Nachklausurtutorium Theoretische Informatik

Karl Jochen Micheel & Christopher Happe

Sommersemester 2019

Überblick

1. Grundlagen
2. Reguläre Sprachen
3. Kontextfreie Sprachen
4. Kontextsensitive und L0 Sprachen
5. Berechenbarkeit
6. Primitive und Partielle Rekursion

Inhaltsverzeichnis

- Primitiv rekursive Funktionen
- Allgemein und partiell rekursive Funktionen
- Ackermann Funktion
- Gödelisierungen

Primitive Rekursion

1. Induktionsbasis
   a) Alle konstanten Funktionen
      (wie z.B. $c \in \{0, 1, 2, \ldots\}$, $c: \mathbb{N} \rightarrow \mathbb{N}$ mit $c(n) = c$
   b) Die Nachfolgerfunktion $s: \mathbb{N} \rightarrow \mathbb{N}$ mit $s(n) = n + 1$
   c) Alle Identitäten $id_k^m: \mathbb{N}^m \rightarrow \mathbb{N}$, $m \geq 0$, $k \leq m$ mit
      $(n_1, n_2, \ldots, n_m) = n_k$

Primitive Rekursion

2. Induktionsschritt
   a) Substitution (Komposition von Funktionen):
      Sind f: $\mathbb{N}^k \rightarrow \mathbb{N}$ und $g_1, g_2, \dots, g_k$: $\mathbb{N}^m \rightarrow \mathbb{N}$ für $k,m \in \mathbb{N}$ primitiv rekursiv, dann ist es auch
      h: $\mathbb{N}^m \rightarrow \mathbb{N}$ mit
      $h(n_1, \dots, n_m) = f(g_1(n_1, \dots, n_m), \dots, g_k(n_1, \dots, n_m))$.

Primitive Rekursion

2. Induktionsschritt
   b) Primitive Rekursion:
   Sind $g: \mathbb{N}^m \to \mathbb{N}$ und $h: \mathbb{N}^{m+2} \to \mathbb{N}$ primitiv rekursiv, dann ist es auch $f: \mathbb{N}^{m+1} \to \mathbb{N}$ mit
   $f(0, x_1, \ldots, x_m) = g(x_1, \ldots, x_m)$
   $f(n+1, x_1, \ldots, x_m) = h(n, f(n, x_1, \ldots, x_m), x_1, \ldots, x_m)$

Beispiel

- Addition:

  $add(0,x) = id_1^1(x)$

  $add(n+1, x) = s(id_2^3(n, add(n, x), x)$

  $= s(add(n, x))$

- Multiplikation:

  $mult(0, x) = 0$

  $mult(n+1, x) = add\,(id_2^3(n, mult(n, x), x), id_3^3(n, mult(n, x), x))$

  $= add(mult(n, x), x)$

Übung

- Exponentialfunktion: exp: $\mathbb{N}^2 \rightarrow \mathbb{N}$ mit $\exp(y, x) = x^y$

Übung

- Exponentialfunktion: exp: $\mathbb{N}^2 \rightarrow \mathbb{N}$ mit exp(y, x) = x^y

  $exp(0, x) = 1$

  $exp(n+1, x) = mult(id_3^3(n, exp(n, x), x), id_2^3(n, exp(n, x), x))$

  $= mult(x, exp(n, x))$

- Fakultätsfunktion: fak: $\mathbb{N} \rightarrow \mathbb{N}$ mit fak(x) = x!

Übung

- Exponentialfunktion: exp: $\mathbb{N}^2 \rightarrow \mathbb{N}$ mit exp(y, x) = x^y

  $exp(0, x) = 1$

  $exp(n+1, x) = mult(id_3^3(n, exp(n, x), x), id_2^3(n, exp(n, x), x))$

  $= mult(x, exp(n, x))$

- Fakultätsfunktion: fa: $\mathbb{N} \rightarrow \mathbb{N}$ mit fa(x) = x!

  $fa(0) = 1$

  $fa(n+1) = mul(id_2^2\ (n, fa(n)), s(id_1^2\ (n, fa(n)))$

  $= mul(fa(n), n+1)$

Weitere nützliche Funktionen

- Vorgängerfunktion V(x) = $\begin{cases} 0 & , falls\ x = 0 \\ x - 1, & falls\ x \geq 1 \end{cases}$
- modifizierte Differenz md(x, y) = $\begin{cases} 0 & , falls\ x < y \\ x - y, & falls\ x \geq y \end{cases}$

Weitere nützliche Funktionen

- Abstand $A(x, y) = \begin{cases} x - y, falls\ y \leq x \\ y - x, falls\ x < y \end{cases}$

- Signumfunktion $S(x) = \begin{cases} 0, falls\ x = 0 \\ 1, falls\ x \ \geq 1 \end{cases}$

Der μ-Operator

- Sinn: Partielle Funktionen ermöglichen
- Sei f: $\mathbb{N}^{k+1} \rightarrow \mathbb{N}$ eine primitiv rekursive Funktion
- Dann ist g: $\mathbb{N}^k \rightarrow \mathbb{N}$ definiert als
  $g(x_1, \ldots, x_k) = \mu f(x_1, \ldots, x_k)$
  $= \min\left\{ n \in \mathbb{N} \,\middle|\, \begin{matrix} f(n, x_1, \ldots, x_k) = 0 \; und \; für \; alle \\ m < n \; ist \; f(m, x_1, \ldots, x_k) \; definiert \end{matrix} \right\}$
- Dabei ist min ∅ nicht definiert

Beispiel

- Ganzzahlige Division: g: $\mathbb{N}^2 \rightarrow \mathbb{N}$ mit

  g(y, z) = $\begin{cases} \frac{z}{y}, falls\ y\ \neq 0\ und\ y\ teilt\ z \\ undefiniert, sonst \end{cases}$

- Zu zeigen: g ist partiell rekursiv
- Idee: Definiere eine primitive rekursive Funktion f mit μf = g

Beispiel

- f(x, y, z) = |x * y - z|$^{S(y)}$ = $\begin{cases} |x \ * y - z|, falls\ y \ \neq 0 \\ 1, sonst \end{cases}$
- Wenn y = 0 ist, oder y kein Teiler von z ist, dann gilt
  μf(y,z) = min$\left\{ n \in \mathbb{N} \middle| \begin{array}{c} f(n, y, z) = 0\ und\ für\ alle \\ m < n\ ist\ f(m, y, z)\ definiert \end{array} \right\}$
  = min ∅
  also undefiniert, da f(x, y, z) für alle x nie 0 wird.
- Ansonsten ist μf(y, z) = $\frac{z}{y}$, da f total ist und ein x mit x * y = z existiert.

Übung

- g: ℕ → ℕ mit
  $g(m) = \begin{cases} \sqrt{m}, falls\ \sqrt{m} \in \mathbb{N} \\ undefiniert\ sonst \end{cases}$ ist partiell rekursiv

Übung

- g: $\mathbb{N} \rightarrow \mathbb{N}$ mit
  $g(m) = \begin{cases} \sqrt{m}, falls\ \sqrt{m} \in \mathbb{N} \\ undefiniert\ sonst \end{cases}$ ist partiell rekursiv
- f: $\mathbb{N}^2 \rightarrow \mathbb{N}$ mit $f(n, m) = |n^2 - m|$ ist partiell rekursiv
- $\mu f = g$
- Wenn $\sqrt{m} \notin \mathbb{N}$ dann ist f nie 0 und μf undefiniert
- Wenn $\sqrt{m} \in \mathbb{N}$ ist $f(\sqrt{m}, m) = 0$ und $\mu f = \sqrt{m}$

Hauptsatz der Berechenbarkeitstheorie

- Die folgenden Klassen sind äquivalent:
  - Turing-Berechenbarkeit
  - WHILE-Berechenbarkeit
  - GOTO-Berechenbarkeit
  - partiell rekursive Funktionen ($\mathbb{P}$)
- Weiterhin sind folgende Klassen äquivalent:
  - LOOP-Berechenbarkeit
  - primitiv rekursive Funktionen ($\mathbb{P}r$)

Äquivalenzen

GOTO-berechenbar =

WHILE-berechenbar =

Turing-berechenbar =

partiell rekursiv

LOOP-berechenbar =

primitiv rekursiv

Die Ackermann-Funktion

- Eine totale Funktion, die nicht Loop-berechenbar ist
- $\alpha : \mathbb{N}^2 \rightarrow \mathbb{N}$ ist definiert durch:
- $\alpha(m, n) = \begin{cases} n + 1, falls\ m = 0\ und\ n \geq 0 \\ \alpha(m-1, 1), falls\ m > 0\ und\ n = 0 \\ \alpha\big(m-1, < \alpha(m, n-1)\big), falls\ m, n > 0 \end{cases}$

Die Ackermann-Funktion

- Es gilt:
  1. $\alpha(0, n) = n + 1$
  2. $\alpha(1, n) = n + 2$
  3. $\alpha(2, n) = 2 * n + 3$
  4. $\alpha(3, n) = 2^{n+3} * -3$

Gödelisierung

- Idee:
  - Ordne jeder Funktion eine Zahl zu, die diese eindeutig identifiziert
- Vorgehen:
  - Stelle alle Funktionen über einem Alphabet dar
  - Bringe diese in eine lexikographische Ordnung
  - Zähle die einzelnen Elemente ab

Gödelisierung von $\mathbb{P}$r

- Alphabet Σ = {x, |, (,), [,], ,, , ;, $*$, s, 0, id, SUB, PR}
- Variablen x_1, x_2, … werden durch x|, x||, … repräsentiert
- Trennsymbole [](),; $*$
- Basisfunktionen s und 0
- Identitäten id_2^3 warden als id ||| $*$||dargestellt

Gödelisierung von $\mathbb{P}r$

- Substitutionen:
  - Werden $g_1, \ldots, g_k$ in f substituiert, dann wird h dargestellt als: $G(h) = SUB[G(f);G(g_1);\ldots;G(g_k)](G(x_1), \ldots, G(x_m))$.
- Primitive Rekursion:
  - Ergibt sich f aus g und h durch primitive Rekursion, so gilt $G(f) = PR[G(g), G(h)](G(x_1), \ldots, G(x_{m+1}))$

Beispiel

- sub: $\mathbb{N}^2 \rightarrow \mathbb{N}$
- sub(0, x) = id_1^1(x) = x
- sub(n+1, x) = V(id_2^3(n, sub(n,x), x)) = V(sub(n, x))
- G(sub) = PR[id|$*$|, SUB[G(V), id|||$*$|||](x|,x||,x|||)](x|, x||)
- Dabei ist G(V) das Gödelwort der Vorgängerfunktion, welches hier nicht konkret angegeben wird

Übung

- Geben Sie das Gödelwort zu exp(x,y) an.

Übung

- Geben Sie das Gödelwort zu exp(x,y) an.
- G(exp) = PR[s(0), SUB[G(mult); id|||∗||; id |||*|||] (x|, x||,x|||] (x|, x||)

Fragen?