



Syntax und Semantik von Programmen 2

Modul 6 (v1.0)

Kanonikvorlesung: Foundations of Computing

Heiko Mantel

MAIS, TU Darmstadt, WS10/11

Motivation

Wie beweist man Aussagen über Programme formal?

- ☐ basierend auf der formal modellierten Syntax und Semantik

Unterschiedliche Beweistechniken

- ☐ Fallunterscheidung
- ☐ Widerspruchsbeweis
- ☐ Strukturelle Induktion
- ☐ Induktion über Herleitungen

Fokus dieses Moduls: Wie beweist man Eigenschaften?

Übersicht: Modul 6

Äquivalenz zweier Programme

- ☐ Beweis mit Fallunterscheidung

Nichtterminierung eines Programms

- ☐ Widerspruchsbeweis

Induktionsprinzipien

- ☐ Induktion auf den natürlichen Zahlen
- ☐ wohlfundierte Induktion
- ☐ Rechtfertigung der Induktion auf den natürlichen Zahlen
- ☐ strukturelle Induktion auf Aexp
- ☐ Rechtfertigung der strukturellen Induktion auf Aexp

Deterministische Auswertung von Ausdrücken

- ☐ Beweis mit struktureller Induktion

Äquivalenz zweier Programme (1)

Definition (aus Modul 5)

Zwei **Kommandos** $c1, c2 \in \text{Com}$ sind **zueinander semantisch äquivalent** wenn für alle Grundsubstitutionen η , deren Definitionsbereich alle Metavariablen in $c1$ und $c2$ einschließt, und für alle Zustände σ, σ' die folgende Bedingung gilt:

□ $\langle c1\eta, \sigma \rangle \rightarrow \sigma'$ ist herleitbar genau dann wenn $\langle c2\eta, \sigma \rangle \rightarrow \sigma'$ herleitbar ist.

Notation

Wir schreiben $c1 \sim c2$, um auszudrücken, dass $c1$ und $c2$ **zueinander semantisch äquivalent** sind.

Äquivalenz zweier Programme (2)

Theorem

$\text{while } B \text{ do } C \text{ od} \sim \text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi}$

Beweis

- Seien σ und σ' beliebige Zustände und η eine beliebige Grundsubstitution, deren Definitionsbereich B und C enthält.
- Folgende beiden Teilaussagen sind zu beweisen:
 - a) Wenn $\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma'$ herleitbar ist, dann ist $\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma'$ ebenfalls herleitbar.
 - b) Wenn $\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma'$ herleitbar ist, dann ist $\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma'$ ebenfalls herleitbar.

Äquivalenz zweier Programme (3)

Beweis (Fortsetzung)

Beweis von Aussage a)

- ☐ Angenommen $\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma'$ sei herleitbar.
- ☐ Es gibt zwei Möglichkeiten für die letzte Regel in der Herleitung:
 - i. rwhf ist die letzte Regel
 - ii. rwht ist die letzte Regel

Wir machen eine Fallunterscheidung über diese Möglichkeiten:

Äquivalenz zweier Programme (4)

Beweis von Fall ai.

- Da rwhf die letzte Regel in der Herleitung ist, muss diese folgende Form haben:

$$\text{rwhf} \quad \frac{\begin{array}{c} \vdots \mathcal{H}1 \\ \langle B\eta, \sigma \rangle \Downarrow \text{false} \end{array}}{\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma}$$

- Wir haben also eine Herleitung $\mathcal{H}1$ von $\langle B\eta, \sigma \rangle \Downarrow \text{false}$ und es gilt $\sigma' = \sigma$.
- Mit Hilfe der Herleitung $\mathcal{H}1$ können wir folgende Herleitung von $\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma'$ konstruieren:

$$\text{riff} \quad \frac{\begin{array}{c} \vdots \mathcal{H}1 \\ \langle B\eta, \sigma \rangle \Downarrow \text{false} \end{array} \quad \text{rsk} \quad \frac{}{\langle (\text{skip})\eta, \sigma \rangle \rightarrow \sigma}}{\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma}$$

Äquivalenz zweier Programme (5)

Beweis von Fall aii.

- Da `rwht` die letzte Regel in der Herleitung ist, muss diese folgende Form haben:

$$\text{rwht} \frac{\begin{array}{c} \vdots \mathcal{H}1 \\ \langle B \eta, \sigma \rangle \Downarrow \text{true} \end{array} \quad \begin{array}{c} \vdots \mathcal{H}2 \\ \langle C \eta, \sigma \rangle \rightarrow \sigma'' \end{array} \quad \begin{array}{c} \vdots \mathcal{H}3 \\ \langle (\text{while } B \text{ do } C \text{ od}) \eta, \sigma'' \rangle \rightarrow \sigma' \end{array}}{\langle (\text{while } B \text{ do } C \text{ od}) \eta, \sigma \rangle \rightarrow \sigma'}$$

- Mit Hilfe von $\mathcal{H}1$, $\mathcal{H}2$ und $\mathcal{H}3$ können wir folgende Herleitung von $\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi}) \eta, \sigma \rangle \rightarrow \sigma'$ konstruieren:

$$\text{riff} \frac{\begin{array}{c} \vdots \mathcal{H}1 \\ \langle B \eta, \sigma \rangle \Downarrow \text{true} \end{array} \quad \begin{array}{c} \vdots \mathcal{H}2 \\ \langle C \eta, \sigma \rangle \rightarrow \sigma'' \end{array} \quad \begin{array}{c} \vdots \mathcal{H}3 \\ \langle (\text{while } B \text{ do } C \text{ od}) \eta, \sigma'' \rangle \rightarrow \sigma' \end{array}}{\begin{array}{c} \text{r;} \\ \langle (C; \text{ while } B \text{ do } C \text{ od}) \eta, \sigma \rangle \rightarrow \sigma' \end{array}} \quad \frac{}{\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi}) \eta, \sigma \rangle \rightarrow \sigma'}$$

Äquivalenz zweier Programme (6)

Beweis (Fortsetzung)

Beweis von Aussage b)

- Angenommen $\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma'$ sei herleitbar.
- Es gibt zwei Möglichkeiten für die letzte Regel in der Herleitung:
 - i. riff ist die letzte Regel
 - ii. rift ist die letzte Regel

Wir machen eine Fallunterscheidung über diese Möglichkeiten:

Äquivalenz zweier Programme (7)

Beweis von Fall bi.

- Da riff die letzte Regel in der Herleitung ist, muss diese folgende Form haben:

$$\text{riff} \frac{\begin{array}{c} \vdots \mathcal{H}_1 \\ \langle B\eta, \sigma \rangle \Downarrow \text{false} \end{array} \quad \begin{array}{c} \vdots \mathcal{H}_2 \\ \langle (\text{skip})\eta, \sigma \rangle \rightarrow \sigma' \end{array}}{\langle (\text{if } B \text{ then } C; \text{ while } B \text{ do } C \text{ od else skip fi})\eta, \sigma \rangle \rightarrow \sigma'}$$

- Die Herleitung \mathcal{H}_2 kann nur folgende Form haben:

$$\text{rsk} \frac{}{\langle (\text{skip})\eta, \sigma \rangle \rightarrow \sigma}$$

- Mit Hilfe von $\sigma' = \sigma$ und \mathcal{H}_1 können wir folgende Herleitung von $\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma'$ konstruieren:

$$\text{rwhf} \frac{\begin{array}{c} \vdots \mathcal{H}_1 \\ \langle B\eta, \sigma \rangle \Downarrow \text{false} \end{array}}{\langle (\text{while } B \text{ do } C \text{ od})\eta, \sigma \rangle \rightarrow \sigma}$$

Äquivalenz zweier Programme (8)

Beweis von Fall bii.

...

Übung: Vervollständigen Sie obigen Beweis!

Übersicht: Modul 6

Äquivalenz zweier Programme

- ☐ Beweis mit Fallunterscheidung

Nichtterminierung eines Programms

- ☐ Widerspruchsbeweis

Induktionsprinzipien

- ☐ Induktion auf den natürlichen Zahlen
- ☐ wohlfundierte Induktion
- ☐ Rechtfertigung der Induktion auf den natürlichen Zahlen
- ☐ strukturelle Induktion auf Aexp
- ☐ Rechtfertigung der strukturellen Induktion auf Aexp

Deterministische Auswertung von Ausdrücken

- ☐ Beweis mit struktureller Induktion

Nichtterminierung (1)

Theorem

Es gibt keine Zustände σ und σ' , so dass

$\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$ herleitbar ist.

Die Intuition ist: das Programm `while true do skip od` terminiert nie.

Beweis (Widerspruchsbeweis)

- ❑ Angenommen es gebe σ, σ' , so dass $\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$ herleitbar ist.
- ❑ Sei \mathcal{H} eine Herleitung von $\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$, so dass es keine Herleitung von $\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$ mit weniger Regelanwendungen gibt, d.h. \mathcal{H} ist eine kleinste Herleitung.
- ❑ Die letzte Regel in \mathcal{H} kann nur die Regel `rwht` sein.
- ❑ Daher muss die Herleitung folgende Form haben:

$$\begin{array}{c}
 \text{rtrue} \quad \text{rwht} \quad \frac{\langle \text{true}, \sigma \rangle \Downarrow \text{true} \quad \frac{\langle \text{skip}, \sigma \rangle \rightarrow \sigma'' \quad \langle \text{while true do skip od}, \sigma'' \rangle \rightarrow \sigma'}{\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'} \quad \mathcal{H}1 \quad \mathcal{H}2}{\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'}
 \end{array}$$

Nichtterminierung (2)

Beweis (Fortsetzung)

- Die Herleitung $\mathcal{H}1$ kann nur folgende Form haben:

$$\text{rsk} \frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma}$$

- Das heißt, es gilt $\sigma'' = \sigma$.
- Somit hat die Herleitung \mathcal{H} folgende Form:

$$\text{rwht} \frac{\text{rtrue} \frac{}{\langle \text{true}, \sigma \rangle \Downarrow \text{true}} \quad \text{rsk} \frac{}{\langle \text{skip}, \sigma \rangle \rightarrow \sigma} \quad \text{rsk} \frac{}{\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'} \quad \mathcal{H}2}{\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'}$$

- $\mathcal{H}2$ ist auch eine Herleitung von $\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$, hat aber weniger Regelanwendungen als \mathcal{H} .
- \mathcal{H} ist also keine kleinste Herleitung von $\langle \text{while true do skip od}, \sigma \rangle \rightarrow \sigma'$.

Ein Widerspruch!

Übersicht: Modul 6

Äquivalenz zweier Programme

- ☐ Beweis mit Fallunterscheidung

Nichtterminierung eines Programms

- ☐ Widerspruchsbeweis

Induktionsprinzipien

- ☐ Induktion auf den natürlichen Zahlen
- ☐ wohlfundierte Induktion
- ☐ Rechtfertigung der Induktion auf den natürlichen Zahlen
- ☐ strukturelle Induktion auf Aexp
- ☐ Rechtfertigung der strukturellen Induktion auf Aexp

Deterministische Auswertung von Ausdrücken

- ☐ Beweis mit struktureller Induktion

Induktion

Beweisprinzip der Induktion auf den natürlichen Zahlen

Sei $P \subseteq \mathbb{N}$ eine einstellige Relation über den natürlichen Zahlen.

Wenn folgende Bedingungen gelten:

☐ $P(0)$

☐ $\forall n \in \mathbb{N}: (P(n) \Rightarrow P(n+1))$

dann gilt auch

☐ $\forall n \in \mathbb{N}: P(n)$

Die Formel $P(n)$ wird als **Induktionsannahme** bezeichnet, und P wird als **Induktionsformel** bezeichnet.

Intuition

Wenn die „Eigenschaft P “ für 0 gilt und wenn aus „ P gilt für n “ auch „ P gilt für $n+1$ “ gefolgert werden kann, dann haben alle natürlichen Zahlen die „Eigenschaft P “.

Wohlfundierte Induktion (1)

Definition

Sei $< \subseteq D \times D$ eine binäre Relation auf einer Menge D . Eine unendliche Folge $f: \mathbb{N} \rightarrow D$ (wurde in Modul 3 eingeführt) heißt **unendlich absteigende Kette für $<$** genau dann wenn $\forall i \in \mathbb{N}: f(i+1) < f(i)$ gilt.

Definition

Eine binäre Relation $< \subseteq D \times D$ auf einer Menge D heißt **wohlfundiert** genau dann wenn es keine unendliche absteigende Kette für $<$ gibt.

Wohlfundierte Induktion (2)

Definition

Die **transitive Hülle** einer binären Relation $R \subseteq D \times D$ auf einer Menge D ist die kleinste Relation $R^* \subseteq D \times D$, so dass

- ☐ $\forall d_1, d_2 \in D: [d_1 R d_2 \Rightarrow d_1 R^* d_2]$
- ☐ $\forall d_1, d_2, d_3 \in D: [(d_1 R^* d_2 \wedge d_2 R^* d_3) \Rightarrow d_1 R^* d_3]$

Theorem

Sei $< \subseteq D \times D$ eine wohlfundierte Relation auf D . Dann gilt:

- ☐ $<$ ist irreflexiv, d.h. $\nexists d \in D: d < d$ und
- ☐ $<^*$ ist eine wohlfundierte Relation.

Beweis

....

Übung: Führen Sie obigen Beweis!

Wohlfundierte Induktion (3)

Beweisprinzip der wohlfundierten Induktion

Sei $P \subseteq D$ eine einstellige Relation auf einer Menge D und $< \subseteq D \times D$ eine wohlfundierte Relation auf D . Wenn folgende Bedingung gilt:

$$\square \forall d \in D: [(\forall d' \in D: (d' < d \Rightarrow P(d'))) \Rightarrow P(d)]$$

dann gilt auch

$$\square \forall d \in D: P(d)$$

Die Formeln $P(d')$ werden als **Induktionsannahme** bezeichnet, und P wird als **Induktionsformel** bezeichnet.

Intuition

Wenn aus „ P gilt für alle d' , die kleiner sind als d “ auch „ P gilt für d “ gefolgert werden kann, dann haben alle Elemente von D die „Eigenschaft P “.

Instanziierung des Beweisprinzips (1)

Beispiel (Rechtfertigung der Induktion auf \mathbb{N})

Die Relation $< \subseteq \mathbb{N} \times \mathbb{N}$ sei definiert durch

- $m < n$ genau dann wenn $m+1=n$

Wir erhalten folgende Spezialisierung des Beweisprinzips der wohlfundierten Induktion:

- „Wenn folgende Bedingung gilt:

- $\forall n \in \mathbb{N}: [(\forall n' \in \mathbb{N}: (n'+1=n \Rightarrow P(n'))) \Rightarrow P(n)]$

dann gilt auch

- $\forall n \in \mathbb{N}: P(n)$ “

Durch eine Fallunterscheidung über „ $n=0$ “ erhalten wir

- „Wenn folgende Bedingungen gelten:

- $(\forall n' \in \mathbb{N}: (n'+1=0 \Rightarrow P(n'))) \Rightarrow P(0)$

- $\forall n \in \mathbb{N}: [n \neq 0 \Rightarrow [(\forall n' \in \mathbb{N}: (n'+1=n \Rightarrow P(n'))) \Rightarrow P(n)]]$

dann gilt auch

- $\forall n \in \mathbb{N}: P(n)$ “

Instanziierung des Beweisprinzips (2)

Beispiel (Fortsetzung)

Also gilt auch

☐ „Wenn folgende Bedingungen gelten:

☐ $\text{true} \Rightarrow P(0)$

☐ $\forall n'' \in \mathbb{N}: [n''+1 \neq 0 \Rightarrow [(\forall n' \in \mathbb{N}: (n'+1 = n''+1 \Rightarrow P(n')))] \Rightarrow P(n''+1)]$

dann gilt auch

☐ $\forall n \in \mathbb{N}: P(n)$ “

Also gilt auch

☐ „Wenn folgende Bedingung gilt:

☐ $P(0)$

☐ $\forall n'' \in \mathbb{N}: [P(n'') \Rightarrow P(n''+1)]$

dann gilt auch

☐ $\forall n \in \mathbb{N}: P(n)$ “

Das Induktionsprinzip auf den natürlichen Zahlen ist also ein Spezialfall der wohlfundierten Induktion.

Strukturelle Induktion für Aexp (1)

Definition (aus Modul 5)

Die Menge Aexp ist durch folgende BNF definiert:

□ $a ::= n \mid X \mid a+a \mid a-a \mid a*a$

Beweisprinzip der strukturellen Induktion für Aexp

Sei $P \subseteq Aexp$ eine einstellige Relation über Aexp. Wenn folgende Bedingungen gelten:

- $\forall n \in \mathbb{N}: P(n)$
- $\forall x \in Loc: P(x)$
- $\forall a_1, a_2 \in Aexp: (P(a_1) \wedge P(a_2)) \Rightarrow P(a_1+a_2)$
- $\forall a_1, a_2 \in Aexp: (P(a_1) \wedge P(a_2)) \Rightarrow P(a_1-a_2)$
- $\forall a_1, a_2 \in Aexp: (P(a_1) \wedge P(a_2)) \Rightarrow P(a_1*a_2)$

dann gilt auch

- $\forall a \in Aexp: P(a)$

Die Formeln $P(a_1)$ und $P(a_2)$ werden als **Induktionsannahmen** bezeichnet, und P wird als **Induktionsformel** bezeichnet.

Strukturelle Induktion für Aexp (2)

Definition

Die Ausdrücke a_1 und a_2 sind die **direkten Teilausdrücke** der Ausdrücke $a_1 + a_2$, $a_1 - a_2$ und $a_1 * a_2$. Die Ausdrücke n und x haben keine direkten Teilausdrücke.

Rechtfertigung der strukturellen Induktion für Aexp

Die Relation $\prec \subseteq \text{Aexp} \times \text{Aexp}$ sei definiert durch

□ $a' \prec a$ genau dann wenn a' ein direkter Teilausdruck von a ist.

Wir erhalten folgende Spezialisierung des Beweisprinzips der wohlfundierten Induktion:

□ „Wenn folgende Bedingung gilt:

□ $\forall a \in \text{Aexp}: [(\forall a' \in \text{Aexp}: (a' \text{ ist ein direkter Teilausdruck von } a \Rightarrow P(a')) \Rightarrow P(a)]$

dann gilt auch

□ $\forall a \in \text{Aexp}: P(a)$ “

Strukturelle Induktion für Aexp (3)

Rechtfertigung (Fortsetzung)

Durch eine Fallunterscheidung über die Struktur der Ausdrücke in Aexp erhalten wir

□ „Wenn folgende Bedingung gilt:

- $\forall n \in \mathbb{N}: P(n)$
- $\forall x \in \text{Loc}: P(x)$
- $\forall a_1, a_2 \in \text{Aexp}: [(P(a_1) \wedge P(a_2)) \Rightarrow P(a_1 + a_2)]$
- $\forall a_1, a_2 \in \text{Aexp}: [(P(a_1) \wedge P(a_2)) \Rightarrow P(a_1 - a_2)]$
- $\forall a_1, a_2 \in \text{Aexp}: [(P(a_1) \wedge P(a_2)) \Rightarrow P(a_1 * a_2)]$

dann gilt auch

- $\forall a \in \text{Aexp}: P(a)$ “

Das Beweisprinzip der strukturellen Induktion auf Aexp ist also ein Spezialfall der wohlfundierten Induktion.

Übersicht: Modul 6

Äquivalenz zweier Programme

- ☐ Beweis mit Fallunterscheidung

Nichtterminierung eines Programms

- ☐ Widerspruchsbeweis

Induktionsprinzipien

- ☐ Induktion auf den natürlichen Zahlen
- ☐ wohlfundierte Induktion
- ☐ Rechtfertigung der Induktion auf den natürlichen Zahlen
- ☐ strukturelle Induktion auf Aexp
- ☐ Rechtfertigung der strukturellen Induktion auf Aexp

Deterministische Auswertung von Ausdrücken

- ☐ Beweis mit struktureller Induktion

Deterministische Auswertung (1)

Theorem

Für alle $a \in A_{\text{exp}}$, $m, m' \in \mathbb{N}$ und alle Zustände σ gilt:
wenn $\langle a, \sigma \rangle \Downarrow m$ und $\langle a, \sigma \rangle \Downarrow m'$ herleitbar sind, dann gilt $m' = m$.

Beweis

- Wir verwenden das Beweisprinzip der strukturellen Induktion für
 $P(a) = \forall \sigma: \forall m, m' \in \mathbb{N}: [(\langle a, \sigma \rangle \Downarrow m \text{ ist herleitbar})$
 $\wedge \langle a, \sigma \rangle \Downarrow m' \text{ ist herleitbar}] \Rightarrow m = m']$

- Wir müssen 5 Fälle beweisen:

Fall $a = n$ für ein $n \in \mathbb{N}$

- Die Herleitung von $\langle n, \sigma \rangle \Downarrow n'$ kann nur folgende Form haben:

$$\frac{rN}{\langle n, \sigma \rangle \Downarrow n}$$

Daher muss $m = n = m'$ gelten.

Deterministische Auswertung (2)

Beweis (Fortsetzung)

Fall $a=x$ für ein $x \in \text{Loc}$

- Die Herleitung von $\langle x, \sigma \rangle \Downarrow n'$ kann nur folgende Form haben:

$$\text{rLoc} \frac{}{\langle x, \sigma \rangle \Downarrow n'} \sigma(x)=n'$$

Daher muss $m = \sigma(x) = m'$ gelten.

Fall $a=a_1+a_2$ für $a_1, a_2 \in \text{Aexp}$ wobei $P(a_1)$ und $P(a_2)$ gelten

- Die Herleitungen von $\langle a, \sigma \rangle \Downarrow m$ und $\langle a, \sigma \rangle \Downarrow m'$ können nur folgende Formen haben:

$$\text{r+} \frac{\begin{array}{c} \vdots \mathcal{H}_1 \\ \langle a_1, \sigma \rangle \Downarrow m_1 \end{array} \quad \begin{array}{c} \vdots \mathcal{H}_2 \\ \langle a_2, \sigma \rangle \Downarrow m_2 \end{array}}{\langle a_1+a_2, \sigma \rangle \Downarrow m} \quad m=m_1+m_2$$

$$\text{r+} \frac{\begin{array}{c} \vdots \mathcal{H}_1' \\ \langle a_1, \sigma \rangle \Downarrow m_1' \end{array} \quad \begin{array}{c} \vdots \mathcal{H}_2' \\ \langle a_2, \sigma \rangle \Downarrow m_2' \end{array}}{\langle a_1+a_2, \sigma \rangle \Downarrow m'} \quad m'=m_1'+m_2'$$

- Aus $P(a_1)$ und $P(a_2)$ folgen $m_1=m_1'$ und $m_2=m_2'$.

Daher muss $m = m_1+m_2 = m_1'+m_2' = m'$ gelten.

Deterministische Auswertung (3)

Beweis (Fortsetzung)

Fall $a=a_1-a_2$ für $a_1, a_2 \in A_{exp}$ wobei $P(a_1)$ und $P(a_2)$ gelten

□ ...

Fall $a=a_1*a_2$ für $a_1, a_2 \in A_{exp}$ wobei $P(a_1)$ und $P(a_2)$ gelten

□ ...

Deterministische Auswertung (4)

Theorem

Für alle $b \in \text{Bexp}$, $t, t' \in T$ und alle Zustände σ gilt:
wenn $\langle b, \sigma \rangle \Downarrow t$ und $\langle b, \sigma \rangle \Downarrow t'$ herleitbar sind, dann gilt $t' = t$.

Beweis

□ ...

Übersicht: Modul 6

Äquivalenz zweier Programme

- ☐ Beweis mit Fallunterscheidung

Nichtterminierung eines Programms

- ☐ Widerspruchsbeweis

Induktionsprinzipien

- ☐ Induktion auf den natürlichen Zahlen
- ☐ wohlfundierte Induktion
- ☐ Rechtfertigung der Induktion auf den natürlichen Zahlen
- ☐ strukturelle Induktion auf Aexp
- ☐ Rechtfertigung der strukturellen Induktion auf Aexp

Deterministische Auswertung von Ausdrücken

- ☐ Beweis mit struktureller Induktion

Rückblick

Einige wesentliche Lernziele dieses Moduls

- ☐ Wie kann ich eine operationelle Semantik zur Verifikation nutzen?
- ☐ Beherrschung elementarer Verifikationstechniken:
 - ☐ Fallunterscheidung
 - ☐ Widerspruchsbeweis
 - ☐ wohlfundierte Induktion
 - ☐ strukturelle Induktion
 - ☐ ... wird in Modul 7 vervollständigt ...
- ☐ Fähigkeit, ein Induktionsprinzip zu rechtfertigen

Literatur

Glynn Winskel

The Formal Semantics of Programming Languages; Kapitel 3
The MIT Press, 1993.