

Grundlagen der Theoretischen Informatik

Sebastian Iwanowski
FH Wedel

Kap. 2: Verifikationstechniken
Teil 1: Verifikation mit Hoare-Tripeln

Programmentwicklung

Konstruktionsproblem:

Gegeben eine Spezifikation:

Funktion, die einem Argument einen Funktionswert zuordnet

Entwirf ein Programm, das jedes Argument des Definitionsbereichs als Eingabe akzeptiert und den zugehörigen Funktionswert als Ausgabe produziert.

Verifikationsproblem:

Gegeben eine Spezifikation und ein Programm:

Beweise, dass das Programm für jedes Argument der Spezifikation als Eingabe den zugehörigen Funktionswert der Ausgabe berechnet.

- Zusatzaufgaben:**
1. Was wird bei anderen Eingaben als den zulässigen Argumenten berechnet ?
 2. Welche Bedingungen muss die Eingabe erfüllen, um bestimmte Ausgaben auszuschließen ?

Programmverifikation

Etwas allgemeiner:

Gegeben eine Spezifikation:

Relation, die jedem erlaubten Startzustand einen oder mehrere zulässige Endzustände zuordnet.

Gegeben ein Programm:

Beweise, dass das Programm für jeden Startzustand einen in der Spezifikation zugelassenen Endzustand erreicht **und danach stoppt**.

- Zusatzaufgaben:**
1. Welche Zustände werden bei anderen als den geforderten Startzuständen erreicht ?
 2. Welche Bedingungen müssen die Startzustände erfüllen, um bestimmte Endzustände auszuschließen ?

Anmerkung: Die gleichen Aufgaben können natürlich auch für beliebige Zwischenzustände untersucht werden.

Programmverifikation

Formalismus zum Lösen der Verifikationsaufgaben: Hoare-Tripel

$$\{ \varphi \} \quad P \quad \{ \psi \}$$

Vorbedingung Programm Nachbedingung

Da ein Programm eine Sequenz von Anweisungen ist, kann man dieses Vorgehen auf die einzelnen Anweisungen reduzieren:

$$\{ \varphi \} \quad S \quad \{ \psi \}$$

Vorbedingung Anweisung Nachbedingung

Fragestellungen:

1. Gegeben φ , finde stärkste Nachbedingung ψ
2. Gegeben ψ , finde schwächste Vorbedingung φ

Programmverifikation

Beispiel für das Arbeiten mit Hoare-Tripeln:

$\{ v \} \quad \varphi$

$z := x \cdot y ; \quad S$

$\{ z \geq 0 \} \quad \psi$

$w := \text{sqrt} (z)$

$\varphi_1 \Leftrightarrow (x > 0) \wedge (y > 0)$ ist eine Vorbedingung für ψ : $\{\varphi_1\} S \{\psi\}$

$\varphi_2 \Leftrightarrow (x < 0) \wedge (y < 0)$ ist auch eine Vorbedingung für ψ : $\{\varphi_2\} S \{\psi\}$

Welche ist die schwächste Vorbedingung V ?

$V \Leftrightarrow ((x > 0) \wedge (y > 0)) \vee ((x < 0) \wedge (y < 0)) \vee (x = 0) \vee (y = 0)$

ist die schwächste Vorbedingung für ψ :

$\{V\} S \{\psi\}$ Außerdem gilt: $\{\varphi\} S \{\psi\} \Rightarrow (\varphi \rightarrow V)$

Programmverifikation

Semantik von Hoare-Tripeln:

$\{ \varphi \}$ **S** $\{ \psi \}$ bedeutet:

Vorbedingung Anweisung Nachbedingung

Bei Vorliegen von φ gilt **nach Beendigung** der Anweisung S die Bedingung ψ .

Merke: φ_1 ist schwächer als φ_2 bedeutet: $\varphi_2 \rightarrow \varphi_1$ (φ_2 ist stärker als φ_1)

Also gilt: \top (w) ist die schwächste aller Bedingungen und \perp (f) die stärkste.

Folgerung: Die schwächste Vorbedingung für \top ist die Bedingung, die garantiert, dass S zu einem Ende kommt.

Forderung (Axiom): Die schwächste Vorbedingung für \perp ist \perp .

(nach Dijkstra: Gesetz des ausgeschlossenen Wunders)

Programmverifikation

Logischer Zusammenhang von Vorbedingungen:

$$(\{\varphi_1\} S \{\psi\}) \wedge (\varphi_2 \rightarrow \varphi_1) \quad \Rightarrow \quad \{\varphi_2\} S \{\psi\}$$

Die Vertauschung gilt nicht:

~~$$(\{\varphi_1\} S \{\psi\}) \wedge (\varphi_1 \rightarrow \varphi_2) \quad \Rightarrow \quad \{\varphi_2\} S \{\psi\}$$~~

Logischer Zusammenhang von Nachbedingungen:

$$(\{\varphi\} S \{\psi_1\}) \wedge (\psi_1 \rightarrow \psi_2) \quad \Rightarrow \quad \{\varphi\} S \{\psi_2\}$$

Die Vertauschung gilt nicht:

~~$$(\{\varphi\} S \{\psi_1\}) \wedge (\psi_2 \rightarrow \psi_1) \quad \Rightarrow \quad \{\varphi\} S \{\psi_2\}$$~~

Verifikation von Zuweisungen

Definition einer Zuweisung:

$x := \text{Ausdruck}$

Hierbei ist x ein beliebiger Variablenname und **Ausdruck** eine beliebige Funktion, die unter Umständen von Variablen abhängt. Die Variablen in **Ausdruck** müssen zum Zeitpunkt der Anweisung Werte haben.

Funktionsweise einer Zuweisung:

**{ Prädikate für die Werte von x_1, \dots, x_k
und eventuell weitere Prädikate }**

$x := \text{Ausdruck} (x_1, \dots, x_k)$

**{ Neues Prädikat für den Wert von x und
eventuell weitere Prädikate }**

Zunächst wird **Ausdruck** (x_1, \dots, x_k) ausgewertet.

Der sich ergebende Funktionswert wird danach in die Variable x geschrieben.

Verifikation von Zuweisungen

Beispiel für die Verifikation einer Zuweisung:

$\{ (x > 0) \wedge (y > 0) \}$ φ

$z := x - \text{sqrt}(y);$ S

$\{ N \}$ ψ

Welche ist die stärkste Nachbedingung N ?

Welche ist die schwächste Vorbedingung für $\psi \Leftrightarrow (z > 0)$?

Verifikation von Zuweisungen

Die Zuweisungsvariable darf auch im zu berechnenden Ausdruck vorkommen:

$\{ (x > 0) \}$ φ

$x := x - \text{sqrt}(x);$ S

$\{ N \}$ ψ

Welche ist die stärkste Nachbedingung N ?

Welche ist die schwächste Vorbedingung für $\psi \Leftrightarrow (x > 0)$?

Verifikation von Zuweisungen

Der allgemeine Fall einer Zuweisung:

$\{ V(x) \}$ φ

$x := f(x);$ S

$\{ N(x) \}$ ψ

Verfahren zur Berechnung der schwächsten Vorbedingung zu gegebener Nachbedingung:

- Ersetze in $N(x)$ jedes x durch $f(x)$ \rightarrow Das Ergebnis ist die schwächste Vorbedingung.

Verfahren zur Berechnung der stärksten Nachbedingung zu gegebener Vorbedingung:

- Berechne den Wertebereich von $f(x)$ für $V(x)$ \rightarrow Das Ergebnis ist eine Nachbedingung, **aber nicht unbedingt die stärkste !**

Verifikation von Zuweisungen: Vorbedingung

$$\{V(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z})\} \quad \varphi$$

$$\mathbf{z} := \mathbf{A}(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z}); \quad \mathbf{S}$$

$$\{N_1(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge N_2(\mathbf{z})\} \quad \psi$$

Finde die **schwächste Vorbedingung** φ zu gegebenem ψ :

$$V(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z}) \Leftrightarrow N_1(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge N_2(\mathbf{A}(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z}))$$

Beispiel:

$$\begin{array}{ll} \{v\} & \varphi \\ \mathbf{x} := \mathbf{x} - \text{sqrt}(\mathbf{x}); & \mathbf{S} \\ \{ \mathbf{x} > 0 \} & \psi \end{array}$$

Schwächste Vorbedingung:

$$V(\mathbf{x}) \Leftrightarrow (\mathbf{x} - \text{sqrt}(\mathbf{x}) > 0)$$

Verifikation von Zuweisungen: Nachbedingung

Zuweisung **ohne** Verwendung der Zuweisungsvariable im Ausdruck:

$$\{V(\mathbf{x}_1, \dots, \mathbf{x}_k)\} \quad \varphi$$

$$\mathbf{z} := A(\mathbf{x}_1, \dots, \mathbf{x}_k); \quad S$$

$$\{N(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z})\} \quad \psi$$

Finde die **stärkste Nachbedingung** ψ zu gegebenem φ :

$$N(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z}) \Leftrightarrow V(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge (\mathbf{z} = A(\mathbf{x}_1, \dots, \mathbf{x}_k))$$

Beispiel:

$$\begin{array}{ll} \{ (x > 0) \wedge (y > 0) \} & \varphi \\ \mathbf{z} := \mathbf{x} - \text{sqrt}(\mathbf{y}); & S \\ \{ N \} & \psi \end{array}$$

Stärkste Nachbedingung:

$$N(\mathbf{x}, \mathbf{y}, \mathbf{z}) \Leftrightarrow (\mathbf{x} > 0) \wedge (\mathbf{y} > 0) \wedge (\mathbf{z} = \mathbf{x} - \text{sqrt}(\mathbf{y}))$$

Verifikation von Zuweisungen: Nachbedingung

Zuweisung **mit** Verwendung der Zuweisungsvariable im Ausdruck:

$$\{V_1(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge V_2(\mathbf{x})\} \quad \varphi$$

$$\mathbf{x} := A(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{x}); \quad S$$

$$\{N(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{x})\} \quad \psi$$

Finde die stärkste Nachbedingung ψ zu gegebenem φ :

$$N(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{x}) \Leftrightarrow$$

$$V_1(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge$$

$$(\mathbf{x} \in \text{Wertebereich von } A(\mathbf{x}_1, \dots, \mathbf{x}_k, \mathbf{z}))$$

$$\text{wenn gilt: } (V_1(\mathbf{x}_1, \dots, \mathbf{x}_k) \wedge V_2(\mathbf{z}))$$

$$\text{Beispiel: } \{ (\mathbf{x} > 0) \} \quad \varphi$$

$$\mathbf{x} := \mathbf{x} - \text{sqrt}(\mathbf{x}); \quad S$$

$$\{ N \} \quad \psi$$

Stärkste Nachbedingung:

$$N(\mathbf{x}) \Leftrightarrow \mathbf{x} \in \text{Wertebereich von } f(\mathbf{z}) = \mathbf{z} - \text{sqrt}(\mathbf{z}) \text{ für } (\mathbf{z} > 0)$$

Verifikation von Verbundanweisungen

Definition einer Verbundanweisung:

```
begin
    Anweisung 1;
    Anweisung 2;
    ...
    Anweisung n
end
```

Hierbei dürfen die Anweisungen 1 bis n beliebige Anweisungen sein: von einfachen Zuweisungen bis hin zu ineinandergeschachtelten Kontrollstrukturen.

Funktionsweise einer Verbundanweisung:

Die Anweisungen werden der Reihe nach **hintereinander** ausgeführt.
Eine **parallele** Ausführung findet **nicht** statt.

Verifikation von Verbundanweisungen

Verifikationstechnik:

```
{Vorbedingung}
begin
    Anweisung 1;
    {Zwischenbedingung 1}
    Anweisung 2;
    {Zwischenbedingung 2}
    ...
    {Zwischenbedingung n-1}
    Anweisung n
end
{Nachbedingung}
```

Achtung:

Geübte Verifizierer schreiben nicht jede Zwischenbedingung auf. Sie müssen aber alle Zwischenbedingungen im Kopf durcharbeiten, da es keine parallele Abarbeitung gibt !

Verifikation von Verbundanweisungen

Beispiel für die Verifikation einer Verbundanweisung:

Spezifikation: 2 mit Werten belegte Variablen sollen ihre Werte tauschen.

Programm:

```
{ (x = Wert1) ∧ (y = Wert2) }  φ
begin
    x := y ;
    { (x = Wert2) ∧ (y = Wert2) }
    y := x ;
    { (x = Wert2) ∧ (y = Wert2) }
end
{ (x = Wert2) ∧ (y = Wert1) }  ψ
```

Unter welchen Bedingungen ist das Programm korrekt ?

Antwort: nur wenn Wert1 = Wert2

Verifikation von Verbundanweisungen

Wie tauscht man verschiedene Werte ?

Lösung: $\{ (x = \text{Wert1}) \wedge (y = \text{Wert2}) \} \quad \varphi$
begin
 $z := x ;$
 $\{ (x = \text{Wert1}) \wedge (y = \text{Wert2}) \wedge (z = \text{Wert1}) \}$
 $x := y ;$
 $\{ (x = \text{Wert2}) \wedge (y = \text{Wert2}) \wedge (z = \text{Wert1}) \}$
 $y := z ;$
 $\{ (x = \text{Wert2}) \wedge (y = \text{Wert1}) \wedge (z = \text{Wert1}) \}$
end
 $\{ (x = \text{Wert2}) \wedge (y = \text{Wert1}) \} \quad \psi$

Keine Einschränkung der Vor- oder Nachbedingung !

Damit ist bewiesen, dass das Programm die Spezifikation erfüllt.