

Informatik I

9. Nachweis von Programmeigenschaften

Jan-Georg Smaus

Albert-Ludwigs-Universität Freiburg

2. Dezember 2010

Informatik I

2. Dezember 2010 — 9. Nachweis von Programmeigenschaften

9.1 Nachweis von Programmeigenschaften

9.1 Nachweis von Programmeigenschaften

- Korrektheit: Ein verklausuliertes Programm
- Garantierte Terminierung
- Alternative Rekursion
- Listen

Ein verklausuliertes Programm

Was tut folgendes Programm?

```
(define h
  (lambda (n)
    (if (zero? n)
        0
        (+ (h (- n 1))
            (/ 1 (* n (+ n 1)))))))
```

Behauptung: $(\forall n) (h\ n) = \frac{n}{n+1}$.

Zeige, dass Behauptung gilt.

Vorgehensweise

- ▶ Unsere Beweise sind eine Mischung aus Programmtext und mathematischer Syntax. Wir haben es nicht mit **konkreten** Ausdrücken zu tun, sondern Ausdrücken, in denen **mathematische** Variablen, also Variablen auf der **Meta-Ebene** vorkommen. Beispiel: $(+ 1 (n + 1))$.
- ▶ Wir werten diese **abstrakten** Ausdrücke gemäß dem Substitutionsmodell aus.
- ▶ Ohne Aufhebens gehen wir immer davon aus, dass die eingebauten mathematischen Operationen wie $+$ $*$... das Übliche tun. Dies erlaubt uns z.B. darauf zu vertrauen dass $(+ 1 (n + 1))$ zu $n + 2$ reduziert wird.
- ▶ Auch „kleine“ Lemmata wie $n + m = m + n$ verwenden wir nach Belieben.
- ▶ Achten Sie sehr auf den Unterschied zwischen „Programmtext $() + *$ “ und „Mathematik $() + \frac{i}{n}$ “.

Nochmals das Programm

```
(define h
  (lambda (n)
    (if (zero? n)
        0
        (+ (h (- n 1))
           (/ 1 (* n (+ n 1)))))))
```

Behauptung: $(\forall n) (h\ n) = \frac{n}{n+1}$.

Zeige, dass Behauptung gilt.

Induktionsanfang $P(0)$

$$(h\ 0) = (if\ (zero?\ 0)\ 0\ \dots) = 0 = \frac{0}{1}$$

gemäß Substitutionsmodell.

Induktionsschritt $P(n) \Rightarrow P(n + 1)$

$$\begin{aligned} (h\ (n + 1)) &= (if\ (zero?\ (n + 1))\ 0\ \dots) \\ &= (+\ (h\ (-\ (n + 1)\ 1)) \\ &\quad (/ 1\ (*\ (n + 1)\ (+\ (n + 1)\ 1)))) \\ &= (+\ (h\ n) \\ &\quad (/ 1\ (*\ (n + 1)\ (n + 2)))) \\ &\stackrel{IH}{=} (+\ \frac{n}{n + 1}\ \frac{1}{(n + 1) \cdot (n + 2)}) \\ &= \frac{n \cdot (n + 2)}{(n + 1) \cdot (n + 2)} + \frac{1}{(n + 1) \cdot (n + 2)} \\ &= \frac{n^2 + 2n + 1}{(n + 1) \cdot (n + 2)} = \frac{(n + 1)^2}{(n + 1) \cdot (n + 2)} \\ &= \frac{n + 1}{n + 2} \end{aligned}$$

Konstruktionsanleitung

Satz

Jede durch die Konstruktionsanleitung für natürliche Zahlen definierte Funktion liefert immer ein Ergebnis.

D.h. sei $(f : \text{natural} \rightarrow \text{natural})$ definiert durch

```
(define f
  (lambda (n)
    (if (zero? n)
        basis
        (step (f (- n 1)) n))))
```

wobei *basis* ein Ausdruck mit Wert aus \mathbb{N} und *step* eine totale (und berechenbare) Funktion $(\text{natural} \text{ natural} \rightarrow \text{natural})$ ist. Dann gilt: $(\forall n \in \mathbb{N})$ ist $(f\ n)$ definiert und $(f\ n) \in \mathbb{N}$.

Beweis

Induktionsbasis:

$$\begin{aligned} (f\ 0) &= (\text{if } (\text{zero? } 0) \text{ basis } (\text{step } \dots)) \\ &= \text{basis} \in \mathbb{N} \end{aligned}$$

Induktionsschritt $P(n) \Rightarrow P(n+1)$:

$$\begin{aligned} (f\ (n+1)) &= (\text{if } (\text{zero? } (n+1)) \dots) \\ &= (\text{step } (f\ (-\ (n+1)\ 1))\ (n+1)) \\ &= (\text{step } (f\ n)\ (n+1)) \end{aligned}$$

Nach Induktionshypothese ist $(f\ n)$ definiert und gleich $f(n) \in \mathbb{N}$, also

$$= (\text{step } f(n)\ (n+1)) \in \mathbb{N}$$

nach Voraussetzung über *step*.

Primitive Rekursion

Unsere Konstruktionsanleitung, so abstrakt unter Verwendung von *basis* und *step* formuliert ...

```
(define f
  (lambda (n)
    (if (zero? n)
        basis
        (step (f (- n 1)) n))))
```

... hat in der **Berechnungstheorie** (3. Semester) einen Namen: *f* ist eine **primitiv rekursive Funktion** (vorausgesetzt, *basis* und *step* sind primitiv rekursiv).

Abweichung von der Konstruktionsanleitung

Wir haben am Beispiel von `factorial` gesehen, dass Funktionen auf den natürlichen Zahlen, die von der Konstruktionsanleitung abweichen, evtl. nicht terminieren.

Trotzdem können Programme, die von der Konstruktionsanleitung abweichen, terminieren.

Man muss nicht unbedingt von n nach $n-1$ gehen ...

Noch ein Programm

```
(define l
  (lambda (n)
    (if (zero? n)
        0
        (+ 1 (l (quotient n 2))))))
```

also $l(0) = 0$, $l(n) = 1 + l(\lfloor \frac{n}{2} \rfloor)$, wobei $\lfloor x \rfloor$ die größte ganze Zahl z mit $x \geq z$ ist.

n	0	1	2	3	4	5	6	7	8	9...	15	16	17	...
$l(n)$	0	1	2	2	3	3	3	3	4	4...	4	5	5	...

Vermutung

$$l(n) = \lceil \log_2(n+1) \rceil$$

wobei ist $\lceil x \rceil$ die kleinste ganze Zahl z mit $x \leq z$ ist.

Vorabbemerkung

Warum präsentiere ich ein Programm sodass

$$l(n) = \lceil \log_2(n+1) \rceil$$

anstatt des einfacheren

$$l(n) = \lceil \log_2(n) \rceil ?$$

Das wäre das Programm

```
(define l
  (lambda (n)
    (if (= n 1)
        0
        (+ 1 (l (quotient n 2))))))
```

Dies wäre keine totale Prozedur der Sorte (natural \rightarrow natural). Die Eingabe 0, für die l undefiniert ist, müssten wir gesondert abfangen und eine Fehlermeldung generieren.

Induktion

Wir kennen

Induktionsschema

Falls $P(0)$ (**Induktionsbasis, -anfang**)

und

$(\forall n) P(n) \Rightarrow P(n')$ (**Induktionsschritt**)

dann

$(\forall n \in \mathbb{N}) P(n)$. ($P(n)$ **Induktionshypothese, -behauptung**)

Das vorliegende Programm erfordert eine andere Art von Induktion. Wir nutzen aus, dass

$$\{(\lfloor \frac{n}{2} \rfloor, n) \mid n \in \mathbb{N}^+\} \subseteq <$$

und „<“ wohlfundiert ist.

Erinnerung: Induktion und wohlfundierte Relationen

1. **Induktionsbasis:** $(\forall x \in A) ((\exists y \in A) y R x) \Rightarrow P(x)$
2. **Induktionsschritt:** $(\forall x \in A) ((\forall y \in A) y R x \Rightarrow P(y)) \Rightarrow P(x)$

also in unserem Fall

1. **Induktionsbasis:** $P(0)$
2. **Induktionsschritt:** $(\forall x \in \mathbb{N}) ((\forall y \in \mathbb{N}) y < x \Rightarrow P(y)) \Rightarrow P(x)$, d.h., zeige $P(x)$ wobei man für alle $y < x$ verwenden kann dass $P(y)$.

Beweis

Induktionsbasis $n = 0$

$$\begin{aligned}
 (1\ 0) &= (\text{if } (\text{zero? } 0) 0 \dots) \\
 &= 0 \\
 &= \lceil \log_2 1 \rceil \\
 &= \lceil \log_2(0 + 1) \rceil
 \end{aligned}$$

Beweis

Induktionsschritt: $n > 0$

$$\begin{aligned}
 l(n) &= (\text{if } (\text{zero? } n) 0 \dots) \\
 &= (+ 1 (1 (\text{quotient } n 2))) \\
 &= (+ 1 (1 \lfloor \frac{n}{2} \rfloor)) \\
 &\stackrel{\text{IH}}{=} (+ 1 \lceil \log_2(\lfloor \frac{n}{2} \rfloor + 1) \rceil) \\
 &= 1 + \lceil \log_2(\lfloor \frac{n}{2} \rfloor + 1) \rceil \\
 &= \lceil 1 + \log_2(\lfloor \frac{n}{2} \rfloor + 1) \rceil \\
 &= \lceil \log_2(2 \cdot (\lfloor \frac{n}{2} \rfloor + 1)) \rceil \\
 &= \lceil \log_2(2 \cdot \lfloor \frac{n}{2} \rfloor + 2) \rceil
 \end{aligned}$$

Es geht gleich weiter ...

Beweis

Induktionsschritt: $n > 0$ (Fortsetzung 1)

Wir haben bisher gezeigt:

$$l(n) = \lceil \log_2(2 \cdot \lfloor \frac{n}{2} \rfloor + 2) \rceil$$

1. Fall: $n = 2k$ mit $k > 0$

$$\begin{aligned}
 &= \lceil \log_2(\lfloor 2 \cdot \frac{2k}{2} \rfloor + 2) \rceil \\
 &= \lceil \log_2(2k + 2) \rceil \\
 &\stackrel{(*)}{=} \lceil \log_2(2k + 1) \rceil \\
 &= \lceil \log_2(n + 1) \rceil
 \end{aligned}$$

Beweis

Nebenrechnung (*): $\lceil \log_2(2k + 2) \rceil = \lceil \log_2(2k + 1) \rceil$ für alle $k > 0$ Wie jede natürliche Zahl ≥ 3 liegt $2k + 2$ zwischen zwei benachbarten Zweierpotenzen, d.h., es gibt ein $j \in \mathbb{N}^+$ so dass

$$2^j < 2k + 2 \leq 2^{j+1}. \quad (1)$$

Da $2k + 2 \in \mathbb{N}$, folgt hieraus $2^j \leq 2k + 1 < 2^{j+1}$.Da $2^j = 2k + 1$ unmöglich ist, kann man verschärfen:

$$2^j < 2k + 1 < 2^{j+1}. \quad (2)$$

Nun wenden wir auf (1) und (2) den Zweierlogarithmus an

$$\begin{aligned}
 j &< \log_2(2k + 2) \leq j + 1 \\
 j &< \log_2(2k + 1) < j + 1.
 \end{aligned}$$

Hieraus folgt

$$\begin{aligned}
 \lceil \log_2(2k + 2) \rceil &= j + 1 \\
 \lceil \log_2(2k + 1) \rceil &= j + 1.
 \end{aligned}$$

Beweis

Induktionsschritt: $n > 0$ (Fortsetzung 2)

Wir haben bisher gezeigt:

$$l(n) = \lceil \log_2(2 \cdot \lfloor \frac{n}{2} \rfloor + 2) \rceil$$

2. Fall: $n = 2k + 1$ mit $k \geq 0$

$$\begin{aligned} &= \lceil \log_2(2 \cdot \lfloor \frac{2k+1}{2} \rfloor + 2) \rceil \\ &= \lceil \log_2(2k + 2) \rceil \\ &= \lceil \log_2(n + 1) \rceil \end{aligned}$$

Erinnerung: Peano-Axiome

Definition (Peano-Axiome)

Die Menge \mathbb{N} der **natürlichen Zahlen** ist definiert durch:

- P1 $0 \in \mathbb{N}$ (Null)
- P2 $(\forall n \in \mathbb{N}) n' \in \mathbb{N}$ (Nachfolger)
- P3 $(\forall n \in \mathbb{N}) n' \neq 0$ (Nachfolger ist nicht Null)
- P4 $(\forall m, n \in \mathbb{N}) m \neq n \Rightarrow m' \neq n'$ (Nachfolger ist injektiv)
- P5 Für jede Menge $M \subseteq \mathbb{N}$ mit $0 \in M$ und $(\forall n) n \in M \Rightarrow n' \in M$ gilt $M = \mathbb{N}$ (Induktionsaxiom)

Listen

Definition

Die Menge M^* der **Listen über M** ist induktiv definiert (P3+P4) durch

1. $\text{empty} \in M^*$ (die **leere Liste**) (P1)
2. $(\forall a \in M) (\forall xs \in M^*) (\text{cons } a \text{ } xs) \in M^*$
(nichtleere Liste mit Kopf a und Rumpf (Rest) xs) (P2)
3. In M^* sind nur die durch (1) und (2) definierten Elemente enthalten (P5)

Dies ist analog zu den Peano-Axiomen.

Allgemein bezeichnet Σ^* die Menge der **Wörter** über dem **Alphabet** Σ .

Listeninduktion

Definition (Das Schema der Listeninduktion)

Sei $P(xs)$ eine Eigenschaft von Listen über M . Es gilt $(\forall xs \in M^*) P(xs)$ genau dann, wenn

1. Induktionsbasis:
 $P(\text{empty})$.
2. Induktionsschritt:
 $(\forall a \in M) (\forall xs \in M^*) P(xs) \Rightarrow P((\text{cons } a \text{ } xs))$.

Beispiel: Listenverkettung

```
(: append ((list-of %a) (list-of %a) -> (list-of %a)))
(define append
  (lambda (xs ys)
    (cond
      ((empty? xs)
       ys)
      ((cons? xs)
       (cons (first xs) (append (rest xs) ys))))))
```

Die Prozedur ist auf dem ersten Argument rekursiv.

Eigenschaften von append

1. $(\text{append empty } bs) = bs$
2. $(\text{append } as \text{ empty}) = as$
3. $(\text{append } (\text{append } as \text{ } bs) \text{ } cs) = (\text{append } as \text{ } (\text{append } bs \text{ } cs))$

Beweise

- ① $(\text{append empty } bs) = bs$

$(\text{append empty } bs) = bs$
folgt direkt aus dem Substitutionsmodell.

Beweise

- ② $(\text{append } as \text{ empty}) = as$

$(\text{append } as \text{ empty}) = as$

Listeninduktion über as .

Induktionsbasis: $as = \text{empty}$

$(\text{append empty empty}) = \text{empty}$

folgt aus dem Substitutionsmodell.

Induktionsschritt: $as = (\text{cons } a \text{ } as')$

$(\text{append } (\text{cons } a \text{ } as') \text{ empty})$

$\stackrel{\text{Sub.}}{=} (\text{cons } (\text{first } (\text{cons } a \text{ } as'))$
 $\quad (\text{append } (\text{rest } (\text{cons } a \text{ } as')) \text{ empty}))$

$\stackrel{\text{Sub.}}{=} (\text{cons } a \text{ } (\text{append } as' \text{ empty}))$

$\stackrel{\text{IH}}{=} (\text{cons } a \text{ } as')$

Beweise

$$\textcircled{3} \text{ (append (append as bs) cs) = (append as (append bs cs))}$$

Listeninduktion über as .

Induktionsbasis: $as = \text{empty}$

$$\begin{aligned} & \text{(append (append empty bs) cs)} \\ \stackrel{\textcircled{2}}{=} & \text{(append bs cs)} \\ \stackrel{\textcircled{2}}{=} & \text{(append empty (append bs cs))} \end{aligned}$$

Beweise

$$\textcircled{3} \text{ (append (append as bs) cs) = (append as (append bs cs))}$$

Listeninduktion über as .

Induktionsschritt: $as = (\text{cons } a \text{ } as')$

$$\begin{aligned} & \text{(append (append (cons } a \text{ } as') bs) cs)} \\ \stackrel{\text{Sub.}}{=} & \text{(append (cons } a \text{ (append } as' \text{ bs)) cs)} \\ \stackrel{\text{Sub.}}{=} & \text{(cons } a \text{ (append (append } as' \text{ bs) cs))} \\ \stackrel{\text{IH}}{=} & \text{(cons } a \text{ (append } as' \text{ (append bs cs)))} \\ \stackrel{\text{Sub.}^{-1}}{=} & \text{(append (cons } a \text{ } as') \text{(append bs cs))} \\ = & \text{(append as (append bs cs))} \end{aligned}$$

Zusammenfassung

- ▶ Nachweis, was ein Programm auf den natürlichen Zahlen ausrechnet
- ▶ Nachweis der Terminierung
- ▶ Rekursion einmal nicht über den direkten Vorgänger einer Zahl
- ▶ Listen und Beweise für Prozeduren, die darauf rechnen