Theorie der Informatik

14. primitive Rekursion und μ-Rekursion

Malte Helmert Gabriele Röger

Universität Basel

16. April 2014

Überblick: Vorlesung

Vorlesungsteile

I. Logik ✓

II. Automatentheorie und formale Sprachen ✓

III. Berechenbarkeitstheorie

IV. Komplexitätstheorie

Überblick: Berechenbarkeitstheorie

III. Berechenbarkeitstheorie

12. Turing-Berechenbarkeit ✓
13. LOOP-, WHILE- und GOTO-Berechenbarkeit ✓
14. primitive Rekursion und μ-Rekursion
15. Ackermannfunktion
16. Entscheidbarkeit, Reduktionen, Halteproblem
17. Postsches Korrespondenzproblem

~~Unentscheidbare Grammatik-Probleme~~

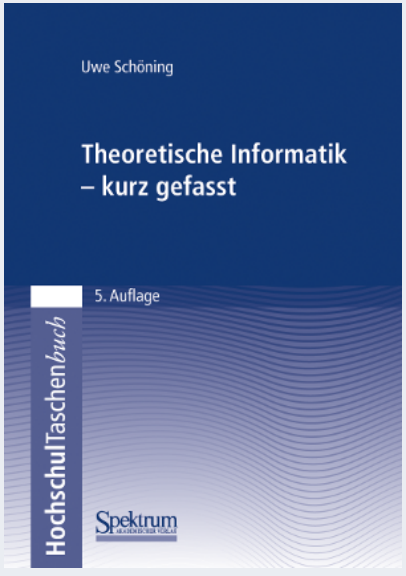
~~Gödelscher Satz und diophantische Gleichungen~~

Nachlesen

Literatur zu diesem Vorlesungskapitel

Theoretische Informatik - kurz gefasst
von Uwe Schöning (5. Auflage)

- Kapitel 2.4



Einleitung

Formale Berechnungsmodelle: primitive und μ-Rekursion

Formale Berechnungsmodelle

- Turingmaschinen
- LOOP-, WHILE-, GOTO-Programme
- primitiv rekursive Funktionen, μ-rekursive Funktionen

In diesem Kapitel lernen wir zwei Berechnungskonzepte kennen, die von Turingmaschinen und imperativen Programmiersprachen grundverschieden sind, da sie keine „Zuweisungen“ oder „Wertveränderungen“ kennen:

- primitiv rekursive Funktionen
- μ-rekursive Funktionen

primitive Rekursion: Grundidee

Primitive Rekursion und μ-Rekursion sind Berechnungsmodelle für Funktionen über (ein oder mehreren) natürlichen Zahlen, die auf folgender Grundidee aufbauen:

- einige einfache Basisfunktionen werden als berechenbar vorausgesetzt (sind „nach Definition“) berechenbar
- aus diesen Funktionen kann man nach gewissen „Konstruktionsregeln“ neue Funktionen zusammenbauen

Basisfunktionen und Einsetzung

Basisfunktionen

Definition (Basisfunktionen)

Die Basisfunktionen sind die folgenden Funktionen in $\mathbb{N}_0^k \to \mathbb{N}_0$:

- konstante Nullfunktion $null : \mathbb{N}_0 \to \mathbb{N}_0$:
  $null(x) = 0 \quad$ für alle $x \in \mathbb{N}_0$
- Nachfolgerfunktion $succ : \mathbb{N}_0 \to \mathbb{N}_0$:
  $succ(x) = x + 1 \quad$ für alle $x \in \mathbb{N}_0$
- Projektionsfunktionen $\pi_j^i : \mathbb{N}_0^i \to \mathbb{N}_0$ für alle $i \geq 1$, $1 \leq j \leq i$:
  $\pi_j^i(x_1, \ldots, x_i) = x_j \quad$ für alle $x_1, \ldots, x_i \in \mathbb{N}_0$
  (schliesst insbesondere Identitätsfunktion ein)

Einsetzungsschema

Definition (Einsetzungsschema)

Die Funktion $f : \mathbb{N}_0^k \to \mathbb{N}_0$ entsteht durch das Einsetzungsschema (durch Einsetzung, durch Komposition) aus den Funktionen $h : \mathbb{N}_0^i \to \mathbb{N}_0$, $g_1, \ldots, g_i : \mathbb{N}_0^k \to \mathbb{N}_0$, wenn gilt:

$$f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), \ldots, g_i(x_1, \ldots, x_k))$$

für alle $x_1, \ldots, x_k \in \mathbb{N}_0$.

Einsetzung: Beispiele

Erinnerung: $f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), \ldots, g_i(x_1, \ldots, x_k))$

Beispiel (Einsetzung)

1. Betrachte $f_1 : \mathbb{N}_0 \to \mathbb{N}_0$ mit $f_1(x) = 1$ für alle $n \in \mathbb{N}_0$.

f_1 entsteht durch Einsetzung aus *succ* und *null*,
da $f_1(x) = succ(null(x))$ für alle $x \in \mathbb{N}_0$.

$\rightsquigarrow$ Einsetzungsregel mit $k = 1$, $i = 1$, $h = succ$, $g_1 = null$.

Einsetzung: Beispiele

Erinnerung: $f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), \ldots, g_i(x_1, \ldots, x_k))$

Beispiel (Einsetzung)

2. Betrachte $f_2 : \mathbb{N}_0^2 \to \mathbb{N}_0$ mit $f_2(x, y) = y + 1$ für alle $n \in \mathbb{N}_0$.

f_2 entsteht durch Einsetzung aus *succ* und π_2^2,
da $f_2(x, y) = succ(\pi_2^2(x, y))$ für alle $x, y \in \mathbb{N}_0$.

$\rightsquigarrow$ Einsetzungsregel mit $k = ?$, $i = ?$, $h = ?$, $g_\ldots = ?$

Einsetzung: Beispiele

Erinnerung: $f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), \ldots, g_i(x_1, \ldots, x_k))$

Beispiel (Einsetzung)

3. Sei $r : \mathbb{N}_0^3 \to \mathbb{N}_0$.

Betrachte die Funktion $f_3 : \mathbb{N}_0^4 \to \mathbb{N}_0$ mit $f_3(a, b, c, d) = r(c, c, b)$.

f_3 entsteht durch Einsetzung aus r und den Projektionsfunktionen, da $f(a, b, c, d) = r(\pi_3^4(a, b, c, d), \pi_3^4(a, b, c, d), \pi_2^4(a, b, c, d))$.

$\rightsquigarrow$ Einsetzungsregel mit $k = ?$, $i = ?$, $h = ?$, $g_\ldots = ?$

Einsetzung: Beispiele

Erinnerung: $f(x_1, \ldots, x_k) = h(g_1(x_1, \ldots, x_k), \ldots, g_i(x_1, \ldots, x_k))$

Beispiel (Einsetzung)

3. Sei $r : \mathbb{N}_0^3 \to \mathbb{N}_0$.

Betrachte die Funktion $f_3 : \mathbb{N}_0^4 \to \mathbb{N}_0$ mit $f_3(a, b, c, d) = r(c, c, b)$.

f_3 entsteht durch Einsetzung aus r und den Projektionsfunktionen, da $f(a, b, c, d) = r(\pi_3^4(a, b, c, d), \pi_3^4(a, b, c, d), \pi_2^4(a, b, c, d))$.

$\rightsquigarrow$ Einsetzungsregel mit $k = ?$, $i = ?$, $h = ?$, $g_\ldots = ?$

$\rightsquigarrow$ Einsetzung und Projektion erlauben generell Vertauschung und Wiederholung von Argumenten.

Quiz

Primitive Rekursion

Primitives Rekursionsschema

Definition (primitives Rekursionsschema)

Seien $g : \mathbb{N}_0^k \to \mathbb{N}_0$ und $h : \mathbb{N}_0^{k+2} \to \mathbb{N}_0$.

Die Funktion $f : \mathbb{N}_0^{k+1} \to \mathbb{N}_0$ entsteht durch das primitive Rekursionsschema (durch primitive Rekursion) aus g und h, wenn gilt:

$$f(0, x_1, \ldots, x_k) = g(x_1, \ldots, x_k)$$
$$f(n+1, x_1, \ldots, x_k) = h(f(n, x_1, \ldots, x_k), n, x_1, \ldots, x_k)$$

für alle $n, x_1, \ldots, x_k \in \mathbb{N}_0$.

Beispiel $k = 1$:

$$f(0, x) = g(x)$$
$$f(n+1, x) = h(f(n, x), n, x)$$

Primitive Rekursion: Beispiele

Erinnerung primitive Rekursion mit $k = 1$:

$f(0, x) = g(x) \qquad f(n+1, x) = h(f(n, x), n, x)$

Beispiel (primitive Rekursion)

1. Sei $g(a) = a$ und $h(a, b, c) = a + 1$. Welche Funktion entsteht mit primitivem Rekursionsschema aus g und h?

$f(0, x) = g(x) = x$
$f(1, x) = h(f(0, x), 0, x) = h(x, 0, x) = x + 1$
$f(2, x) = h(f(1, x), 1, x) = h(x + 1, 1, x) = (x + 1) + 1 = x + 2$
$f(3, x) = h(f(2, x), 2, x) = h(x + 2, 2, x) = (x + 2) + 1 = x + 3$
$\ldots$
$\rightsquigarrow f(a, b) = a + b$

Primitive Rekursion: Beispiele

Erinnerung primitive Rekursion mit $k = 1$:

$f(0, x) = g(x)$ $\qquad$ $f(n+1, x) = h(f(n, x), n, x)$

Beispiel (primitive Rekursion)

2. Sei $g(a) = 0$ und $h(a, b, c) = a + c$. Welche Funktion entsteht mit dem primitiven Rekursionsschema aus g und h?

Primitive Rekursion: Beispiele

Erinnerung primitive Rekursion mit $k = 1$:

$f(0, x) = g(x)$ $\qquad$ $f(n+1, x) = h(f(n, x), n, x)$

Beispiel (primitive Rekursion)

3. Sei $g(a) = 0$ und $h(a, b, c) = b$. Welche Funktion entsteht mit dem primitiven Rekursionsschema aus g und h?

$f(0, x) = g(x) = 0$
$f(1, x) = h(f(0, x), 0, x) = 0$
$f(2, x) = h(f(1, x), 1, x) = 1$
$f(3, x) = h(f(2, x), 2, x) = 2$
...
$\rightsquigarrow f(a, b) = \max(a - 1, 0)$

$\rightsquigarrow$ mit Projektion und Einsetzung: modifizierte Vorgängerfunktion

Primitiv rekursive Funktionen

Definition (primitiv rekursive Funktion)

Die Menge der primitiv rekursiven Funktionen (PRFs) ist induktiv durch endliche Anwendung der folgenden Regeln definiert:

1. Jede Basisfunktion ist eine PRF.
2. Funktionen, die durch das Einsetzungsschema aus PRFs gewonnen werden können, sind PRFs.
3. Funktionen, die durch das primitive Rekursionsschema aus PRFs gewonnen werden können, sind PRFs.

Anmerkung: primitiv rekursive Funktionen sind immer total. (Warum?)

Primitiv rekursive Funktionen: Beispiele

Beispiel

Die folgenden Funktionen sind PRFs:

- $succ(x) = x + 1$ ($\rightsquigarrow$ Basisfunktion)
- $add(x, y) = x + y$ ($\rightsquigarrow$ gezeigt)
- $mul(x, y) = x \cdot y$ ($\rightsquigarrow$ gezeigt)
- $pred(x) = \max(x - 1, 0)$ ($\rightsquigarrow$ gezeigt)
- $sub(x, y) = \max(x - y, 0)$ ($\rightsquigarrow$ Hausaufgaben)
- $binom_2(x) = \binom{x}{2}$ ($\rightsquigarrow$ Hausaufgaben)

Notation: im Folgenden schreiben wir $x \ominus y$ für die modifizierte Subtraktion $sub(x, y)$ (also z. B. $pred(x) = x \ominus 1$).

Und jetzt?

Hat das denn irgend etwas
mit den vorigen Kapiteln zu tun?

$\rightsquigarrow$ Noch etwas Geduld!

μ-Rekursion

μ-Operator

Definition (μ-Operator)

Sei $f : \mathbb{N}_0^{k+1} \to \mathbb{N}_0$ mit $k \geq 1$.

Die Funktion $\mu f : \mathbb{N}_0^k \to \mathbb{N}_0$ ist definiert durch

$$(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0 \text{ und für alle } m < n \text{ ist } f(m, x_1, \ldots, x_k) \text{ definiert}\}$$

Falls die zu minimierende Menge leer ist,
ist $(\mu f)(x_1, \ldots, x_k)$ undefiniert.

μ heisst μ-Operator.

μ-Operator: Beispiele

Erinnerung μf: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0$ und für alle $m < n$ ist $f(m, x_1, \ldots, x_k)$ definiert$\}$

wenn f total: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0\}$

μ-Operator: Beispiele

Erinnerung μf: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0$ und für alle $m < n$ ist $f(m, x_1, \ldots, x_k)$ definiert$\}$

wenn f total: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0\}$

Beispiel (μ-Operator)

1. Sei $f(a, b, c) = b \ominus (a \cdot c)$

Welche Funktion ist μf?

$$
\begin{aligned}
(\mu f)(x_1, x_2) &= \min\{n \in \mathbb{N}_0 \mid f(n, x_1, x_2) = 0\} \\
&= \min\{n \in \mathbb{N}_0 \mid x_1 \ominus (n \cdot x_2) = 0\} \\
&= \begin{cases} 0 & \text{wenn } x_1 = 0 \\ \text{undefiniert} & \text{wenn } x_1 \neq 0, x_2 = 0 \\ \lceil \frac{x_1}{x_2} \rceil & \text{sonst} \end{cases}
\end{aligned}
$$

μ-Operator: Beispiele

Erinnerung μf: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0$ und für alle $m < n$ ist $f(m, x_1, \ldots, x_k)$ definiert$\}$

wenn f total: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0\}$

Beispiel (μ-Operator)

2. Sei $f(a, b) = b \ominus (a \cdot a)$

Welche Funktion ist μf?

μ-Operator: Beispiele

Erinnerung μf: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0$ und für alle $m < n$ ist $f(m, x_1, \ldots, x_k)$ definiert$\}$

wenn f total: $(\mu f)(x_1, \ldots, x_k) = \min\{n \in \mathbb{N}_0 \mid f(n, x_1, \ldots, x_k) = 0\}$

Beispiel (μ-Operator)

3. Sei $f(a, b) = (b \ominus (a \cdot a)) + ((a \cdot a) \ominus b)$.

Welche Funktion ist μf?

μ-rekursive Funktionen

Definition (μ-rekursive Funktion)

Die Menge der μ-rekursiven Funktionen (μRFs) ist induktiv durch endliche Anwendung der folgenden Regeln definiert:

1. Jede Basisfunktion ist eine μRF.
2. Funktionen, die durch das Einsetzungsschema aus μRFs gewonnen werden können, sind μRFs.
3. Funktionen, die durch das primitive Rekursionsschema aus μRFs gewonnen werden können, sind μRFs.
4. Funktionen, die durch den μ-Operator aus μRFs gewonnen werden können, sind μRFs.

Zusammenhänge

Kodierung und Dekodierung

Betrachte die Funktion $\mathit{encode} : \mathbb{N}_0^2 \to \mathbb{N}_0$ mit:

$$\mathit{encode}(x, y) := \binom{x + y + 1}{2} + x$$

Kodierung und Dekodierung

Betrachte die Funktion $\mathit{encode} : \mathbb{N}_0^2 \to \mathbb{N}_0$ mit:

$$\mathit{encode}(x, y) := \binom{x + y + 1}{2} + x$$

- $\rightsquigarrow$ *encode* ist PRF

Kodierung und Dekodierung

Betrachte die Funktion $\mathit{encode} : \mathbb{N}_0^2 \to \mathbb{N}_0$ mit:

$$\mathit{encode}(x, y) := \binom{x + y + 1}{2} + x$$

- $\rightsquigarrow$ *encode* ist PRF
- $\rightsquigarrow$ *encode* ist bijektiv (ohne Beweis)

Kodierung und Dekodierung

Betrachte die Funktion $encode : \mathbb{N}_0^2 \to \mathbb{N}_0$ mit:

$$encode(x, y) := \binom{x + y + 1}{2} + x$$

- $\rightsquigarrow$ *encode* ist PRF
- $\rightsquigarrow$ *encode* ist bijektiv (ohne Beweis)

	$x = 0$	$x = 1$	$x = 2$	$x = 3$	$x = 4$
$y = 0$	0	2	5	9	14
$y = 1$	1	4	8	13	19
$y = 2$	3	7	12	18	25
$y = 3$	6	11	17	24	32
$y = 4$	10	16	23	31	40

Kodierung und Dekodierung

Betrachte die Umkehrfunktionen
$decode_1 : \mathbb{N}_0 \to \mathbb{N}_0$ und $decode_2 : \mathbb{N}_0 \to \mathbb{N}_0$ von $encode$:

$$decode_1(encode(x, y)) = x$$
$$decode_2(encode(x, y)) = y$$

Kodierung und Dekodierung

Betrachte die Umkehrfunktionen
$decode_1 : \mathbb{N}_0 \to \mathbb{N}_0$ und $decode_2 : \mathbb{N}_0 \to \mathbb{N}_0$ von *encode* :

$$decode_1(encode(x, y)) = x$$
$$decode_2(encode(x, y)) = y$$

- $decode_1$ und $decode_2$ sind PRFs (ohne Beweis)

Verhältnis zu anderen Berechnungsmodellen

Satz

1. *Jede primitiv rekursive Funktion ist LOOP-berechenbar.*
2. *Jede LOOP-berechenbare Funktion ist primitiv rekursiv.*
3. *Jede μ-rekursive Funktion ist WHILE-berechenbar.*
4. *Jede WHILE-berechenbare Funktion ist μ-rekursiv.*

Verhältnis zu anderen Berechnungsmodellen

Satz

1. *Jede primitiv rekursive Funktion ist LOOP-berechenbar.*
2. *Jede LOOP-berechenbare Funktion ist primitiv rekursiv.*
3. *Jede μ-rekursive Funktion ist WHILE-berechenbar.*
4. *Jede WHILE-berechenbare Funktion ist μ-rekursiv.*

Beweis.

1.+2.: $\rightsquigarrow$ Tafel

3.+4.: $\rightsquigarrow$ ohne Beweis $\square$

Zusammenfassung

Kapitel-Zusammenfassung

- Idee: baue komplexe Funktionen aus Basisfunktionen und Bauregeln zusammen.
- Basisfunktionen (B):
  - konstante Nullfunktion
  - Nachfolgerfunktion
  - Projektionsfunktionen
- Bauregeln:
  - Einsetzungsschema (E)
  - primitives Rekursionsschema (P)
  - μ-Operator (μ)
- primitiv rekursive Funktionen (PRFs):
  gebaut aus (B) + (E) + (P)
- μ-rekursive Funktionen (μRFs):
  gebaut aus (B) + (E) + (P) + (μ)

Übersicht Berechnungsformalismen

Folgerung

Sei $f : \mathbb{N}_0^k \to \mathbb{N}_0$ eine partielle Funktion.

Die folgenden Aussagen sind äquivalent:

- *f ist Turing-berechenbar.*
- *f ist WHILE-berechenbar.*
- *f ist GOTO-berechenbar.*
- *f ist μ-rekursiv.*

Übersicht Berechnungsformalismen

Folgerung

Sei $f : \mathbb{N}_0^k \to \mathbb{N}_0$ eine partielle Funktion.

Die folgenden Aussagen sind äquivalent:

- *f ist LOOP-berechenbar.*
- *f ist primitiv rekursiv.*

Ferner gilt:

- *Jede LOOP-berechenbare Funktion/primitiv rekursive Funktion ist Turing-/WHILE-/GOTO-berechenbar/μ-rekursiv.*
- *Die Umkehrung gilt im Allgemeinen nicht.*