

Grundbegriffe der Informatik

Tutorium 33

Lukas Bach, lukas.bach@student.kit.edu | 22.12.2016



Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Es existiert eine **endliche** Beschreibung

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Es existiert eine **endliche** Beschreibung
- Es wird zu einer beliebig großen, aber **endlichen** Eingabe eine **endliche** Ausgabe berechnet

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Es existiert eine **endliche** Beschreibung
- Es wird zu einer beliebig großen, aber **endlichen** Eingabe eine **endliche** Ausgabe berechnet
- Es finden **endlich** viele Schritte statt (der Algorithmus terminiert)

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Es existiert eine **endliche** Beschreibung
- Es wird zu einer beliebig großen, aber **endlichen** Eingabe eine **endliche** Ausgabe berechnet
- Es finden **endlich** viele Schritte statt (der Algorithmus terminiert)
- Deterministisch (bei mehrmaliger Ausführung kommt immer das selbe raus)

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

■ Zuweisungssymbol \leftarrow

- Zuweisungssymbol \leftarrow
- Schlüsselwörter für Verzweigungen **if, then, else, fi**

- Zuweisungssymbol \leftarrow
- Schlüsselwörter für Verzweigungen **if, then, else, fi**
- Schlüsselwörter für Schleifen **while, do, od, for, to**

- Zuweisungssymbol \leftarrow
- Schlüsselwörter für Verzweigungen **if**, **then**, **else**, **fi**
- Schlüsselwörter für Schleifen **while**, **do**, **od**, **for**, **to**
- Symbole für Konstanten, Funktionen und Relationen

Eine **if**-Verzweigung

```
1 if  $x < y$  then  
2    $s \leftarrow x$   
3 else  
4    $s \leftarrow y$   
5 fi
```

Eine **if**-Verzweigung

```
1 if  $x < y$  then  
2    $s \leftarrow x$   
3 else  
4    $s \leftarrow y$   
5 fi
```

Eine **while**-Schleife

```
1 while  $x > 0$  do  
2    $x \leftarrow x \text{ div } 2$   
3    $s \leftarrow s + x$   
4 od
```

Eine **if**-Verzweigung

```
1 if  $x < y$  then  
2    $s \leftarrow x$   
3 else  
4    $s \leftarrow y$   
5 fi
```

Eine **while**-Schleife

```
1 while  $x > 0$  do  
2    $x \leftarrow x \text{ div } 2$   
3    $s \leftarrow s + x$   
4 od
```

Eine **for**-Schleife

```
1 for  $i \leftarrow 1$  to  $n$  do  
2    $s \leftarrow s + i$   
3 od
```

Was kann man mit Algorithmen machen?

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Was kann man mit Algorithmen machen?

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen

Was kann man mit Algorithmen machen?

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen, im Modul [Algorithmen I](#)

Was kann man mit Algorithmen machen?

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen, im Modul [Algorithmen I](#)
- Laufzeitanalyse von Algorithmen

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen, im Modul [Algorithmen I](#)
- Laufzeitanalyse von Algorithmen, später.

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen, im Modul [Algorithmen I](#)
- Laufzeitanalyse von Algorithmen, später.
- Korrektheitsbeweise

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Komplexe Algorithmen mit Pseudocode definieren zu Sortierung, Graphen, Datenstrukturen, im Modul [Algorithmen I](#)
- Laufzeitanalyse von Algorithmen, später.
- Korrektheitsbeweise, jetzt.

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Wie findet man heraus, ob ein Algorithmus korrekt funktioniert?

Algorithmen

Pseudocode

Das Hoare-Kalkül

Wie findet man heraus, ob ein Algorithmus korrekt funktioniert?

- Durch den Beweis von Zusicherungen, die an bestimmten Stellen des Algorithmus gelten.

Algorithmen

Pseudocode

Das Hoare-Kalkül

Wie findet man heraus, ob ein Algorithmus korrekt funktioniert?

- Durch den Beweis von Zusicherungen, die an bestimmten Stellen des Algorithmus gelten.

Was sind Zusicherungen?

Algorithmen

Pseudocode

Das Hoare-Kalkül

Wie findet man heraus, ob ein Algorithmus korrekt funktioniert?

- Durch den Beweis von Zusicherungen, die an bestimmten Stellen des Algorithmus gelten.

Was sind Zusicherungen?

- prädikatenlogische Formeln, die Aussagen über (Zusammenhänge zwischen) Variablen machen

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel.

Algorithmen

Pseudocode

Das Hoare-Kalkül

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode
- P und Q sind Zusicherungen

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode
 - P und Q sind Zusicherungen
-
- P nennt man Vorbedingung, Q Nachbedingung

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode
 - P und Q sind Zusicherungen
-
- P nennt man Vorbedingung, Q Nachbedingung
 - Prädikatenlogische Formeln

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode
- P und Q sind Zusicherungen
- P nennt man Vorbedingung, Q Nachbedingung
- Prädikatenlogische Formeln
- Beispiel (Vorausblick): $\{x \doteq 1\}x \leftarrow x + 1\{x \doteq 2\}$

Definition

$\{P\}S\{Q\}$ heißt Hoare-Tripel. Dabei gilt:

- S ist ein Programmstück im Pseudocode
- P und Q sind Zusicherungen
- P nennt man Vorbedingung, Q Nachbedingung
- Prädikatenlogische Formeln
- Beispiel (Vorausblick): $\{x \doteq 1\}x \leftarrow x + 1\{x \doteq 2\}$
- Meistens in jeder Zeile nur eine Zeile Code oder ein Zusicherungsblock

Gültigkeit von Hoare-Tripeln

$\{P\}S\{Q\}$ ist gültig, wenn für jede gültige Interpretation (D, I) und Variablenbelegung β gilt:

Gültigkeit von Hoare-Tripeln

$\{P\}S\{Q\}$ ist gültig, wenn für jede gültige Interpretation (D, I) und Variablenbelegung β gilt:

Aus

Gültigkeit von Hoare-Tripeln

$\{P\}S\{Q\}$ ist gültig, wenn für jede gültige Interpretation (D, I) und Variablenbelegung β gilt:

Aus

- $val_{D,I,\beta}(P) = w$

Gültigkeit von Hoare-Tripeln

$\{P\}S\{Q\}$ ist gültig, wenn für jede gültige Interpretation (D, I) und Variablenbelegung β gilt:

Aus

- $val_{D,I,\beta}(P) = w$
- β' ist Variablenbelegung nach Ausführung von S

Gültigkeit von Hoare-Tripeln

$\{P\}S\{Q\}$ ist gültig, wenn für jede gültige Interpretation (D, I) und Variablenbelegung β gilt:

Aus

- $val_{D,I,\beta}(P) = w$
- β' ist Variablenbelegung nach Ausführung von S

folgt $val_{D,I,\beta'}(Q) = w$

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Axiom HT-A

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und
- $\sigma_{\{x/E\}}$ kollisionsfrei für Q

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und
- $\sigma_{\{x/E\}}$ kollisionsfrei für Q

Dann ist $\sigma_{\{x/E\}}(Q)x \leftarrow E\{Q\}$ ein gültiges Hoare-Tripel

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und
- $\sigma_{\{x/E\}}$ kollisionsfrei für Q

Dann ist $\sigma_{\{x/E\}}(Q)x \leftarrow E\{Q\}$ ein gültiges Hoare-Tripel

Bemerkung

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und
- $\sigma_{\{x/E\}}$ kollisionsfrei für Q

Dann ist $\sigma_{\{x/E\}}(Q)x \leftarrow E\{Q\}$ ein gültiges Hoare-Tripel

Bemerkung

- $\sigma_{\{x/E\}}$ ist die Substitution von x mit E

Axiom HT-A

- Sei $x \leftarrow E$ eine Zuweisung
- Q eine Nachbedingung von $x \leftarrow E$ und
- $\sigma_{\{x/E\}}$ kollisionsfrei für Q

Dann ist $\sigma_{\{x/E\}}(Q)x \leftarrow E\{Q\}$ ein gültiges Hoare-Tripel

Bemerkung

- $\sigma_{\{x/E\}}$ ist die Substitution von x mit E
- Bei Anwendung der Regel rückwärts vorgehen

Grundbegriffe der Informatik

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Beispiel

Betrachte die Zuweisung

$x \leftarrow x + 1$

und die Nachbedingung

$\{x=1\}$

Grundbegriffe der Informatik

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Beispiel

Betrachte die Zuweisung

$x \leftarrow x + 1$

und die Nachbedingung

$\{x \doteq 1\}$

Nach HT-A gilt

Beispiel

Betrachte die Zuweisung

$x \leftarrow x + 1$

und die Nachbedingung

$\{x \doteq 1\}$

Nach HT-A gilt

$\{x + 1 \doteq 1\} \ x \leftarrow x + 1 \ \{x \doteq 1\}$ ist ein gültiges Hoare-Tripel.

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Verstärkung der Vorbedingung
- Abschwächung der Nachbedingung

- Verstärkung der Vorbedingung
- Abschwächung der Nachbedingung

HT-E

Wenn $\{P\}S\{Q\}$ ein gültiges Hoare-Tripel ist und $P' \vdash P$ und $Q \vdash Q'$ gelten, dann folgt:

- Verstärkung der Vorbedingung
- Abschwächung der Nachbedingung

HT-E

Wenn $\{P\}S\{Q\}$ ein gültiges Hoare-Tripel ist und $P' \vdash P$ und $Q \vdash Q'$ gelten, dann folgt:
 $\{P'\}S\{Q'\}$ ist ein gültiges Hoare-Tripel.

- Verstärkung der Vorbedingung
- Abschwächung der Nachbedingung

HT-E

Wenn $\{P\}S\{Q\}$ ein gültiges Hoare-Tripel ist und $P' \vdash P$ und $Q \vdash Q'$ gelten, dann folgt:
 $\{P'\}S\{Q'\}$ ist ein gültiges Hoare-Tripel.

Bemerkung

$B \vdash A :\Leftrightarrow$ Aussage A ist syntaktisch aus Aussage B ableitbar

Beispiel

Angenommen es sei $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ ein gültiges Hoare-Tripel.

Es gilt $\{(y > 4)\} \vdash \{(y > 3)\}$ und $\{(x > 1)\} \vdash \{(x > 0)\}$.

Also folgt nach HT-E:

Beispiel

Angenommen es sei $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ ein gültiges Hoare-Tripel.

Es gilt $\{(y > 4)\} \vdash \{(y > 3)\}$ und $\{(x > 1)\} \vdash \{(x > 0)\}$.

Also folgt nach HT-E:

$\{y > 4\} x \leftarrow y - 1 \{x > 0\}$ ist ein gültiges Hoare-Tripel.

Beispiel

Angenommen es sei $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ ein gültiges Hoare-Tripel.

Es gilt $\{(y > 4)\} \vdash \{(y > 3)\}$ und $\{(x > 1)\} \vdash \{(x > 0)\}$.

Also folgt nach HT-E:

$\{y > 4\} x \leftarrow y - 1 \{x > 0\}$ ist ein gültiges Hoare-Tripel.

Bemerkung

Es müssen sich nicht unbedingt beide Bedingungen ändern!

Aus $\{(y > 3)\} \vdash \{(y > 3)\}$ und $\{(x > 1)\} \vdash \{(x > 0)\}$

folgt nach HT-E auch

$\{y > 3\} x \leftarrow y - 1 \{x > 0\}$ ist ein gültiges Hoare-Tripel.

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Hintereinanderausführung von durch Hoare-Triple bewiesene Code
Segmente sind selbst gültig.

Hintereinanderausführung von durch Hoare-Triple bewiesene Code
Segmente sind selbst gültig.

HT-S

Wenn $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültige Hoare-Tripel sind

Hintereinanderausführung von durch Hoare-Tripel bewiesene Code
Segmente sind selbst gültig.

HT-S

Wenn $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültige Hoare-Tripel sind, dann
folgt:

Hintereinanderausführung von durch Hoare-Tripel bewiesene Code
Segmente sind selbst gültig.

HT-S

Wenn $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültige Hoare-Tripel sind, dann
folgt: $\{P\}S_1; S_2\{R\}$ ist ein gültiges Hoare-Tripel.

Hintereinanderausführung von durch Hoare-Tripel bewiesene Code
Segmente sind selbst gültig.

HT-S

Wenn $\{P\}S_1\{Q\}$ und $\{Q\}S_2\{R\}$ gültige Hoare-Tripel sind, dann
folgt: $\{P\}S_1; S_2\{R\}$ ist ein gültiges Hoare-Tripel.

Bemerkung

";" trennt hier zwei Programmstücke

Beispiel

Angenommen es seien $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ und $\{x > 1\} z \leftarrow x - 1 \{z > -1\}$ gültige Hoare-Tripel.

Beispiel

Angenommen es seien $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ und
 $\{x > 1\} z \leftarrow x - 1 \{z > -1\}$ gültige Hoare-Tripel.

Dann folgt nach HT-S:

Beispiel

Angenommen es seien $\{y > 3\} x \leftarrow y - 1 \{x > 1\}$ und
 $\{x > 1\} z \leftarrow x - 1 \{z > -1\}$ gültige Hoare-Tripel.

Dann folgt nach HT-S:

$\{y > 3\} x \leftarrow y - 1; z \leftarrow x - 1 \{z > -1\}$ ein gültiges Hoare-Tripel.

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

HT-I

Wenn $\{P \wedge B\}S_1\{Q\}$ und $\{P \wedge \neg B\}S_2\{Q\}$ gültige Hoare-Tripel sind

HT-I

Wenn $\{P \wedge B\}S_1\{Q\}$ und $\{P \wedge \neg B\}S_2\{Q\}$ gültige Hoare-Tripel sind, dann folgt:

```
{P}  
  if B then S1  
  else S2  
  fi  
{Q}
```

ist ein gültiges Hoare-Tripel.

$\{ x = a \wedge y = b \}$

if $x > y$

then

$\{ \dots \}$

$z \leftarrow x$

$\{ \dots \}$

else

$\{ \dots \}$

$z \leftarrow y$

$\{ \dots \}$

fi

$\{ z = \min(a, b) \}$

$\{ x = a \wedge y = b \}$

if $x > y$

then

$\{ x = a \wedge y = b \wedge \neg(x > y) \}$

$\{ x = \min(a, b) \}$

$z \leftarrow x$

$\{ z = \min(a, b) \}$

else

$\{ x = a \wedge y = b \wedge x > y \}$

$\{ y = \min(a, b) \}$

$z \leftarrow y$

$\{ z = \min(a, b) \}$

fi

$\{ z = \min(a, b) \}$

HT-W

Wenn $\{I \wedge B\}S\{I\}$ ein gültiges Hoare-Tripel ist

HT-W

Wenn $\{I \wedge B\}S\{I\}$ ein gültiges Hoare-Tripel ist, dann folgt:

HT-W

Wenn $\{I \wedge B\}S\{I\}$ ein gültiges Hoare-Tripel ist, dann folgt:

$\{I\}$

while B **do** S

od

$\{I \wedge \neg B\}$

ist ein gültiges Hoare-Tripel.

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül

- Eine spezielle Zusicherung

- Eine spezielle Zusicherung
- Schleifeninvarianten müssen **vor**, **während** und **nach** jedem Schleifendurchlauf gelten

- Eine spezielle Zusicherung
- Schleifeninvarianten müssen **vor**, **während** und **nach** jedem Schleifendurchlauf gelten
- Garantiert, dass die Schleife nicht während einem beliebigen Durchlauf “kaputt” geht.

Beispiel

Algorithmen

Pseudocode

Das Hoare-Kalkül

```
{  $x = a \wedge y = b$  }
```

```
{ ... }
```

```
while  $y \neq 0$ 
```

```
do
```

```
    { ... }
```

```
     $y \leftarrow y - 1$ 
```

```
    { ... }
```

```
     $x \leftarrow x + 1$ 
```

```
    { ... }
```

```
od
```

```
{ ... }
```

```
{  $x = a + b$  }
```


Beispiel

Algorithmen

Pseudocode

Das Hoare-Kalkül

$$\{ x = a \wedge y = b \}$$

$$\{ x + y = a + b \}$$

while $y \neq 0$

do

$$\{ x + y = a + b \wedge y \neq 0 \}$$

$$\{ x + 1 + y - 1 = a + b \}$$

$$y \leftarrow y - 1$$

$$\{ x + 1 + y = a + b \}$$

$$x \leftarrow x + 1$$

$$\{ x + y = a + b \}$$

od

$$\{ x + y = a + b \wedge \neg(y \neq 0) \}$$

$$\{ x = a + b \}$$

Grundbegriffe der Informatik

Lukas Bach, lu-
kas.bach@student.kit.edu

Algorithmen

Pseudocode

Das Hoare-Kalkül



That's all Folks!