma$ ).

2. Fall:  $c = (X := a)$ ; die einzigen Regeln, die eine Transition

$$(X := a, \sigma) \rightarrow_1 \dots$$

erzeugen, sind die Zuweisungsregeln. Für jede Variable  $X$ , jeden arithmetischen Ausdruck  $a$  und jedes  $\sigma \in \Sigma$  gibt es genau eine Zuweisungsregel, die eine Transition

$$(X := a, \sigma) \rightarrow_1 \sigma'$$

erzeugt, nämlich die mit  $\sigma' = \sigma[m/X]$ , wobei

$$m = \mathcal{A}[[a]]\sigma$$

ist.

Also ist der Ausgangsgrad von  $(X := a, \sigma)$  in unserem Transitionssystem 1 für alle Variablen  $X$ , alle  $a \in \mathbf{Aexp}$  und alle  $\sigma \in \Sigma$ .

3. Fall:  $c = c_1; c_2$ ; wir nehmen induktiv an, dass der Ausgangsgrad von  $(c_i, \tau)$  gleich 1 ist für alle  $\tau \in \Sigma$  und  $i = 1, 2$ . Es existieren zwei Regeltypen, die Regeln der Form

$$(c_1; c_2, \sigma) \rightarrow_1 \dots$$

erzeugen können. Der Ausgangsgrad von  $(c_1, \sigma)$  ist 1, d.h. es gibt genau ein  $\sigma'$  oder ein  $(c'_1, \sigma')$  mit  $(c_1, \sigma) \rightarrow_1 \sigma'$  bzw.  $(c_1, \sigma) \rightarrow_1 (c'_1, \sigma')$ . Da eine Transition  $(c_1; c_2, \sigma) \rightarrow_1 \dots$  nach den Regeln bereits festlegt, wie  $\sigma'$  bzw.  $(c'_1, \sigma')$  aussieht (induktiver Rückschritt auf  $(c_1, \sigma)$ !), gibt es genau eine Transition dieser Form.

Also ist für alle  $c_1, c_2 \in \mathbf{Cmd}$  und alle  $\sigma \in \Sigma$  der Ausgangsgrad von  $(c_1; c_2, \sigma)$  gleich 1.

4. Fall:  $c = \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2 \mathbf{ fi}$ ; es gibt genau zwei Regeltypen, in denen eine Transition der Form  $(c, \sigma) \rightarrow_1 \dots$  definiert werden kann. Da  $\mathcal{B}[[b]]\sigma = \mathbf{true}$  oder  $\mathcal{B}[[b]]\sigma = \mathbf{false}$  nur von  $b$  und  $\sigma$  abhängt, kann genau eine der Regelformen Anwendung finden.

Dann gibt es aber genau ein Ziel, das erreichbar ist, nämlich  $(c_1, \sigma)$  bzw.  $(c_2, \sigma)$ . Das Ziel ist durch  $(c, \sigma)$  eindeutig bestimmt.

Also ist der Ausgangsgrad von  $(c, \sigma)$  gleich 1 für alle **if**-Anweisungen  $c$  und alle  $\sigma \in \Sigma$ .

5. Fall:  $c = \mathbf{while } b \mathbf{ do } c \mathbf{ od}$ ; analog zum Fall der **if**-Anweisungen können wir schließen, dass genau eine Transition der Form

$$(c, \sigma) \rightarrow_1 \dots$$

existiert.

Also ist der Ausgangsgrad von  $(c, \sigma)$  gleich 1 für alle **while**-Anweisungen  $c$  und alle  $\sigma \in \Sigma$ .

Damit ist gezeigt, dass die SOS-Semantik die behauptete Eigenschaft hat.

Als Anwendung der SOS-Einzelschrittsemantik kann man z.B. zeigen, dass in der Sprache IMP die Anweisungen  $c_1; (c_2; c_3)$  und  $(c_1; c_2); c_3$  äquivalent sind. („Assoziativität der Hintereinanderausführung“ – Übungsaufgabe!)

### 4.8.3 Ergebnissemantik

Die Ergebnissemantik, oder auch Bigstep-Semantik, führt die Auswertung einer Anweisung nicht schrittweise durch, sondern ordnet direkt jedem Paar  $(c, \sigma)$  eine neue Belegung  $\sigma'$  zu (falls möglich).

Man benutzt auch hier ein Transitionssystem, die Knotenmenge ist dieselbe wie bei der Einzelschrittsemantik, d.h.  $(\mathbf{Cmd} \times \Sigma) \cup \Sigma$ .

Die Kantenmenge muss neu definiert werden, diesmal wird sie mit  $\rightarrow$  bezeichnet. Die Einzelheiten findet der interessierte Leser im Skript der Grundlagenvorlesung „Formale Semantik“.

## 4.9 Denotationale Semantik

Wir lösen uns jetzt von dem Prinzip der „Ausführung“ eines Programms. Stattdessen identifizieren wir ein Programm mit der von ihm berechneten Funktion.

Als Auswertungsschema für Ausdrücke (arithmetische und boolesche) werden wir die Funktionen  $\mathcal{A}$  und  $\mathcal{B}$  wie bisher verwenden.

Die Anweisungen erhalten direkt eine Bedeutung in Form einer partiellen Funktion von  $\Sigma$  nach  $\Sigma$ :

$$\mathcal{C} : \mathbf{Cmd} \rightarrow (\Sigma \rightarrow^p \Sigma).$$

Sinnvollerweise soll  $\mathcal{C}[[c]]\sigma = \mathcal{C}_{\text{SOS}}(c, \sigma)$  für alle  $c \in \mathbf{Cmd}$  und  $\sigma \in \Sigma$  gelten.

**skip:**

$$\mathcal{C}[[\mathbf{skip}]] = \text{id}_{\Sigma}$$

**Zuweisung:**

$$\mathcal{C}[[X := a]] = \lambda\sigma.\sigma[\mathcal{A}[[a]]\sigma/X]$$

(d.h. die Funktion, die in Abhängigkeit von  $\sigma$  das Element  $\sigma[\mathcal{A}[[a]]\sigma/X]$  als Funktionswert annimmt.)

Wir prüfen für  $c = \mathbf{skip}$  und  $c = (X := a)$  die Gültigkeit von

$$\mathcal{C}[[c]]\sigma = \mathcal{C}_{\text{SOS}}(c, \sigma).$$

(Die Beweise sind sehr einfach – leichte Übungsaufgabe...)

Hintereinanderausführung:

$$\mathcal{C}[[c_1; c_2]] = \mathcal{C}[[c_2]] \circ \mathcal{C}[[c_1]].$$

Wir prüfen, dass  $\mathcal{C}[[c_1; c_2]]\sigma = \mathcal{C}_{\text{SOS}}(c_1; c_2, \sigma)$  gilt.

(Hier ist der Beweis eine Nuance schwieriger, aber immer noch eine unterdurchschnittlich schwere Übungsaufgabe.)

**if:** Wir definieren zunächst die folgende Funktion:

$$\text{cond}: (\Sigma \rightarrow \mathbb{B}) \times (\Sigma \rightarrow^p \Sigma) \times (\Sigma \rightarrow^p \Sigma) \rightarrow (\Sigma \rightarrow^p \Sigma)$$

$$(\beta, f, g) \mapsto \text{cond}(\beta, f, g),$$

wobei  $\text{cond}(\beta, f, g)(\sigma) = f(\sigma)$ , falls  $\beta(\sigma) = \mathbf{true}$ , und  $\text{cond}(\beta, f, g)(\sigma) = g(\sigma)$ , falls  $\beta(\sigma) = \mathbf{false}$ .

Nun definieren wir für  $c = \mathbf{if\ } b \mathbf{\ then\ } c_1 \mathbf{\ else\ } c_2 \mathbf{\ fi}$

$$\mathcal{C}[[c]] = \text{cond}(\mathcal{B}[[b]], \mathcal{C}[[c_1]], \mathcal{C}[[c_2]]).$$

Wir prüfen, dass damit für alle  $\sigma \in \Sigma$  gilt:

$$\mathcal{C}[[c]]\sigma = \mathcal{C}_{\text{SOS}}(c, \sigma).$$

(Man unterscheidet die Fälle  $\mathcal{B}[[b]]\sigma$  gleich **false** bzw. **true** und greift induktiv auf die entsprechende Aussage für  $c_2$  bzw.  $c_1$  zurück.)

**while:** Wir betrachten  $w = \mathbf{while\ } b \mathbf{\ do\ } c \mathbf{\ od}$  und gehen davon aus, dass  $\mathcal{C}[[c]]$  schon definiert ist. Ferner soll für alle  $\sigma \in \Sigma$  gelten:

$$\mathcal{C}[[c]]\sigma = \mathcal{C}_{\text{SOS}}(c, \sigma).$$

Wir wissen auch

$$\mathcal{C}_{\text{SOS}}(w, \sigma) = \mathcal{C}_{\text{SOS}}(\mathbf{if\ } b \mathbf{\ then\ } c; w \mathbf{\ else\ skip\ fi}, \sigma),$$

also

$$\mathcal{C}_{\text{SOS}}[[w]] = \text{cond}(\mathcal{B}[[b]], \mathcal{C}_{\text{SOS}}[[w]] \circ \mathcal{C}[[c]], \mathcal{C}[[\mathbf{skip}]]).$$

$\mathcal{C}_{\text{SOS}}[[w]]$  taucht hier links und rechts auf, löst also die folgende Gleichung:

$$f = \text{cond}(\mathcal{B}[[b]], f \circ \mathcal{C}[[c]], \text{id}_{\Sigma}).$$

Wir definieren den folgenden Operator:

$$\Gamma_{b,c}: (\Sigma \rightarrow^p \Sigma) \rightarrow (\Sigma \rightarrow^p \Sigma)$$

durch

$$\Gamma_{b,c}(f) = \text{cond}(\mathcal{B}[[b]], f \circ \mathcal{C}[[c]], \text{id}_{\Sigma}).$$

Wir sehen  $\Gamma_{b,c}$  als Funktion an in der auf der Grundmenge  $(\Sigma \rightarrow^p \Sigma)$  definierten vollständigen Halbordnung, wobei die Ordnungsrelation so definiert ist:  $f \sqsubseteq g$  genau dann, wenn aus „ $f(x)$  definiert“ folgt, dass auch  $g(x)$  definiert ist und  $f(x) = g(x)$  gilt.

Offenbar ist  $\Gamma_{b,c}(\mathcal{C}_{\text{SOS}}[[w]]) = \mathcal{C}_{\text{SOS}}[[w]]$ , das heißt  $\mathcal{C}_{\text{SOS}}[[w]]$  ist Fixpunkt von  $\Gamma_{b,c}$ . Wir definieren nun  $\mathcal{C}[[w]]$  als den kleinsten Fixpunkt von  $\Gamma_{b,c}$ . Dieser existiert und ist eindeutig bestimmt (folgt aus dem Fixpunktsatz von Kleene).

Auch hier findet man die Einzelheiten im Skript zur Grundlagenvorlesung „Formale Semantik“.

## 4.10 Axiomatische Semantik

### 4.10.1 Grundidee

Wir beschreiben die Bedeutung eines Programms, indem wir eine Vor- und eine Nachbedingung angeben:

$$\{A\} c \{B\}$$

$A$ : Vorbedingung

$c \in \mathbf{Cmd}$

$B$ : Nachbedingung

Die Bedeutung von  $\{A\} c \{B\}$  sei:

Wenn vor Ausführung von  $c$  die Bedingung  $A$  erfüllt ist und  $c$  terminiert, dann ist nach Ausführung von  $c$  die Bedingung  $B$  erfüllt.

Als Beispiel verwenden wir das Fakultätsprogramm `faku`:

$\{X = N\}$	$Y := 1;$	$\{X = N \wedge Y = 1\}$
$\{Y \cdot X! = N!\}$	<b>while</b> $X > 1$ <b>do</b>	
$\{Y \cdot X! = N!\}$	$Y := Y \cdot X;$	$\{Y \cdot X! = N! \cdot X\}$
$\{Y \cdot X! = N! \cdot X\}$	$X := X - 1$	$\{Y \cdot X! = N!\}$
	<b>od</b>	$\{Y \cdot X! = N! \wedge X \leq 1\}$
		$\{Y = N!\}$

Die Aussage  $\{A\} c \{B\}$  ist eine

**partielle Korrektheitsaussage,**

da wir nur etwas aussagen über den Fall, dass  $c$  terminiert.

Analog nennt man  $[A] c [B]$  eine

**totale Korrektheitsaussage**

mit der Bedeutung: Wenn  $A$  erfüllt ist und  $c$  ausgeführt wird, dann terminiert  $c$  und anschließend ist  $B$  erfüllt.

Wir müssen festlegen, welche Art von Aussagen für  $A$  und  $B$  in Frage kommen.

Wir definieren zunächst eine erweiterte Form der Menge **Aexp** von arithmetischen Ausdrücken:

$$a ::= n \mid X \mid i \mid a_0 + a_1 \mid a_0 - a_1 \mid a_0 \cdot a_1$$

mit  $n \in \mathbb{N}$ ,  $X \in \mathbf{Loc}$ ,  $i \in \mathbf{Intvar}$ .

**Intvar** ist eine Menge von Integer-Variablen, z.B.

$$\mathbf{Intvar} = \{i_1, i_2, i_3, \dots\}.$$

Wir definieren die Menge **Assn** von Zusicherungen (englisch: assertions):

$$\begin{aligned} A ::= & \mathbf{true} \mid \mathbf{false} \mid a_0 = a_1 \mid a_0 < a_1 \mid \\ & a_0 \leq a_1 \mid a_0 > a_1 \mid a_0 \geq a_1 \mid \neg A \mid \\ & A_1 \wedge A_2 \mid A_1 \vee A_2 \mid A_1 \Rightarrow A_2 \mid \\ & A_1 \Leftrightarrow A_2 \mid \forall i : A \mid \exists i : A \end{aligned}$$

(Klammerungen werden gesetzt, wo immer es notwendig oder erwünscht ist...)

Die Menge der Variablen ist nun

$$\mathbb{V} = \mathbf{Loc} \cup \mathbf{Intvar}.$$

In einer Zusicherung  $A \in \mathbf{Assn}$  können die Integer-Variablen gebunden oder frei vorkommen. Wir verzichten an dieser Stelle auf eine exakte Definition des freien Vorkommens von Variablen und appellieren an die Intuition (oder die Kenntnisse aus der Logik).

Auswertung von Zusicherungen:

Gegeben sei  $A \in \mathbf{Assn}$ ,  $\sigma \in \Sigma$  und es gelte  $FV(A) = \emptyset$ . Dann kann man  $A$  unter  $\sigma$  auswerten:

$$\sigma(A) \in \mathbb{B}.$$

Für Zusicherungen mit freien Variablen werden diese zunächst „interpretiert“, d.h. es wird ihnen eine natürliche Zahl als Wert zugeordnet, durch die Interpretation  $I: \mathbf{Intvar} \rightarrow \mathbb{N}$ . Wir schreiben

$$\sigma \models^I A,$$

falls die durch  $I$  interpretierte Zusicherung  $A$  im Speicherzustand  $\sigma$  wahr ist, also

$$\sigma(I(A)) = \mathbf{true}.$$

Wir schreiben

$$\sigma \models A,$$

falls  $\sigma \models^I A$  für alle Interpretationen  $I$  gilt.

**Beispiel:** Wir wollen eine Formel  $A \in \mathbf{Assn}$  konstruieren, die eine freie Variable  $i$  enthält und die genau dann erfüllt ist, wenn  $i$  eine Primzahl ist, d.h.

$$(\sigma \models^I A) \Leftrightarrow (I(i) \text{ ist Primzahl})$$

Lösung:

Wir nehmen die folgende Formel:

$$A \equiv \forall j : ((j \leq 1) \vee (j \geq i) \vee (\forall k : j \cdot k \neq i))$$

Sei nämlich  $I(i) = n \in \mathbb{N}$ . Dann gilt

$$I(A) = \forall j : ((j \leq 1) \vee (j \geq n) \vee (\forall k : j \cdot k \neq n))$$

1. Sei  $n$  eine Primzahl,  $j \in \mathbb{N}$  gegeben. Falls ein  $k$  existiert mit  $j \cdot k = n$ , dann gilt  $j = 1$  oder  $j = n$ .

Falls kein solches  $k$  existiert, ist  $\forall k : j \cdot k \neq n$  wahr. Also gilt  $\forall j : (\dots)$  und daher  $\sigma \models^I A$ .

2. Sei  $n$  keine Primzahl, etwa  $n = a \cdot b$  mit  $1 < a < n$ . Wähle  $j = a$ , dann gilt  $\neg(j \leq 1) \wedge \neg(j \geq n) \wedge \exists k : j \cdot k = n$ . Also ist  $\sigma \models^I A$  falsch.

**Beispiel:** Konstruiere eine Formel  $\text{KGV} \in \mathbf{Assn}$  mit freien Variablen  $i, j, k$ , die wahr ist genau dann, wenn  $i$  das kleinste gemeinsame Vielfache von  $j$  und  $k$  ist:

$$(\sigma \models^I A) \Leftrightarrow (\text{kgv}(I(j), I(k)) = I(i))$$

Lösung:

$$\begin{aligned} \text{KGV} \equiv & (\exists j' : j' \cdot j = i) \wedge (\exists k' : k' \cdot k = i) \wedge \\ & (\forall i' : (((\exists j' : j' \cdot j = i') \wedge (\exists k' : k' \cdot k = i')) \\ & \Rightarrow (i \leq i'))) \end{aligned}$$

Zu zeigen:

1.  $I(i) = n, I(j) = m, I(k) = r$  und  $I(\text{KGV})$  erfüllt; dann ist  $n = \text{kgv}(m, r)$ .
2.  $I(i) = n, I(j) = m, I(k) = r$  und  $n = \text{kgv}(m, r)$ ; dann ist  $I(\text{KGV})$  erfüllt.

Anmerkung: Unsere kurze Bedingung

$$\sigma(I(A)) = \mathbf{true}$$

kann auch formalisiert werden. (Wie?)

Wir wollen nicht ständig unterscheiden zwischen Definiiertheit und Nichtdefiniiertheit. Daher betrachten wir Funktionen von  $\Sigma_{\perp} \rightarrow \Sigma_{\perp}$ , die  $\perp$  auf  $\perp$  abbilden, anstelle der partiellen Funktionen  $\Sigma \rightarrow^p \Sigma$ . ( $\perp$  ist hier ein sogenanntes *bottom*-Symbol, das den undefinierten Wert ersetzt; dabei ist  $\perp$  ein minimales Element in der zugrundegelegten Ordnung.)

Wir verwenden folgende Vereinbarung:

**Konvention:**  $\perp \models^I A$  ist wahr für alle Interpretationen  $I$  und alle Zusicherungen  $A$ . Anders ausgedrückt:

$$\perp(I(A)) = \mathbf{true}.$$

Insbesondere gilt also auch

$$\perp \models \mathbf{false}.$$

### Gültigkeit der partiellen Korrektheitsaussage

Für  $\sigma \in \Sigma_{\perp}$  und  $I: \mathbf{Intvar} \rightarrow \mathbb{N}$  ist die Aussage  $\{A\}c\{B\}$  mit  $A, B \in \mathbf{Assn}$  und  $c \in \mathbf{Cmd}$  wahr, falls

$$(\sigma \models^I A) \Rightarrow (\mathcal{C}[[c]]\sigma \models^I B)$$

gilt. Wir schreiben dann

$$\sigma \models^I \{A\}c\{B\}$$

Ferner schreiben wir

$$\models^I \{A\}c\{B\}$$

für  $\forall \sigma \in \Sigma_{\perp} : \sigma \models^I \{A\}c\{B\}$ .

Schließlich schreiben wir

$$\models \{A\}c\{B\}$$

falls für alle  $I: \mathbf{Intvar} \rightarrow \mathbb{N}$  gilt  $\models^I \{A\}c\{B\}$ . In diesem Fall sagen wir, die partielle Korrektheitsaussage  $\{A\}c\{B\}$  sei gültig.

## Beispiele:

1.  $\{i < X\} X := X - 1 \{i \leq X\}$  ist gültig.

2. Für alle  $A, B \in \mathbf{Assn}$  und

$w = \mathbf{while\ true\ do\ skip\ od}$

gilt:

$\models \{A\}w\{B\}$

3. Die Aussage  $\{X = 1\}p\{Y = 5050\}$  ist wahr, wenn  $p$  das folgende Programm ist:

$Y := 0;$

$\mathbf{while\ } X \leq 100 \mathbf{\ do}$

$Y := Y + X;$

$X := X + 1;$

$\mathbf{od}$

## 4.10.2 Hoare-Logik

Die Hoare-Logik definiert eine Menge ableitbarer Aussagen der Form  $\{A\}c\{B\}$  für  $A, B \in \mathbf{Assn}$  und  $c \in \mathbf{Cmd}$  durch Regeln:

1. **skip**:

$$\frac{}{\{A\} \text{ skip } \{A\}}$$

für alle  $A \in \mathbf{Assn}$ .

2. Zuweisung:

$$\frac{}{\{A[a/X]\} X := a \{A\}}$$

für alle  $A \in \mathbf{Assn}$ ,  $a \in \mathbf{Aexp}$ ,  $X \in \mathbf{Loc}$ .

3. Sequenz:

$$\frac{\{A\} c_1 \{C\} \quad \{C\} c_2 \{B\}}{\{A\} c_1; c_2 \{B\}}$$

für alle  $A, B, C \in \mathbf{Assn}$ ,  $c_1, c_2 \in \mathbf{Cmd}$ .

4. **if**:

$$\frac{\{A \wedge b\} c_1 \{B\} \quad \{A \wedge \neg b\} c_2 \{B\}}{\{A\} \text{ if } b \text{ then } c_1 \text{ else } c_2 \text{ fi } \{B\}}$$

für alle  $A, B \in \mathbf{Assn}$ ,  $b \in \mathbf{Bexp}$ ,  $c_1, c_2 \in \mathbf{Cmd}$ .

5. **while**:

$$\frac{\{A \wedge b\} c \{A\}}{\{A\} \text{ while } b \text{ do } c \text{ od } \{A \wedge \neg b\}}$$

für alle  $A \in \mathbf{Assn}$ ,  $b \in \mathbf{Bexp}$ ,  $c \in \mathbf{Cmd}$ .

6. Konsequenz:

$$\frac{\models (A \Rightarrow A') \quad \{A'\} c \{B'\} \quad \models (B' \Rightarrow B)}{\{A\} c \{B\}}$$

für alle  $A, A', B, B' \in \mathbf{Assn}$ ,  $c \in \mathbf{Cmd}$ .

**Definition 5.2.1** Wir schreiben  $\vdash \{A\}c\{B\}$ , falls sich die Aussage  $\{A\}c\{B\}$  aus den Regeln der Hoare-Logik ableiten läßt.

Als Beispiel betrachten wir die Anweisung

$w = \mathbf{while\ true\ do\ skip\ od}$

Damit gilt

$\vdash \{\mathbf{true}\}w\{\mathbf{false}\}$

Beweis:

$$\vdash \{\mathbf{true}\}\mathbf{skip}\{\mathbf{true}\}$$

Daher  $\vdash \{\mathbf{true} \wedge \mathbf{true}\}\mathbf{skip}\{\mathbf{true}\}$

Weiter  $\vdash \{\mathbf{true}\}w\{\mathbf{true} \wedge \mathbf{false}\}$

Schließlich  $\vdash \{\mathbf{true}\}w\{\mathbf{false}\}$

Ein weiteres Beispiel:

$$\vdash \{X = 3\} X := 8 + X \{X = 11\}$$

Denn aus der Zuweisungsregel folgt

$$\vdash \{(X = 11)[8 + X/X]\} X := 8 + X \{X = 11\}$$

Also

$$\vdash \{8 + X = 11\} X := 8 + X \{X = 11\}$$

und mit der Konsequenzregel

$$\vdash \{X = 3\} X := 8 + X \{X = 11\}$$

**Satz 5.2.2** Die Regeln der Hoare-Logik erzeugen nur korrekte Aussagen, d.h.

$$\vdash \{A\}c\{B\} \Rightarrow \models \{A\}c\{B\}$$

**Beweis:** Es sei  $\{A\}c\{B\}$  herleitbar. Wir müssen zeigen, dass für alle  $\sigma \in \Sigma_{\perp}$  und alle  $I: \text{Intvar} \rightarrow \mathbb{N}$  gilt:

$$\sigma \models^I \{A\}c\{B\},$$

d.h. aus  $\sigma(I(A)) = \text{true}$  folgt  $\mathcal{C}[[c]]\sigma(I(B)) = \text{true}$ .

Man sieht leicht:

$$\vdash \{A\}c\{B\} \Rightarrow \vdash \{I(A)\}c\{I(B)\}$$

für alle  $I$ . Wenn wir dann folgern, dass

$$\sigma \models \{I(A)\}c\{I(B)\}$$

gilt, haben wir gezeigt, dass aus der Ableitbarkeit von  $\{A\}c\{B\}$  bereits für alle  $I$  die Korrektheit von  $\{I(A)\}c\{I(B)\}$  folgt, und damit die Korrektheit von  $\{A\}c\{B\}$ .

Also bleibt zu zeigen:

Seien  $A, B \in \mathbf{Assn}$  ohne freie Variablen, und sei  $\{A\}c\{B\}$  herleitbar. Dann gilt

$$\models \{A\}c\{B\}.$$

Unter den gemachten Voraussetzungen ist also nachzuprüfen, dass für alle  $\sigma \in \Sigma_{\perp}$  mit  $\sigma' = \mathcal{C}[[c]]\sigma$  gilt

$$\sigma(A) \text{ wahr} \Rightarrow \sigma'(B) \text{ wahr.}$$

Sei  $\{A\}c\{B\}$  durch die **skip**-Regel entstanden, d.h.  $\{A\}\mathbf{skip}\{B\}$ . Dann ist  $\sigma' = \sigma$  und  $B = A$ , also ist die Behauptung trivial.

Sei nun  $c$  die Zuweisung  $X := a$  für  $X \in \mathbf{Loc}$  und  $a \in \mathbf{Aexp}$ . Dann ist  $A = B[n/X]$  mit  $n = \sigma(a)$ , und es gilt  $\sigma' = \sigma[n/X]$ . Offensichtlich gilt

$$\sigma(B[n/X]) = \sigma[n/X](B).$$

(Beweis durch Induktion über die Erzeugung von  $B$ ), also ist  $\sigma(A) = \sigma(B[n/X])$  wahr genau dann, wenn  $\sigma'(B) = \sigma[n/X](B)$  wahr ist. Insbesondere also

$$\sigma(A) \text{ wahr} \Rightarrow \sigma'(B) \text{ wahr.}$$

Da  $\sigma' = \mathcal{C}[[c]]\sigma$  gilt, haben wir die Gültigkeit der Zusicherung  $\{A\}c\{B\}$  gezeigt.

Nun sei  $c = c_1; c_2$ . Dann gibt es eine Zusicherung  $C$  ohne freie Variablen (Beweis: Übung!), so dass  $\{A\}c_1\{C\}$  und  $\{C\}c_2\{B\}$  herleitbar sind. Induktiv folgt

$$\sigma(A) \text{ wahr} \Rightarrow \mathcal{C}[[c_1]]\sigma(C) \text{ wahr}$$

und

$$\sigma'(C) \text{ wahr} \Rightarrow \mathcal{C}[[c_2]]\sigma'(B) \text{ wahr.}$$

Wir wählen  $\sigma' = \mathcal{C}[[c_1]]\sigma$  und beachten, dass dann

$$\mathcal{C}[[c_2]]\sigma' = \mathcal{C}[[c_2]] \circ \mathcal{C}[[c_1]]\sigma = \mathcal{C}[[c_1; c_2]]\sigma$$

gilt. Also erhalten wir, dass aus  $\sigma(A)$  wahr folgt  $\sigma'(C)$  wahr und schließlich  $\sigma''(B)$  wahr, mit

$$\sigma'' = \mathcal{C}[[c_1; c_2]]\sigma = \mathcal{C}[[c]]\sigma.$$

Insbesondere ergibt sich aus  $\sigma(A)$  wahr  $\Rightarrow \sigma''(B)$  wahr die Gültigkeit von  $\{A\}_c\{B\}$ .

Sei  $c = \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2 \mathbf{ fi}$ . Aus der Ableitbarkeit von  $\{A\}_c\{B\}$  folgt die Ableitbarkeit von  $\{A \wedge b\}_{c_1}\{B\}$  und von  $\{A \wedge \neg b\}_{c_2}\{B\}$ . Damit erhalten wir

$$\sigma(A \wedge b) \text{ wahr} \Rightarrow \mathcal{C}[[c_1]]\sigma(B) \text{ wahr}$$

und

$$\sigma(A \wedge \neg b) \text{ wahr} \Rightarrow \mathcal{C}[[c_2]]\sigma(B) \text{ wahr.}$$

Sei  $\sigma(b)$  wahr. Dann gilt  $\mathcal{C}[[c]]\sigma = \mathcal{C}[[c_1]]\sigma$ , also

$$\sigma(A) \text{ wahr} \Rightarrow \mathcal{C}[[c]]\sigma(B) \text{ wahr.}$$

Sei  $\sigma(b)$  falsch. Dann gilt  $\mathcal{C}[[c]]\sigma = \mathcal{C}[[c_2]]\sigma$ , und daher

$$\sigma(A) \text{ wahr} \Rightarrow \mathcal{C}[[c]]\sigma(B) \text{ wahr.}$$

Diese Implikation gilt also in jedem Fall.

Nun sei  $c = \mathbf{while } b \mathbf{ do } c_1 \mathbf{ od}$ . Die Herleitbarkeit von  $\{A\}c\{B\}$  bedeutet, dass  $\{A \wedge b\}c_1\{A\}$  herleitbar ist und dass  $B = A \wedge \neg b$  gilt. Wir benötigen  $\sigma' = \mathcal{C}[[c]]\sigma$ . Nach der denotationalen Semantik der while-Schleife gilt

$$\sigma' = \text{fix}(\Gamma_{b,c_1})(\sigma)$$

mit  $\Gamma_{b,c_1}(f) = \text{cond}(\mathcal{B}[[b]], f \circ \mathcal{C}[[c_1]], \text{id}_\Sigma)$  für  $f \in \Sigma_\perp \rightarrow \Sigma_\perp$ . Den Fixpunkt kann man als Supremum der folgenden Menge erhalten:

$$\{\Gamma_{b,c_1}^{(n)}(\perp) \mid n > 0\}.$$

Die folgende Behauptung kann durch Induktion über  $n$  bewiesen werden:

Für alle  $n \geq 0$  und alle  $\sigma, \sigma' \in \Sigma_\perp$  gilt

$$\Gamma_{b,c_1}^{(n)}(\perp)(\sigma) = \sigma', \quad \sigma(A) \text{ wahr}$$

$$\Rightarrow \sigma'(A \wedge \neg b) \text{ wahr.}$$

Die Induktionsbehauptung ist damit erfüllt. Wir wissen also: Wenn  $\mathcal{C}[[c]]\sigma$  definiert ist, dann folgt aus  $\sigma(A)$  wahr auch  $\mathcal{C}[[c]]\sigma(A \wedge \neg b)$  wahr. Wenn aber  $\mathcal{C}[[c]]\sigma$  nicht definiert ist, dann ist die Zusicherung  $\{A\}_c\{B\}$  für dieses  $\sigma$  ohnehin erfüllt. Das komplettiert den Beweis für die while-Schleife. Es fehlt noch der Fall, dass der letzte Schritt der Herleitung von  $\{A\}_c\{B\}$  eine Anwendung der Konsequenz-Regel war. Es seien also  $A \Rightarrow A'$  und  $B' \Rightarrow B$  wahr (unter allen Belegungen  $\sigma \in \Sigma$ ) und  $\{A'\}_c\{B'\}$  herleitbar, also unter allen  $\sigma$  gültig (durch Induktionsannahme).

Dann ist die herleitbare Formel  $\{A\}_c\{B\}$  auch gültig, denn es gilt für alle  $\sigma \in \Sigma$ :

$\sigma(A)$  wahr  $\Rightarrow$

$\sigma(A')$  wahr  $\Rightarrow$

$\mathcal{C}[[c]]\sigma(B')$  wahr  $\Rightarrow$

$\mathcal{C}[[c]]\sigma(B)$  wahr.

## Beispiele:

1. Das Plateau-Problem: Eingabe sei ein Feld

$a[1..n]$ , sortiert.

Als Ausgabe wollen wir die Zahl  $p$  berechnen, wobei

$$p = \max\{q \mid \exists i : a[i] = a[i + 1] = \dots = a[i + q - 1]\}$$

Lösung:

```
plateau  $\equiv$  j := 1;  
          p := 1;  
          while j < n do  
            j := j + 1;  
            if a[j] = a[j - p] then  
              p := p + 1  
            else skip  
            fi  
          od
```

Beweis der Korrektheit:

Wir formulieren die Bedingung  $mp(j, p)$ :

Auf dem Teilfeld  $a[1..j]$  hat das längste Plateau die Länge  $p$ .

Wir wollen die Herleitbarkeit von

$$\{\mathbf{true}\} \text{ plateau } \{mp(n, p)\}$$

zeigen. Als Vorbedingung verwenden wir nicht  $\{\mathbf{true}\}$ , sondern

$$V = (mp(1, 1) \text{ und „a ist sortiert“}).$$

Es gilt offenbar, dass  $V$  erfüllt ist (unter den Voraussetzungen  $n \geq 1$  und  $a$  ist sortiert).

Durch zweimaliges Anwenden der Zuweisungsregel und einmaliges Anwenden der Sequenz-Regel erhalten wir:

$$\{mp(1, 1), a \text{ sortiert}\} \text{ } j := 1; p := 1 \{mp(j, p), a \text{ sortiert}\}.$$

Mit nochmaliger Verwendung der Sequenz-Regel müssen wir nur noch zeigen, dass

$$\{mp(j, p), a \text{ sortiert}\} \text{ } w \{mp(n, p)\}$$

für  $w = \text{while}$ -Schleife aus dem Programm `plateau` herleitbar ist. Von jetzt an kürzen wir die Bedingung „a sortiert“ ab durch  $A$ . Sicher sind die beiden folgenden Aussagen herleitbar:

$$\{mp(j, p + 1) \wedge A\} p := p + 1 \{mp(j, p) \wedge A\}$$

und

$$\{mp(j, p) \wedge A\} \text{skip} \{mp(j, p) \wedge A\}.$$

Damit behaupten wir

$$\{mp(j - 1, p) \wedge A\} \text{if...fi} \{mp(j, p) \wedge A\}.$$

Denn: Wir unterscheiden zwei Fälle.

1. Fall:  $a[j]=a[j-p]$  (d.h.  $b$  wahr):

$$\{mp(j - 1, p) \wedge A \wedge b\} \Rightarrow \{mp(j, p + 1) \wedge A \wedge b\}.$$

Also folgt aus

$$\{mp(j, p + 1) \wedge A \wedge b\} p := p + 1 \{mp(j, p) \wedge A \wedge b\}$$

mit der Konsequenzregel

$$\{mp(j - 1, p) \wedge A \wedge b\} p := p + 1 \{mp(j, p) \wedge A\}.$$

2. Fall:  $a[j] \neq a[j-p]$  (d.h.  $b$  falsch):

Es folgt ähnlich

$$\{mp(j-1, p) \wedge A \wedge \neg b\} \text{ skip } \{mp(j, p) \wedge A\}.$$

In jedem Fall erhalten wir

$$\{mp(j-1, p) \wedge A\} \text{ if...fi } \{mp(j, p) \wedge A\}.$$

Also mit der Sequenzregel:

$$\{mp(j, p) \wedge A\} j := j + 1; \text{ if...fi } \{mp(j, p) \wedge A\}.$$

Also erhalten wir für die gesamte while-Schleife  $w$  die Herleitbarkeit von

$$\{mp(j, p) \wedge A\} w \{mp(j, p) \wedge A \wedge (j = n)\},$$

und daher mit der Sequenzregel für das gesamte Plateau-Programm:

$$\{\text{true}\} \text{ plateau } \{mp(j, p) \wedge A \wedge (j = n)\}$$

und mit der Konsequenzregel wie gewünscht:

$$\{\text{true}\} \text{ plateau } \{mp(n, p)\}.$$

## 2. Potenzieren:

```
Z := 1;
while ¬(Y=0) do
  while even(Y) do
    X := X·X;
    Y := Y/2
  od;
  Z := Z·X;
  Y := Y-1
od
```

Behauptung: Für obiges Programm  $c$  gilt  $\{A\} c \{B\}$ , wobei

$$A = \{X = m \wedge Y = n\},$$

$$B = \{Z = m^n\}.$$

(Achtung:  $B$  ist so nicht in **Assn** definiert!)

Beweis: Sei  $w$  die äußere while-Schleife in  $c$ , d.h.

$$c = (Z := 1; w).$$

Wir zeigen

$$\{X = m \wedge Y = n\} Z := 1 \{X = m \wedge Y = n \wedge Z = 1\}$$

und

$$\{X = m \wedge Y = n \wedge Z = 1\} w \{Z = m^n\},$$

indem wir zeigen, dass

$$\{Z \cdot X^Y = m^n\}$$

eine Invariante für beide while-Schleifen ist.

Die Termination des Programms ist klar. Also ist das Programm korrekt.

## Zur Vollständigkeit des Hoare Kalküls:

Unser Wunsch wäre ein Algorithmus, der für gegebenes Programm  $c \in \mathbf{Cmd}$  und gegebene Zusicherungen  $A, B \in \mathbf{Assn}$  die Frage der Gültigkeit von  $\{A\} c \{B\}$  entscheidet.

Etwas bescheidener: Aufzählung aller gültigen Aussagen der Form  $\{A\} c \{B\}$ .

Beides ist unmöglich.

Den Beweis dieser Behauptung kann man auf zwei Arten führen.

1. Betrachte die Aussage

$$\{\mathbf{true}\} \mathbf{skip} \{B\}.$$

Diese ist wahr genau dann, wenn  $B \in \mathbf{Assn}$  eine wahre Aussage ist. Ein Verfahren zur Entscheidung oder Aufzählung aller gültigen Aussagen der Hoare-Logik ergäbe also insbesondere ein Aufzählungsverfahren für alle wahren Aussagen der Arithmetik.

Nach dem

### **Gödelschen Unvollständigkeitssatz**

existiert ein solches Verfahren nicht!

2. Betrachte die Aussage

$$\{\mathbf{true}\} c \{\mathbf{false}\}.$$

Diese ist wahr genau dann, wenn  $c$  auf allen Speicherbelegungen nicht terminiert. Gäbe es hierfür ein Aufzählungsverfahren, so wäre das Halteproblem entscheidbar.

Wir wissen aber, dass das nicht der Fall ist, also ist die Menge

$$\{(A, B, c) \mid \models \{A\} c \{B\}\}$$

nicht rekursiv aufzählbar.

Wir beobachten nun allerdings, dass die Konsequenzregel einen nicht-effektiven Schritt erlaubt:

$$\frac{\{\mathbf{true}\} \mathbf{skip} \{\mathbf{true}\} \quad \mathbf{true} \Rightarrow B}{\{\mathbf{true}\} \mathbf{skip} \{B\}}.$$

Die Regel ist „anwendbar“, wenn  $B$  wahr ist. Also sind alle Regeln der Form

$$\{\mathbf{true}\} \mathbf{skip} \{B\}$$

für wahre  $B$  herleitbar, obwohl wir diese Herleitung im allgemeinen nicht finden können.

Daher ist es kein Widerspruch, wenn man zeigt, dass aus der Gültigkeit

$$\models \{A\} c \{B\}$$

die Herleitbarkeit

$$\vdash \{A\} c \{B\}$$

folgt, d.h. eine Aussage ist genau dann gültig, wenn sie im Hoare-Kalkül herleitbar ist. (Der Beweis ist aufwändig und wird deshalb hier weggelassen.)

Die Tatsache, dass wir trotzdem keine formale Überprüfbarkeit der Gültigkeit von Aussagen  $\{A\} c \{B\}$  bekommen, wird angedeutet, indem man von

**relativer Vollständigkeit**

spricht.