Grundlagen der Programmierung

Vorlesung 7

Sebastian Iwanowski
FH Wedel

Grundlagen der Programmierung

1. Einführung
   - Grundlegende Eigenschaften von Algorithmen und Programmen
2. Logik
   - Aussagenlogik
   - Prädikatenlogik
3. Programmentwicklung und –verifikation
   - Grundlagen der Programmverifikation
   - Zuweisungen und Verbundanweisungen
   - Verzweigungen
   - ➔ Schleifen
   - Modularisierung
   - Rekursion
4. Entwurf und Analyse von Algorithmen
   - Klassifikation von Algorithmen
   - Programmierung von Algorithmen
   - Bewertung von Algorithmen

Verifikation und Konstruktion von Schleifen

Definition einer Schleife:

```
while Schleifenbedingung do
Rumpfanweisung
```

`Schleifenbedingung` muss eine logische Funktion sein, die nur von Variablen abhängen darf, die mit Werten belegt sind.

Funktionsweise:

1) Zunächst wird **`Schleifenbedingung`** ausgewertet.

2) Wenn **`Schleifenbedingung`** falsch ist, wird die Schleife sofort beendet.
   Wenn **`Schleifenbedingung`** wahr ist, wird **`Rumpfanweisung`** ausgeführt.
   Dann wird bei Schritt 1) fortgefahren.

Verifikation von Schleifen

Verifikationstechnik:

W

```
{Vorbedingung}                    φ
while Schleifenbedingung do       β
      {Eintrittsbedingung}        φ_i
      Rumpfanweisung              S
      {Austrittsbedingung}        ψ_i
{Nachbedingung}                   ψ
```

Definition:

Es sei φ_i die Eintrittsbedingung vor der i-ten Ausführung der Rumpfanweisung und ψ_i die Austrittsbedingung nach der i-ten Ausführung der Rumpfanweisung
Die Schleife werde nach k Ausführungen beendet.

Dann gilt:

1) $\varphi_1 \Leftrightarrow \varphi \wedge \beta$ $\quad \{\varphi_1\}$ **Rumpfanweisung** $\{\psi_1\}$

2) $\varphi_2 \Leftrightarrow \psi_1 \wedge \beta$ $\quad \{\varphi_2\}$ **Rumpfanweisung** $\{\psi_2\}$

.

i) $\varphi_i \Leftrightarrow \psi_{i-1} \wedge \beta$ $\quad \{\varphi_i\}$ **Rumpfanweisung** $\{\psi_i\}$

.

k) $\varphi_k \Leftrightarrow \psi_{k-1} \wedge \beta$ $\quad \{\varphi_k\}$ **Rumpfanweisung** $\{\psi_k\}$

k+1) $\psi \Leftrightarrow \psi_k \wedge \neg\beta$ $\quad \{\varphi_k\}$ **Rumpfanweisung** $\{\psi_k\}$

Verifikation von Schleifen

Verifikationstechnik:

W

```
{Vorbedingung}                    φ
while Schleifenbedingung do       β
      {Eintrittsbedingung}        φ_i
      Rumpfanweisung              S
      {Austrittsbedingung}        ψ_i
{Nachbedingung}                   ψ
```

Gegeben ψ, berechne φ: Wie findet man die schwächste Vorbedingung P für φ ?

Beobachtung:

0) Sei P_0 die schwächste Vorbedingung, falls die Schleife gar nicht durchlaufen wird.

1) Sei P_1 die schwächste Vorbedingung, falls die Schleife genau einmal durchlaufen wird.

i) Sei P_i die schwächste Vorbedingung, falls die Schleife genau i-mal durchlaufen wird.

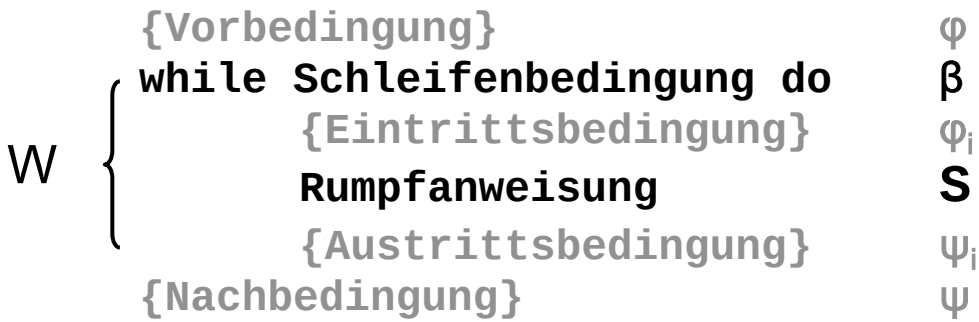
Lösung: Dann gilt: $P = P_0 \vee P_1 \vee \ldots \vee P_i \vee \ldots$

Problem: Im allgemeinen Fall könnten sich die P_i's alle unterscheiden !

Damit kann keine allgemeine Lösungstechnik angegeben werden !

Verifikation von Schleifen

Verifikationstechnik:



Einfachere Aufgabe:

Gegeben φ und ψ:

Beweise, dass gilt: **{φ} W {ψ} !**

Verifikation von Schleifen

Zur Erinnerung:

$\varphi_i \Leftrightarrow \psi_{i-1} \wedge \beta$ **d.h.:** $\{\psi_{i-1} \wedge \beta\}$ **Rumpfanweisung** $\{\psi_i\}$

Beweiselemente: Invariantenbedingung und Variantenzahl

Verwendung einer Invariantenbedingung I zum Beweis der Gültigkeit des Zusammenhangs zwischen Vor- und Nachbedingung

Wir betrachten den Fall, dass die Rumpfanweisung mindestens einmal ausgeführt wird.
Dann soll die Invariantenbedingung I Folgendes erfüllen:

1) $\{I \wedge \beta\}\ S\ \{I\}$ *Innerhalb der Schleife gilt immer die Invariante.*

2) $I \wedge \neg\beta \Rightarrow \psi$ *In der Nachbedingung gilt immer die Invariante und niemals die Schleifenbedingung.*

3) $\varphi \Rightarrow I \wedge \beta$ *Die Vorbedingung erfüllt immer die Invariante und die Schleifenbedingung.*

Fazit: *Die Invariante gilt **immer** (vor, während und nach der Schleife)*

Problem: *Damit ist **noch nicht** gewährleistet, dass die Schleife terminiert.*

Verifikation von Schleifen

Beweiselemente: Invariantenbedingung und Variantenzahl

Verwendung einer Variantenzahl $z \in \mathbb{Z}$ zur Gewährleistung der Terminierung der Schleife

1)	$z = z_0$	*z hat bei Schleifeneintritt einen definierten Wert*
2)	$\{I \wedge \beta \wedge (z=z_0)\}\ S\ \{z<z_0\}$	*z hat nach Ausführung der Rumpfanweisung einen kleineren Wert als vorher.*
3)	$I \wedge (z \leq 0) \Rightarrow \neg\beta$	*Der Wert von z sorgt dafür, dass irgendwann einmal die Schleifenbedingung verletzt wird.*

Lösung: ***Damit ist gewährleistet, dass die Schleife terminiert.***

Verifikation von Schleifen

Zusammenfassung des Verfahrens mit Invariantenbedingung und Variantenzahl:

1) Finde zu gegebenem ψ eine Invariante I mit: $I \wedge \neg\beta \Rightarrow \psi$
2) Falls φ ebenfalls gegeben ist, sorge dafür, dass $\varphi \Rightarrow I$
3) Finde eine Variantenzahl $z \in \mathbb{Z}$, sodass Folgendes erfüllt ist:

a) $\{I \wedge \beta\}$	S	$\{I\}$	(Invariantenbedingung)
b) $\{I \wedge \beta \wedge (z=z_0)\}$	S	$\{z<z_0\}$	(Fortschrittsbedingung)
c) $I \wedge (z \leq 0)$	$\Rightarrow$	$\neg\beta$	(Terminierungsbedingung)

$\{\varphi\}$
$\{I \wedge \beta\}$
while β do
begin
  $\{I \wedge \beta \wedge (z=z_0)\}$
  S
  $\{I \wedge (z<z_0)\}$
end
$\{I \wedge \neg\beta\}$
$\{\psi\}$

i. *Die Variantenzahl muss keine Programmvariable sein.*

ii. *Die Variantenzahl muss nicht genau die hier geforderten Eigenschaften erfüllen.*
<u>Wesentlich ist:</u> Sie muss nach endlich vielen Schritten für eine Terminierungsbedingung sorgen.

Konstruktion von Schleifen

Das eben geschilderte Verifikationsverfahren kann auch zur Programmentwicklung eingesetzt werden:

1) Finde zu gegebenem ψ eine Zerlegung in I und ¬β: $I \wedge \neg\beta \Rightarrow \psi$
2) Sorge dafür, dass I schon vor der Schleife gilt: Finde S_0 mit: $\{\varphi\}\ S_0\ \{I\}$
3) Finde eine Rumpfanweisung S und eine Variantenzahl $z \in \mathbb{Z}$, sodass Folgendes erfüllt ist:

a) $\{I \wedge \beta\}$	S	$\{I\}$	(Invariantenbedingung)
b) $\{I \wedge \beta \wedge (z=z_0)\}$	S	$\{z<z_0\}$	(Fortschrittsbedingung)
c) $I \wedge (z \leq 0)$	$\Rightarrow$	¬β	(Terminierungsbedingung)

```
{φ}
S0
{I}
while β do
begin
        {I ∧ β ∧ (z=z0)} }
        S
        {I ∧ (z<z0) }
end
{I ∧ ¬β}
{ψ}
```

i. *Die Variantenzahl muss keine Programmvariable sein.*

ii. *Die Variantenzahl muss nicht genau die hier geforderten Eigenschaften erfüllen.*
<u>Wesentlich ist:</u> Sie muss nach endlich vielen Schritten für eine Terminierungsbedingung sorgen.

Beim nächsten Mal:

Programmentwicklung und –verifikation:
Induktionsbeweise für Schleifen, Modularisierung