Theoretische Informatik
Übungsblatt 4 (2023W)
Lösungsvorschlag

Sei Σ das Programm

```
while x > 1 do {
  y := x/2;
  if x = 2 * y then
    x := y
  else
    x := 3 * x + 1
}
```

Aufgabe 4.1
Zeigen Sie, dass das Programm Σ nach Hinzufügen einiger Klammernpaare ein syntaktisch korrektes SIMPLE($\mathbb{Z}$)-Programm ist.
Anleitung: Geben Sie eine Parallelableitung an.

Lösung 4.1

Programm
$\Rightarrow_P$ "`while`" *Term* "`do`" *Programm*
$\Rightarrow_P$ "`while (`" *Term BinOp Term* "`) do {`" *Programm* "`;`" *Programm* "`}`"
$\Rightarrow_P$ "`while (`" *Var* "`>`" *Const* "`) do {`" *Var* "`:=`" *Term* "`;`"
"`if`" *Term* "`then`" *Programm* "`else`" *Programm* "`}`"
$\Rightarrow_P$ "`while (x > 1) do {y := (`" *Term BinOp Term* "`);`"
"`if (`" *Term BinOp Term* "`) then`" *Var* "`:=`" *Term* "`else`" *Var* "`:=`" *Term* "`}`"
$\Rightarrow_P$ "`while (x > 1) do {y := (`" *Var* "`/`" *Const* "`);`"
"`if (`" *Var* "`= (`" *Term BinOp Term* "`)) then x :=`" *Var* "`else x := (`" *Term BinOp Term* "`)}`"
$\Rightarrow_P$ "`while (x > 1) do {y := (x/2);`"
"`if (x = (`" *Const* "`*`" *Var* "`)) then x := y else x := ((`" *Term BinOp Term* "`) + `" *Const* "`)}`"
$\Rightarrow_P$ "`while (x > 1) do {y := (x/2); if (x = (2 * y)) then x := y else x := ((`" *Const* "`*`" *Var* "`) + 1)}`"
$\Rightarrow_P$ "`while (x > 1) do {y := (x/2); if (x = (2 * y)) then x := y else x := ((3 * x) + 1)}`"

Aufgabe 4.2
Sei σ ein Zustand mit $\sigma(x) = 2$. Berechnen Sie $[\Sigma]\sigma$ mittels der strukturellen operationalen Semantik von SIMPLE($\mathbb{Z}$).
Anleitung: Teile der Auswertung, die vorangegangene Schritte in ähnlicher Form wiederholen, können abgekürzt werden.

Lösung 4.2
Um das Ergebnis des Programms, $[\Sigma]\sigma$, zu erhalten, müssen wir mit der Konfiguration (Σ, σ) beginnen und solange Rechenschritte durchführen, bis wir eine finale Konfiguration erreichen.

$$
\begin{aligned}
&(\Sigma, \sigma) = (\texttt{while } x > 1 \texttt{ do } \{\cdots\}, \sigma) \\
&\qquad [x > 1]\sigma = ([x]\sigma > [1]\sigma) = (2 > 1) = 1 \\
&\quad \Rightarrow (\{\{y := x/2; \texttt{if} \cdots\}; \texttt{while } x > 1 \texttt{ do } \{\cdots\}\}, \sigma) \\
&\qquad (\{y := x/2; \texttt{if} \cdots\}, \sigma) \\
&\qquad\qquad (y := x/2, \sigma) \Rightarrow \sigma_1, \text{ wobei } \sigma_1(v) = \begin{cases} [x/2]\sigma = ([x]\sigma/[2]\sigma) = 2/2 = 1 & \text{für } v = y \\ \sigma(v) & \text{sonst} \end{cases} \\
&\qquad \Rightarrow (\texttt{if} \cdots, \sigma_1)
\end{aligned}
$$

$\Rightarrow (\{\texttt{if } x = 2 * y \texttt{ then} \cdots \texttt{;while } x > 1 \texttt{ do } \{\cdots\}\}, \sigma_1)$

$(\texttt{if } x = 2 * y \texttt{ then} \cdots, \sigma_1)$

$[x = 2 * y]\sigma_1 = ([x]\sigma_1 = [2 * y]\sigma_1) = ([x]\sigma = [2]\sigma_1 * [y]\sigma_1) = (2 = 2 * 1) = 1$

$\Rightarrow (x \coloneqq y, \sigma_1)$

$\Rightarrow (\{x \coloneqq y\texttt{;while } x > 1 \texttt{ do } \{\cdots\}\}, \sigma_1)$

$(x \coloneqq y, \sigma_1) \Rightarrow \sigma_2$, wobei $\sigma_2(v) = \begin{cases} [y]\sigma_1 = 1 & \text{für } v = x \\ \sigma_1(v) & \text{sonst} \end{cases}$

$\Rightarrow (\texttt{while } x > 1 \texttt{ do } \{\cdots\}, \sigma_2)$

$[x > 1]\sigma_2 = ([x]\sigma_2 > [1]\sigma_2) = (1 > 1) = 0$

$\Rightarrow \sigma_2$

Somit gilt $[\Sigma]\sigma = \sigma_2$, da $(\Sigma, \sigma) \overset{*}{\Rightarrow} \sigma_2$.

Aufgabe 4.3

Sei σ ein Zustand mit $\sigma(x) = 5$. Berechnen Sie $[\Sigma]\,\sigma$ mittels der natürlichen Semantik von $\mathsf{SIMPLE}(\mathbb{Z})$.
Anleitung: Teile der Auswertung, die vorangegangene Schritte in ähnlicher Form wiederholen, können abgekürzt werden.

Lösung 4.3

Die natürliche Semantik erlaubt es, $[\Sigma]\,\sigma$ direkt zu berechnen.

$[\Sigma]\,\sigma = [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma$

$[x > 1]\sigma = ([x]\sigma > [1]\sigma) = (5 > 1) = 1$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\{y \coloneqq x/2\texttt{;if} \cdots\}]\,\sigma$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\texttt{if} \cdots]\,[y \coloneqq x/2]\,\sigma$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\texttt{if} \cdots]\,\sigma_1$, wobei $\sigma_1(v) = \begin{cases} [x/2]\sigma = ([x]\sigma/[2]\sigma) = 5/2 = 2 & \text{für } v = y \\ \sigma(v) & \text{sonst} \end{cases}$

$[x = 2 * y]\sigma_1 = ([x]\sigma_1 = [2 * y]\sigma_1) = ([x]\sigma = [2]\sigma_1 * [y]\sigma_1) = (5 = 2 * 2) = 0$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[x \coloneqq 3 * x + 1]\,\sigma_1$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma_2$, wobei

$\sigma_2(v) = \begin{cases} [3 * x + 1]\sigma_1 = ([3 * x]\sigma_1 + [1]\sigma_1) = ([3]\sigma_1 * [x]\sigma_1 + 1) = (3 * 5 + 1) = 16 & \text{für } v = x \\ \sigma_1(v) & \text{sonst} \end{cases}$

$[x > 1]\sigma_2 = ([x]\sigma_2 > [1]\sigma_2) = (16 > 1) = 1$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\{y \coloneqq x/2\texttt{;if} \cdots\}]\,\sigma_2$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\texttt{if} \cdots]\,[y \coloneqq x/2]\,\sigma_2$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[\texttt{if} \cdots]\,\sigma_2$, wobei $\sigma_3(v) = \begin{cases} [x/2]\sigma_2 = \cdots = 8 & \text{für } v = y \\ \sigma_2(v) & \text{sonst} \end{cases}$

$[x = 2 * y]\sigma_3 = \cdots = 1$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,[x \coloneqq y]\,\sigma_3$

$= [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma_4$, wobei $\sigma_4(v) = \begin{cases} [y]\sigma_3 = 8 & \text{für } v = x \\ \sigma_1(v) & \text{sonst} \end{cases}$

$[x > 1]\sigma_4 = \cdots = (8 > 1) = 1$

$= \cdots = [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma_5$, wobei $\sigma_5(v) = \begin{cases} 4 & \text{für } v \in \{x, y\} \\ \sigma(v) & \text{sonst} \end{cases}$

$[x > 1]\sigma_5 = \cdots = (4 > 1) = 1$

$= \cdots = [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma_6$, wobei $\sigma_6(v) = \begin{cases} 2 & \text{für } v \in \{x, y\} \\ \sigma(v) & \text{sonst} \end{cases}$

$[x > 1]\sigma_6 = \cdots = (2 > 1) = 1$

$= \cdots = [\texttt{while } x > 1 \texttt{ do } \{\cdots\}]\,\sigma_7$, wobei $\sigma_7(v) = \begin{cases} 1 & \text{für } v \in \{x, y\} \\ \sigma(v) & \text{sonst} \end{cases}$

$[x > 1]\sigma_5 = \cdots = (1 > 1) = 0$

$= \sigma_7$

Wir erhalten als Ergebnis $[\Sigma]\,\sigma = \sigma_7$.

Aufgabe 4.4

Zeigen Sie, dass die Aussage $\{x \geq 1\}\,\Sigma\,\{x = 1\}$ partiell korrekt ist. Leiten Sie die Korrektheitsaussage mittels der Regeln und Axiome des Hoare-Kalküls ab. Argumentieren Sie, warum die entstehenden Implikationen gültig sind.

Anleitung: Als Invariante ist $x \geq 1$ geeignet.

Lösung 4.4

Wir zerlegen die Korrektheitsaussage nach den Regeln des Hoare-Kalküls. Die Ableitung lässt sich von oben nach unten als eine Reihe von Regelanwendungen lesen, bei denen man von wahren Korrektheitsaussagen und gültigen Formeln auf neue wahre Korrektheitsaussagen schließt. Konstruiert wird die Ableitung aber von unten nach oben, beginnend mit der zu beweisenden Korrektheitsaussage.

$$\dfrac{\dfrac{\dfrac{P \wedge x = 2 * y \supset Q \quad \{Q\}\,x \coloneqq y\,\{Inv\}}{\{P \wedge x = 2 * y\}\,x \coloneqq y\,\{Inv\}}\,\text{(zw)} \quad \dfrac{P \wedge x \neq 2 * y \supset R \quad \{R\}\,x \coloneqq 3 * x + 1\,\{Inv\}}{\{P \wedge x \neq 2 * y\}\,x \coloneqq 3 * x + 1\,\{Inv\}}\,\text{(zw)}}{\{P\}\,\texttt{if } x = 2 * y \texttt{ then } x \coloneqq y \texttt{ else } x \coloneqq 3 * x + 1\,\{Inv\}}\,\text{(imp)}\,\text{(if)} \quad \dfrac{\{Inv \wedge x > 1\}\,y \coloneqq x/2\,\{P\}}{\{Inv \wedge x > 1\}\,\{y \coloneqq x/2;\ \texttt{if} \cdots\}\,\{Inv\}}\,\text{(zw')}\,\text{(ha)}}{\{Inv\}\,\texttt{while } x > 1 \texttt{ do } \{y \coloneqq x/2;\ \texttt{if} \cdots\}\,\{Inv \wedge x \not> 1\}}\,\text{(wh)} \quad \dfrac{x \geq 1 \supset Inv \qquad Inv \wedge x \not> 1 \supset x = 1}{\{x \geq 1\}\,\texttt{while } x > 1 \texttt{ do } \{\cdots\}\,\{x = 1\}}\,\text{(imp)}$$

Bei dieser Zerlegung kamen folgende Überlegungen zur Anwendung.

- Die `while`-Anweisung kann nur mit der Regel (wh) hergeleitet werden. Dafür benötigen wir aber die Vorbedingung Inv und die Nachbedingung $Inv \wedge \neg e$. Diese stellen wir her, indem wir mit der Implikationsregel beginnen.
- Bei der Hintereinanderausführungsregel (ha) wählen wir die Interpolante P so, dass die Korrektheitsaussage im linken Zweig zu einer Instanz des Axioms (zw′) wird. Das bietet sich an, weil die Variable y weder in der Vorbedingung $Inv \wedge x > 1$ (mit $x \geq 1$ als Invariante) noch im Ausdruck $x/2$ auf der rechten Seite der Zuweisung vorkommt. Die Nachbedingung der Zuweisung (und die Vorbedingung der `if`-Anweisung) ist also die Formel P: $Inv \wedge x > 1 \wedge (y = x/2)$.
- Bei den beiden Zuweisungen in der `if`-Anweisung haben wir, anders als bei der Interpolanten P, nicht die Möglichkeit, eine Formel geeignet zu wählen. Wir zerlegen die beiden Korrektheitsaussagen mit den Zuweisungen also zunächst mit der Implikationsregel, um dann Q und R so zu wählen, dass wir Instanzen des Axioms (zw) erhalten, d.h., Q: $Inv\left[\begin{smallmatrix} y \\ x \end{smallmatrix}\right]$ und R: $Inv\left[\begin{smallmatrix} 3*x+1 \\ x \end{smallmatrix}\right]$.

Die Ableitung ist korrekt, wenn wir zeigen können, dass die vier Formeln, die die Implikationsregel einführt, gültig sind.

- $x \geq 1 \supset Inv$

  $x \geq 1 \supset x \geq 1$

  Diese Formel ist eine Tautologie der Form $F \supset F$.
- $Inv \wedge x \not> 1 \supset x = 1$

  $x \geq 1 \wedge x \leq 1 \supset x = 1$

  Diese Implikation ist aufgrund der Eigenschaften der Ungleichungen wahr: aus $x \geq 1$ und $x \leq 1$ folgt $x = 1$.
- $P \wedge x = 2 * y \supset Q$

  $Inv \wedge x > 1 \wedge (y = x/2) \wedge (x = 2 * y) \supset Inv\left[\begin{smallmatrix} y \\ x \end{smallmatrix}\right]$

  $x \geq 1 \wedge x > 1 \wedge (y = x/2) \wedge (x = 2 * y) \supset x \geq 1\left[\begin{smallmatrix} y \\ x \end{smallmatrix}\right]$

  $x \geq 1 \wedge x > 1 \wedge (y = x/2) \wedge (x = 2 * y) \supset y \geq 1$

Die Prämisse $x > 1$ ist gleichbedeutend mit $x \geq 2$. Diese Verhältnis bliebt bei ganzzahliger Division durch 2 erhalten, d.h., $x/2 \geq 1$. Wegen $y = x/2$ ergibt sich daraus $y \geq 1$.

- $$P \wedge x \neq 2 * y \supset R$$
  $$Inv \wedge x > 1 \wedge (y = x/2) \wedge x \neq 2 * y \supset Inv\left[\begin{smallmatrix}3*x+1\\ x\end{smallmatrix}\right]$$
  $$x \geq 1 \wedge x > 1 \wedge (y = x/2) \wedge x \neq 2 * y \supset x \geq 1\left[\begin{smallmatrix}3*x+1\\ x\end{smallmatrix}\right]$$
  $$x \geq 1 \wedge x > 1 \wedge (y = x/2) \wedge x \neq 2 * y \supset 3 * x + 1 \geq 1$$
  Aus $x > 1$ erhalten wir durch Multiplikation mit 3 und Addition von 1 die Ungleichung $3*x+1 > 4$. Daraus folgt $3 * x + 1 \geq 1$.

Damit haben wir gezeigt, dass die angegebene Korrektheitsaussage wahr hinsichtlicher partieller Korrektheit ist: Für alle Eingaben, für die $x \geq 1$ gilt und das Programm terminiert, gilt nach der Ausführung $x = 1$.

Alternative Lösung 4.4: Statt dem Axiom (zw′) und der Regel (if) können auch (zw) und (if′) verwendet werden. In diesem Fall ergibt sich die Interpolante der Regel (ha) aus der Regel (if′) als $P\colon (x = 2y \supset Inv\left[\begin{smallmatrix}y\\ x\end{smallmatrix}\right]) \wedge (x \neq 2y \supset Inv\left[\begin{smallmatrix}3x+1\\ x\end{smallmatrix}\right])$.

$$\dfrac{\dfrac{(Inv \wedge x > 1) \supset P\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right] \quad \{P\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right]\}\, y \coloneqq x/2\, \{P\}}{\{Inv \wedge x > 1\}\, y \coloneqq x/2\, \{P\}}\ (\text{imp}) \quad \dfrac{\{Inv\left[\begin{smallmatrix}y\\ x\end{smallmatrix}\right]\}\, x \coloneqq y\, \{Inv\} \quad \{Inv\left[\begin{smallmatrix}3x+1\\ x\end{smallmatrix}\right]\}\, x \coloneqq 3 * x + 1\, \{Inv\}}{\{P\}\ \texttt{if}\ x = 2 * y\ \texttt{then}\ x \coloneqq y\ \texttt{else}\ x \coloneqq 3 * x + 1\, \{Inv\}}\ (\text{if}')}{\dfrac{x \geq 1 \supset Inv \quad \dfrac{\{Inv \wedge x > 1\}\, \{y \coloneqq x/2;\ \texttt{if} \cdots\}\, \{Inv\}}{\{Inv\}\ \texttt{while}\ x > 1\ \texttt{do}\ \{y \coloneqq x/2;\ \texttt{if} \cdots\}\, \{Inv \wedge x \not> 1\}}\ (\text{wh}) \quad Inv \wedge x \not> 1 \supset x = 1}{\{x \geq 1\}\ \texttt{while}\ x > 1\ \texttt{do}\ \{\cdots\}\, \{x = 1\}}\ (\text{imp})}$$

Zusätzlich zu den Implikationen $x \geq 1 \supset Inv$ und $Inv \wedge x \not> 1 \supset x = 1$, die bereits oben diskutiert wurden, müssen wir die Gültigkeit der Formel zeigen, die sich aus der oberen Anwendung der Implikationsregel ergibt.

$$\begin{aligned}
&(Inv \wedge x > 1) \supset P\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right] \\
&(Inv \wedge x > 1) \supset \big((x = 2y \supset Inv\left[\begin{smallmatrix}y\\ x\end{smallmatrix}\right]) \wedge (x \neq 2y \supset Inv\left[\begin{smallmatrix}3x+1\\ x\end{smallmatrix}\right])\big)\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right] \\
&(x \geq 1 \wedge x > 1) \supset \big((x = 2y \supset (x \geq 1)\left[\begin{smallmatrix}y\\ x\end{smallmatrix}\right]) \wedge (x \neq 2y \supset (x \geq 1)\left[\begin{smallmatrix}3x+1\\ x\end{smallmatrix}\right])\big)\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right] \\
&(x \geq 1 \wedge x > 1) \supset \big((x = 2y \supset y \geq 1) \wedge (x \neq 2y \supset 3x + 1 \geq 1)\big)\left[\begin{smallmatrix}x/2\\ y\end{smallmatrix}\right] \\
&(x \geq 1 \wedge x > 1) \supset \big((x = 2(x/2) \supset x/2 \geq 1) \wedge (x \neq 2(x/2) \supset 3x + 1 \geq 1)\big)
\end{aligned}$$

Die Gültigkeit dieser Formel ergibt sich daraus, dass aus der Prämisse $x > 1$, die gleichbedeutend mit $x \geq 2$ ist, sowohl $x/2 \geq 1$ als auch $3x + 1 \geq 1$ folgt. Wenn also die linke Seite der Implikation wahr ist, erhalten wir

$$\underbrace{\big(\underbrace{(x = 2(x/2) \supset \underbrace{x/2 \geq 1}_{\text{wahr}})}_{\text{wahr}} \wedge \underbrace{(x \neq 2(x/2) \supset \underbrace{3x + 1 \geq 1}_{\text{wahr}})}_{\text{wahr}}\big)}_{\text{wahr}}$$

Aufgabe 4.5

Zeigen Sie, dass die Aussage $\{x \geq 1\}\, \Sigma\, \{x = 1\}$ partiell korrekt ist. Führen Sie den Korrektheitsbeweis, indem Sie das Programm mithilfe der Annotierungsregeln um Zustandsbeschreibungen ergänzen und dann die Gültigkeit der dadurch entstehenden Implikationen argumentieren.

Anleitung: Als Invariante ist $x \geq 1$ geeignet.

Lösung 4.5

Wir beginnen mit der zu beweisenden Korrektheitsaussage und fügen schrittweise Zustandsbeschreibungen hinzu. Wir nummerieren die Formeln in der Reihenfolge des Hinzufügens.

$\{\, \mathit{Pre} \colon x \geq 1 \,\}$
wh $\{\, F_1 \colon \mathit{Inv} \,\}$
`while` $x > 1$ `do {`
wh $\{\, F_2 \colon \mathit{Inv} \wedge x > 1 \,\}$
$y \coloneqq x/2$;
zw↓ $\{\, F_9 \colon \mathit{Inv} \wedge x > 1 \wedge (y = x/2) \,\}$
`if` $x = 2 * y$ `then`
if↓ $\{\, F_{10} \colon \mathit{Inv} \wedge x > 1 \wedge (y = x/2) \wedge (x = 2 * y) \,\}$
zw↑ $\{\, F_7 \colon \mathit{Inv}\left[{}^{y}_{x}\right] \,\}$
$x \coloneqq y$
endif↑ $\{\, F_5 \colon \mathit{Inv} \,\}$
`else`
if↓ $\{\, F_{11} \colon \mathit{Inv} \wedge x > 1 \wedge (y = x/2) \wedge (x \neq 2 * y) \,\}$
zw↑ $\{\, F_8 \colon \mathit{Inv}\left[{}^{3*x+1}_{x}\right] \,\}$
$x \coloneqq 3 * x + 1$
endif↑ $\{\, F_6 \colon \mathit{Inv} \,\}$
wh $\{\, F_3 \colon \mathit{Inv} \,\}$
`}`
wh $\{\, F_4 \colon \mathit{Inv} \wedge x \not> 1 \,\}$
$\{\, \mathit{Post} \colon x = 1 \,\}$

Die Implikationsregel verlangt nun, dass wir die Gültigkeit der Formeln $\mathit{Pre} \supset F_1$, $F_{10} \supset F_7$, $F_{11} \supset F_8$ und $F_4 \supset \mathit{Post}$ argumentieren, was wir bereits am Ende der Lösung 4.4 gemacht haben.

Alternative Lösung 4.5: Die oben angegebene Lösung entspricht der Ableitung in Lösung 4.4, daher erhält man auch dieselben Implikationen. Wendet man statt den Annotierungsregeln zw↓ und if↓ die Regeln if↑ und zw↑ an, erhalten wir ein annotiertes Programm, das der Ableitung in der alternativen Lösung 4.4 entspricht. Wie dort steht P für die Formel $(x = 2y \supset \mathit{Inv}\left[{}^{y}_{x}\right]) \wedge (x \neq 2y \supset \mathit{Inv}\left[{}^{3x+1}_{x}\right])$.

$\{\, \mathit{Pre} \colon x \geq 1 \,\}$
wh $\{\, F_1 \colon \mathit{Inv} \,\}$
`while` $x > 1$ `do {`
wh $\{\, F_2 \colon \mathit{Inv} \wedge x > 1 \,\}$
zw↑ $\{\, F_{10} \colon P\left[{}^{x/2}_{y}\right] \,\}$
$y \coloneqq x/2$;
if↑ $\{\, F_9 \colon P \,\}$
`if` $x = 2 * y$ `then`
zw↑ $\{\, F_7 \colon \mathit{Inv}\left[{}^{y}_{x}\right] \,\}$
$x \coloneqq y$
endif↑ $\{\, F_5 \colon \mathit{Inv} \,\}$
`else`
zw↑ $\{\, F_8 \colon \mathit{Inv}\left[{}^{3*x+1}_{x}\right] \,\}$
$x \coloneqq 3 * x + 1$
endif↑ $\{\, F_6 \colon \mathit{Inv} \,\}$
wh $\{\, F_3 \colon \mathit{Inv} \,\}$
`}`
wh $\{\, F_4 \colon \mathit{Inv} \wedge x \not> 1 \,\}$
$\{\, \mathit{Post} \colon x = 1 \,\}$

Die Gültigkeit der Implikation $F_2 \supset F_{10}$, also von $(\mathit{Inv} \wedge x > 1) \supset P\left[{}^{x/2}_{y}\right]$, wurde in der alternativen Lösung 4.4 argumentiert.

Aufgabe 4.6

Zeigen Sie, dass ein vollständiger Hoare-Kalkül nur eine der beiden folgenden Regeln benötigt.

$$\frac{\{F \wedge e\}\,\Pi\,\{G\} \quad \{F \wedge \neg e\}\,\Omega\,\{G\}}{\{F\}\ \texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\ \{G\}}\ (\text{if}) \qquad \frac{\{F\}\,\Pi\,\{H\} \quad \{G\}\,\Omega\,\{H\}}{\{(e \supset F) \wedge (\neg e \supset G)\}\ \texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\ \{H\}}\ (\text{if}')$$

Anleitung: Der Beweis besteht aus zwei Teilen. Zeigen Sie, dass jede Anwendung der Regel (if) in einer Ableitung durch eine Anwendung von (if′) ersetzt werden kann, und umgekehrt, dass jede Anwendung von (if′) durch (if) ersetzt werden kann.

Das Programm ist in beiden Regeln dasselbe, die Formel-Platzhalter F, G und H haben aber unterschiedliche Bedeutung. Es hilft, wenn Sie diese Platzhalter umbenennen, sodass sie in den beiden Regeln unterschiedlich heißen.

Lösung 4.6

Teil A: „Jede Anwendung der Regel (if) *kann durch eine Anwendung der Regel* (if′) *ersetzt werden.“* Angenommen, wir haben eine Ableitung, die die Regel (if) verwendet, wie die folgende.

$$\dfrac{\dfrac{\vdots}{\{F' \wedge e\}\,\Pi\,\{G'\}} \quad \dfrac{\vdots}{\{F' \wedge \neg e\}\,\Omega\,\{G'\}}}{\dfrac{\{F'\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{G'\}}{\vdots}}\ \text{(if)}$$

Die Ableitung besteht aus den drei Teilen

$$\dfrac{\vdots}{\{F' \wedge e\}\,\Pi\,\{G'\}} \qquad \dfrac{\vdots}{\{F' \wedge \neg e\}\,\Omega\,\{G'\}} \qquad \dfrac{\{F'\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{G'\}}{\vdots}$$

die durch die Anwendung der Regel (if) verknüpft sind. Die drei Teile können wir weiterverwenden, die Regelanwendung müssen wir durch eine oder mehrere andere Regeln ersetzen. Wir nehmen die ersten beiden Teile und kombinieren sie mit der Regel (if′).

$$\dfrac{\dfrac{\vdots}{\{F' \wedge e\}\,\Pi\,\{G'\}} \quad \dfrac{\vdots}{\{F' \wedge \neg e\}\,\Omega\,\{G'\}}}{\{(e \supset (F' \wedge e)) \wedge (\neg e \supset (F' \wedge \neg e))\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{G'\}}\ \text{(if}')$$

Dabei entsprechen die Platzhalter F, G und H in der Regel den Formeln $F' \wedge e$, $F' \wedge \neg e$ bzw. G'. Wir vereinfachen die Vorbedingung.

$$\begin{aligned}
(e \supset (F' \wedge e)) \wedge (\neg e \supset (F' \wedge \neg e)) &= (\neg e \vee (F' \wedge e)) \wedge (e \vee (F' \wedge \neg e)) \\
&= ((\neg e \vee F') \wedge (\neg e \vee e)) \wedge ((e \vee F') \wedge (e \vee \neg e)) \\
&= (\neg e \vee F') \wedge 1 \wedge (e \vee F') \wedge 1 \\
&= (\neg e \vee F') \wedge (e \vee F') \\
&= (\neg e \wedge e) \vee F' \\
&= 0 \vee F' \\
&= F'
\end{aligned}$$

Nach der Vereinfachung entspricht die Korrektheitsaussage genau jener im dritten Ableitungsteil, den wir damit direkt anfügen können.

$$\dfrac{\dfrac{\dfrac{\vdots}{\{F' \wedge e\}\,\Pi\,\{G'\}} \quad \dfrac{\vdots}{\{F' \wedge \neg e\}\,\Omega\,\{G'\}}}{\{(e \supset (F' \wedge e)) \wedge (\neg e \supset (F' \wedge \neg e))\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{G'\}}\ \text{(if}')}{\{F'\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{G'\}}$$
$$\vdots$$

Bei der Vereinfachung handelt es sich um keine Regel, sondern nur um eine logische Umformung der Vorbedingung.

Teil B: „Jede Anwendung der Regel (if′) *kann durch eine Anwendung der Regel* (if) *ersetzt werden.“* Angenommen, wir haben eine Ableitung, die die Regel (if′) verwendet, wie die folgende.

$$\dfrac{\dfrac{\vdots}{\{F'\}\,\Pi\,\{H'\}} \quad \dfrac{\vdots}{\{G'\}\,\Omega\,\{H'\}}}{\{(e \supset F') \wedge (\neg e \supset G')\}\,\texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{H'\}}\ \text{(if}')$$
$$\vdots$$

Diese Ableitung besteht aus den drei Teilen

$$\frac{\vdots}{\{F'\}\,\Pi\,\{H'\}} \qquad \frac{\vdots}{\{G'\}\,\Omega\,\{H'\}} \qquad \frac{\{(e \supset F') \wedge (\neg e \supset G')\}\ \texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{H'\}}{\vdots}$$

die durch eine Anwendung der Regel (if′) verknüpft sind. Die drei Teile können wir weiterverwenden, die Regelanwendung müssen wir durch eine oder mehrere andere Regeln ersetzen. Wir nehmen den dritten Teil und erweitern ihn mit der Regel (if).

$$\frac{\dfrac{\{(e \supset F') \wedge (\neg e \supset G') \wedge e\}\,\Pi\,\{H'\} \quad \{(e \supset F') \wedge (\neg e \supset G') \wedge \neg e\}\,\Omega\,\{H'\}}{\{(e \supset F') \wedge (\neg e \supset G')\}\ \texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{H'\}}\ (\text{if})}{\vdots}$$

Dabei entsprechen die Platzhalter F und G in der Regel den Formeln $(e \supset F') \wedge (\neg e \supset G')$ bzw. H'. Wir vereinfachen die Vorbedingungen der beiden Prämissen.

$$\begin{aligned}
(e \supset F') \wedge (\neg e \supset G') \wedge e &= (\neg e \vee F') \wedge (e \vee G') \wedge e && \text{Absorption: } (e \vee \cdots) \wedge e = e \\
&= (\neg e \vee F') \wedge e && \text{Distributivgesetz} \\
&= (\neg e \wedge e) \vee (F' \wedge e) \\
&= 0 \vee (F' \wedge e) \\
&= F' \wedge e \\
(e \supset F') \wedge (\neg e \supset G') \wedge \neg e &= (\neg e \vee F') \wedge (e \vee G') \wedge \neg e && \text{Absorption: } (\neg e \vee \cdots) \wedge \neg e = \neg e \\
&= (e \vee G') \wedge \neg e && \text{Distributivgesetz} \\
&= (e \wedge \neg e) \vee (G' \wedge \neg e) \\
&= 0 \vee (G' \wedge \neg e) \\
&= G' \wedge \neg e
\end{aligned}$$

Die Prämissen der (if)-Regel vereinfachen sich damit zu $\{F' \wedge e\}\,\Pi\,\{H'\}$ und $\{G' \wedge \neg e\}\,\Omega\,\{H'\}$. Deren Wahrheit lässt sich mit den anderen beiden Teilen der ursprünglichen Ableitung und zwei Anwendungen der Implikationsregel zeigen.

$$\frac{\dfrac{\dfrac{(F' \wedge e) \supset F' \quad \dfrac{\vdots}{\{F'\}\,\Pi\,\{H'\}}}{\{F' \wedge e\}\,\Pi\,\{H'\}}\ (\text{imp}) \qquad \dfrac{\dfrac{(G' \wedge \neg e) \supset G' \quad \dfrac{\vdots}{\{G'\}\,\Omega\,\{H'\}}}{\{G' \wedge \neg e\}\,\Omega\,\{H'\}}\ (\text{imp})}{\{(e \supset F') \wedge (\neg e \supset G') \wedge e\}\,\Pi\,\{H'\} \qquad \{(e \supset F') \wedge (\neg e \supset G') \wedge \neg e\}\,\Omega\,\{H'\}}}{\{(e \supset F') \wedge (\neg e \supset G')\}\ \texttt{if}\ e\ \texttt{then}\ \Pi\ \texttt{else}\ \Omega\,\{H'\}}\ (\text{if})}{\vdots}$$

Wir haben somit gezeigt, dass jede Anwendung einer if-Regel in einer Ableitung lokal gegen die andere ausgetauscht werden kann. Lässt sich also eine Korrektheitsaussage mit Hilfe der einen Regel herleiten, geht das auch mit der anderen. Ein vollständiger Hoare-Kalkül benötigt daher nur eine der beiden Regeln.

Aufgabe 4.7

Verwenden Sie die Definition von partieller bzw. totaler Korrektheit um festzustellen, für welche Programme Π die folgenden Korrektheitsaussagen partiell bzw. total korrekt sind.

(a) $\{1\}\,\Pi\,\{1\}$

(b) $\{1\}\,\Pi\,\{0\}$

(c) $\{0\}\,\Pi\,\{1\}$

(d) $\{0\}\,\Pi\,\{0\}$

Anleitung: Das sind in Summe acht Fragen (vier verschiedene Kombinationen von Vor- und Nachbedingungen, jeweils untersucht hinsichtlich partieller und totaler Korrektheit), die sich aber gemeinsam mit

nur wenigen Fallunterscheidungen beantworten lassen. Es sind nicht konkrete Programme gefragt, die die Korrektheitsaussagen wahr werden lassen, sondern die notwendigen und hinreichenden Eigenschaften, die ein Programm erfüllen muss, sodass die Aussagen wahr sind.

Lösung 4.7

Partielle Korrektheit: Eine Aussage $\{F\}\,\Pi\,\{G\}$ ist partiell korrekt, wenn für alle Zustände $\sigma \in \mathcal{S}$ gilt:

[PK] Wenn $[F]\sigma$ wahr und $\sigma' = [\Pi]\sigma$ definiert ist, dann ist $[G]\sigma' = [G][\Pi]\sigma$ wahr.

In den Unteraufgaben (a), (c) und (d) ist entweder die Vorbedingung falsch oder die Nachbedingung wahr (oder beides), daher trifft die Implikation [PK] unabhängig von der Wahl des Programms Π immer zu, die Aussage ist daher für beliebiges Π partiell korrekt.

In Unteraufgabe (b) ist die Vorbedingung wahr und die Nachbedingung falsch, die Implikation [PK] vereinfacht sich zu „Wenn $[\Pi]\sigma$ definiert ist, dann falsch." Wegen $F \supset 0 \equiv \neg F$ ist das gleichbedeutend mit „$[\Pi]\sigma$ ist nicht definiert". Somit ist die Korrektheitsaussage $\{1\}\,\Pi\,\{0\}$ genau für jene Programme Π korrekt, die für keine Eingabe terminieren.

Totale Korrektheit: Eine Aussage $\{F\}\,\Pi\,\{G\}$ ist total korrekt, wenn für alle Zustände $\sigma \in \mathcal{S}$ gilt:

[TK] Wenn $[F]\sigma$ wahr ist, dann ist $\sigma' = [\Pi]\sigma$ definiert und $[G]\sigma' = [G][\Pi]\sigma$ wahr.

In den Unteraufgaben (c) und (d) gilt $[F]\sigma = [0]\sigma = 0$, damit ist der Wenn-Teil immer falsch, die Implikation also erfüllt. Die Korrektheitsaussagen sind daher unabhängig vom Programm Π wahr.

In den Unteraufgaben (a) und (b) gilt $[F]\sigma = [1]\sigma = 1$, damit vereinfacht sich die Implikation [TK] zu

[TK′] $\sigma' = [\Pi]\sigma$ ist definiert und $[G]\sigma' = [G][\Pi]\sigma$ wahr.

Für Unteraufgabe (a) gilt $[G]\sigma' = [1]\sigma' = 1$, womit sich [TK′] auf „$[\Pi]\sigma$ ist definiert" reduziert. Die Korrektheitsaussage $\{1\}\,\Pi\,\{1\}$ ist daher total korrekt, wenn $[\Pi]\sigma$ für alle σ definiert ist, wenn also Π für jede Eingabe terminiert.

Für Unteraufgabe (b) hingegen gilt $[G]\sigma' = [0]\sigma' = 0$, womit [TK′] unabhängig vom Programm Π immer falsch ist. Somit ist die Korrektheitsaussage $\{1\}\,\Pi\,\{0\}$ für kein Programm Π total korrekt.

Zusammengefasst erhalten wir:

(a) Die Korrektheitsaussage $\{1\}\,\Pi\,\{1\}$ ist für alle Programme Π partiell korrekt und für Programme, die immer terminieren, auch total korrekt.

(b) Die Korrektheitsaussage $\{1\}\,\Pi\,\{0\}$ ist für Programme Π, die nie terminieren, partiell korrekt, aber für kein Programm total korrekt.

(c) Die Korrektheitsaussage $\{0\}\,\Pi\,\{1\}$ ist für alle Programme Π partiell und total korrekt.

(d) Die Korrektheitsaussage $\{0\}\,\Pi\,\{0\}$ ist für alle Programme Π partiell und total korrekt.

Aufgabe 4.8

Zeigen Sie, dass die folgende Aussage total korrekt ist.

$\{\, x = x_0 \wedge x \geq 0 \,\}$
`{` $y \coloneqq 3 * x$`;`
`while` $2 * x \neq y$ `do {`
  $x \coloneqq x + 1$`;`
  $y \coloneqq y + 1$
`}}`
$\{\, x = 2 * x_0 \,\}$

Anleitung: Als Invariante ist die Formel $y = 2 * x_0 + x \wedge y \geq 2 * x$ geeignet, als Variante der Ausdruck $y - 2 * x$. Argumentieren Sie, warum die auftretenden Formeln gültig sind.

Lösung 4.8

Wir beginnen damit, der zu beweisenden Korrektheitsaussage schrittweise Zustandsbeschreibungen hinzuzufügen. Wir nummerieren die Formeln in der Reihenfolge des Hinzufügens.

$\{\, \mathit{Pre}\colon x = x_0 \wedge x \geq 0 \,\}$
`{` $y \coloneqq 3 * x$ `;`
zw↓ $\{\, F_7\colon x = x_0 \wedge x \geq 0 \wedge y = 3 * x \,\}$
wht $\{\, F_1\colon \mathit{Inv} \,\}$
`while` $2 * x \neq y$ `do {`
wht $\{\, F_2\colon \mathit{Inv} \wedge 2 * x \neq y \wedge t = t_0 \,\}$
zw↑ $\{\, F_6\colon (\mathit{Inv} \wedge 0 \leq t < t_0)\left[\begin{smallmatrix} y+1 \\ y \end{smallmatrix}\right]\left[\begin{smallmatrix} x+1 \\ x \end{smallmatrix}\right] \,\}$
$x \coloneqq x + 1$ `;`
zw↑ $\{\, F_5\colon (\mathit{Inv} \wedge 0 \leq t < t_0)\left[\begin{smallmatrix} y+1 \\ y \end{smallmatrix}\right] \,\}$
$y \coloneqq y + 1$
wht $\{\, F_3\colon \mathit{Inv} \wedge 0 \leq t < t_0 \,\}$
`}}`
wht $\{\, F_4\colon \mathit{Inv} \wedge 2 * x = y \,\}$
$\{\, \mathit{Post}\colon x = 2 * x_0 \,\}$

Die Implikationsregel verlangt nun, dass wir die Gültigkeit der Formeln $F_7 \supset F_1$, $F_2 \supset F_6$ und $F_4 \supset \mathit{Post}$ nachweisen. Dem Hinweis in der Angabe folgend verwenden wir die Invariante $\mathit{Inv}\colon y = 2{*}x_0{+}x \wedge y \geq 2{*}x$ und die Variante $t = y - 2 * x$.

- $F_7 \supset F_1$

$$x = x_0 \wedge x \geq 0 \wedge y = 3x \supset \mathit{Inv}$$
$$x = x_0 \wedge x \geq 0 \wedge y = 3x \supset y = 2x_0 + x \wedge y \geq 2x$$

  Wir müssen zeigen, dass die beiden Bedingungen auf der rechten Seite erfüllt sind, wobei wir voraussetzen können, dass die linke Seite der Implikation gilt.

  $y = 2x_0 + x$**:** Aus $y = 3x$ erhalten wir $y = 2x + x$ und mittels $x = x_0$ die gewünschte Gleichung $y = 2x_0 + x$.

  $y \geq 2x$**:** Aus $x \geq 0$ erhalten wir durch Addition von $2x$ auf beiden Seiten $3x \geq 2x$ und mittels $y = 3x$ die Bedingung $y \geq 2x$.

- $F_2 \supset F_6$

$$\mathit{Inv} \wedge 2x \neq y \wedge t = t_0 \supset (\mathit{Inv} \wedge 0 \leq t < t_0)\left[\begin{smallmatrix} y+1 \\ y \end{smallmatrix}\right]\left[\begin{smallmatrix} x+1 \\ x \end{smallmatrix}\right]$$
$$y = 2x_0{+}x \wedge y \geq 2x \wedge 2x \neq y \wedge y{-}2x = t_0 \supset (y = 2x_0{+}x \wedge y \geq 2x \wedge 0 \leq y{-}2x < t_0)\left[\begin{smallmatrix} y+1 \\ y \end{smallmatrix}\right]\left[\begin{smallmatrix} x+1 \\ x \end{smallmatrix}\right]$$
$$y = 2x_0{+}x \wedge y \geq 2x \wedge 2x \neq y \wedge y{-}2x = t_0 \supset (y{+}1 = 2x_0{+}x \wedge y{+}1 \geq 2x \wedge 0 \leq y{+}1{-}2x < t_0)\left[\begin{smallmatrix} x+1 \\ x \end{smallmatrix}\right]$$
$$y = 2x_0{+}x \wedge y \geq 2x \wedge 2x \neq y \wedge y{-}2x = t_0 \supset y{+}1 = 2x_0{+}x{+}1 \wedge y{+}1 \geq 2(x{+}1) \wedge 0 \leq y{+}1{-}2(x{+}1) < t_0$$

  Es ist die Gültigkeit von vier Bedingungen zu zeigen.

  $y{+}1 = 2x_0{+}x{+}1$ ergibt sich aus der Prämisse $y = 2x_0{+}x$ durch Addition von 1 auf beiden Seiten.

  $y{+}1 \geq 2(x{+}1)$**:** Aus den Prämissen $y \geq 2x$ und $2x \neq y$ können wir $y \geq 2x{+}1$ schließen, woraus sich durch Addition von 1 auf beiden Seiten $y{+}1 \geq 2x{+}2 = 2(x{+}1)$ ergibt.

  $0 \leq y{+}1{-}2(x{+}1)$**:** Diese Bedingung ist äquivalent zu $2(x{+}1) \leq y{+}1$, also zur vorigen Formel, deren Gültigkeit wir bereits gezeigt haben.

  $y{+}1{-}2(x{+}1) < t_0$**:** Aus der Prämisse $y{-}2x = t_0$ folgt $y{-}2x{-}1 < t_0$. Da $y{+}1{-}2(x{+}1) = y{-}2x{-}1$ gilt, folgt die Bedingung.

- $F_4 \supset \mathit{Post}$

$$\mathit{Inv} \wedge 2x = y \supset x = 2x_0$$
$$y = 2x_0 + x \wedge y \geq 2x \wedge 2x = y \supset x = 2x_0$$

  Aus der ersten Prämisse ergibt sich unter Verwendung von $2x = y$ die Gleichung $2x = 2x_0 + x$, die sich zu $x = 2x_0$ vereinfacht.

Damit haben wir gezeigt, dass das gegebene Programm für nicht-negative Anfangswerte von x terminiert. Nach dem Programm hat x das Doppelte seines Anfangswertes.

Alternativlösung 4.8: Statt zw↓ kann auch die Annotierungsregel zw↑ verwendet werden. Dann ändern sich die ersten Zeilen des annotierten Programms zu

$\{\, \mathit{Pre}\colon x = x_0 \wedge x \geq 0 \,\}$
zw↑ $\{\, F_7\colon \mathit{Inv}\left[\begin{smallmatrix}3*x\\ y\end{smallmatrix}\right] \,\}$
$\{\; y := 3 * x\,;$
wht $\{\, F_1\colon \mathit{Inv} \,\}$

Anstelle von $F_7 \supset F_1$ ist dann die Gültigkeit folgender Implikation zu zeigen.

- $\mathit{Pre} \supset F_7$

$$(x = x_0 \wedge x \geq 0) \supset \mathit{Inv}\left[\begin{smallmatrix}3*x\\ y\end{smallmatrix}\right]$$
$$(x = x_0 \wedge x \geq 0) \supset (y = 2 * x_0 + x \wedge y \geq 2 * x)\left[\begin{smallmatrix}3*x\\ y\end{smallmatrix}\right]$$
$$(x = x_0 \wedge x \geq 0) \supset (3 * x = 2 * x_0 + x \wedge 3 * x \geq 2 * x)$$

Wir müssen zeigen, dass die beiden Bedingungen auf der rechten Seite erfüllt sind, wobei wir voraussetzen können, dass die linke Seite der Implikation gilt.

$3x = 2x_0 + x$**:** gilt wegen der Prämisse $x = x_0$, da $3x = 2x + x$.

$3x \geq 2x$**:** ergibt sich aus der Prämisse $x \geq 0$ durch Addition von $2x$ auf beiden Seiten.