

Informatik IV

Theoretische Informatik

Kapitel 10

Entscheidbarkeit und Aufzählbarkeit

Sommersemester 2019

Dozent: Prof. Dr. J. Rothe



Vorbereitungen

- Es sei eine Gödelisierung $\varphi_0, \varphi_1, \varphi_2, \dots$ der Klasse \mathbb{P} fixiert.
- Diese erhält man etwa durch eine Erweiterung der Gödelisierung $\psi_0, \psi_1, \psi_2, \dots$ der Klasse \mathbb{P}_r um den μ -Operator.
- Dazu äquivalent kann man eine Gödelisierung

$$M_0, M_1, M_2, \dots$$

aller Turingmaschinen angeben, wobei M_i die Funktion φ_i berechnet.

- Für jedes $k \geq 0$ bezeichne

$$\varphi_0^{(k)}, \varphi_1^{(k)}, \varphi_2^{(k)}, \dots$$

eine Gödelisierung aller k -stelligen Funktionen in \mathbb{P} .

Aufzählbarkeitssatz; Satz von der universellen Funktion

Theorem

Es gibt eine zweistellige Funktion $u \in \mathbb{P}$ (die so genannte „universelle Funktion“), so dass für alle $i, x \in \mathbb{N}$: $u(i, x) = \varphi_i^{(1)}(x)$.

Beweis: Setze $u(i, x) = \varphi_i^{(1)}(x)$. Ist u berechenbar?

Ja, nämlich mit dem folgenden Algorithmus: Bei Eingabe von i und x

- 1 berechne das Programm (d.h. die Turingmaschine M_i) von φ_i ,
- 2 wende es auf die Eingabe x an und
- 3 gib den Funktionswert $\varphi_i^{(1)}(x) = u(i, x)$ aus.

Es folgt, dass $u \in \mathbb{P}$. □

Iterationssatz; *s-m-n*-Theorem

Theorem

Für alle $m, n \in \mathbb{N}$ gibt es eine $(m+1)$ -stellige Funktion $s \in \mathbb{R}$, so dass für alle $i, x_1, \dots, x_n, y_1, \dots, y_m \in \mathbb{N}$:

$$\varphi_i^{(m+n)}(x_1, \dots, x_n, y_1, \dots, y_m) = \varphi_{s(i, y_1, \dots, y_m)}^{(n)}(x_1, \dots, x_n).$$

(Hierbei fasst man die x_i als echte Variablen und die y_j als fixierte Parameter auf.)

Beweis: Seien $i \in \mathbb{N}$ und $y_1, \dots, y_m \in \mathbb{N}$ gegeben. Betrachte

$$f(x_1, \dots, x_n) = \varphi_i^{(m+n)}(x_1, \dots, x_n, y_1, \dots, y_m)$$

als Funktion von x_1, \dots, x_n . Dann gibt es eine TM M , die f berechnet (und in deren Programm die Werte i und y_1, \dots, y_m hart codiert sind).

Iterationssatz; s - m - n -Theorem

M arbeitet bei Eingabe x_1, \dots, x_n so:

- 1 berechne das Programm (d.h. die Turingmaschine M_i) von φ_i ,
- 2 simuliere $M_i(x_1, \dots, x_n, y_1, \dots, y_m)$.

M hat natürlich das Ergebnis $\varphi_i^{(m+n)}(x_1, \dots, x_n, y_1, \dots, y_m)$.

Ist zum Beispiel $f(x_1, \dots, x_n)$ nicht definiert, so hält M nie an. Es folgt $f \in \mathbb{P}$.

M hat selbst eine Gödelnummer, sagen wir j , die von i und y_1, \dots, y_m abhängt.

Für gegebenes i und y_1, \dots, y_m kann diese Nummer j algorithmisch bestimmt werden.

Iterationssatz; s - m - n -Theorem

Setzen wir

$$s(i, y_1, \dots, y_m) = j,$$

so gibt es also ein algorithmisches Verfahren zur Berechnung von s , d.h., $s \in \mathbb{R}$, und es gilt:

$$f \equiv \varphi_j^{(n)} \equiv \varphi_{s(i, y_1, \dots, y_m)}^{(n)}.$$

Der Satz ist bewiesen. □

Kleenescher Fixpunktsatz

Theorem

Ist $h \in \mathbb{R}$, so existiert ein Fixpunkt $a \in \mathbb{N}$ mit

$$(\forall x \in \mathbb{N}) [\varphi_{h(a)}(x) = \varphi_a(x)].$$

Beweis: Sei $h \in \mathbb{R}$.

- Da h somit auch in \mathbb{P} ist, gibt es eine Gödelnummer für h , etwa

$$h = \varphi_i.$$

- Wir wenden den Iterationssatz auf zweistellige Funktionen in \mathbb{P} an. Demnach existiert eine Funktion $\sigma \in \mathbb{R}$ mit

$$\varphi_k(x, y) = \varphi_{\sigma(k, x)}(y). \tag{1}$$

Kleenescher Fixpunktsatz: Der Trick

Kleenescher Fixpunktsatz: Der Trick

Der Trick?

Welcher Trick?

Ein **Zaubertrick**?



Kleenescher Fixpunktsatz: Der Trick

Der Trick ist der folgende:

$$\varphi_{h(\sigma(x,x))}(y) = v(i, x, y) \quad \text{mit } v \in \mathbb{P} \text{ und } v = \varphi_m$$

$$= \varphi_m(i, x, y)$$

$$= \varphi_{g(m,i)}(x, y) \quad \text{mit } g \in \mathbb{R} \text{ nach dem Iterationssatz}$$

$$= \varphi_{s(i)}(x, y) \quad \text{denn } m \text{ ist ja fest, } s \in \mathbb{R}$$

$$= \varphi_{\sigma(s(i),x)}(y) \quad \text{nach (1) mit } k = s(i).$$

Kleenescher Fixpunktsatz

Also existieren Funktionen $s, \sigma \in \mathbb{R}$, so dass für alle $x \in \mathbb{N}$ gilt:

$$\varphi_{h(\sigma(x,x))} \equiv \varphi_{\sigma(s(i),x)}. \quad (2)$$

Setze

$$a = \sigma(s(i), s(i)).$$

Da $s, \sigma \in \mathbb{R}$, existiert dieser Fixpunkt a stets.

Aus (2) folgt mit $x = s(i)$:

$$\varphi_{h(a)} \equiv \varphi_{h(\sigma(s(i),s(i)))} \equiv \varphi_{\sigma(s(i),s(i))} \equiv \varphi_a.$$

Der Satz ist bewiesen. □

Entscheidbarkeit

Definition

Es sei $A \subseteq \Sigma^*$ eine Menge (analog für $A \subseteq \mathbb{N}$). Die *charakteristische Funktion von A* ist definiert durch:

$$\chi_A(x) = \begin{cases} 1 & \text{falls } x \in A \\ 0 & \text{falls } x \notin A. \end{cases}$$

A heißt *entscheidbar*, falls $\chi_A : \Sigma^* \rightarrow \{0, 1\}$ berechenbar ist.

REC bezeichne die *Klasse aller entscheidbaren Mengen*.

Bemerkung: Das heißt, eine Sprache A ist entscheidbar, falls es eine DTM gibt, die für jedes $x \in \Sigma^*$ entscheidet, ob $x \in A$ oder $x \notin A$.

Semi-Entscheidbarkeit

Definition

Es sei $A \subseteq \Sigma^*$ eine Menge (analog für $A \subseteq \mathbb{N}$). Die *partielle charakteristische Funktion von A* ist definiert durch:

$$\chi'_A(x) = \begin{cases} 1 & \text{falls } x \in A \\ \text{nicht definiert} & \text{falls } x \notin A \end{cases}$$

A heißt *semi-entscheidbar*, falls χ'_A berechenbar ist.

Semi-Entscheidbarkeit

Bemerkung: Das heißt, eine Sprache A ist semi-entscheidbar, falls es eine DTM M gibt, die A akzeptiert, d.h., $L(M) = A$.

- Für $x \in A$ stoppt die Maschine nach endlich vielen Schritten in einem Endzustand.
- Für $x \notin A$ braucht die Maschine nicht zu stoppen – jedenfalls nicht in einem Endzustand.
- Hat die Maschine noch nicht gestoppt, so ist unklar, ob die Maschine noch stoppen wird ($x \in A$) oder nicht ($x \notin A$).

Entscheidbarkeit und Semi-Entscheidbarkeit

Beispiel: Die folgenden Mengen sind entscheidbar (und damit natürlich auch semi-entscheidbar), da sich leicht Algorithmen zur Berechnung ihrer charakteristischen Funktion angeben lassen:

- 1 die Menge der Quadratzahlen: $A_1 = \{n^2 \mid n \in \mathbb{N}\} \in \text{REC}$,
- 2 die Menge der Zweierpotenzen: $A_2 = \{2^n \mid n \in \mathbb{N}\} \in \text{REC}$,
- 3 die Menge der Primzahlen: $A_3 = \{p \mid p \text{ Primzahl}\} \in \text{REC}$.

Entscheidbarkeit und Semi-Entscheidbarkeit

Theorem

A ist entscheidbar \iff A und $\bar{A} = \Sigma^ - A$ sind semi-entscheidbar.*

Beweis:

- (\Rightarrow) Eine Turingmaschine, die A entscheidet, kann leicht zu einer Turingmaschine modifiziert werden, die A bzw. \bar{A} akzeptiert.
- (\Leftarrow) Nach Voraussetzung gibt es zwei Turingmaschinen M_A und $M_{\bar{A}}$, die die Sprachen A bzw. \bar{A} akzeptieren.

Entscheidbarkeit und Semi-Entscheidbarkeit

Diese beiden Maschinen können wie folgt zu einer Maschine kombiniert werden, die für jedes $x \in \Sigma^*$ entscheidet, ob

- $x \in A$ oder
- $x \notin A$.

INPUT(x);

FOR $i = 1, 2, 3, \dots$ DO

 IF M_A hält bei Eingabe von x nach i Schritten THEN OUTPUT(1);

 IF $M_{\bar{A}}$ hält bei Eingabe von x nach i Schritten THEN OUTPUT(0);

END



Abschlusseigenschaften von REC

Theorem

REC ist abgeschlossen unter

- 1 *Schnitt,*
- 2 *Vereinigung*
- 3 *Komplement,*
- 4 *Konkatenation und*
- 5 *Iteration.*

Abschlusseigenschaften von REC

Beweis:

1 Schnitt:

$$\chi_{A \cap B}(x) = \min\{\chi_A(x), \chi_B(x)\} = \chi_A(x) \cdot \chi_B(x).$$

2 Vereinigung:

$$\chi_{A \cup B}(x) = \max\{\chi_A(x), \chi_B(x)\}.$$

3 Komplement:

$$\chi_{\bar{A}}(x) = 1 - \chi_A(x).$$

4 Konkatenation:

$$\chi_{AB}(x) = \max_{x \in \Sigma^*, x = x_1 x_2} \chi_A(x_1) \cdot \chi_B(x_2).$$

5 Iteration:

$$\chi_{A^n}(x) = \max_{x \in \Sigma^*, x = x_1 \dots x_n} \chi_A(x_1) \cdot \dots \cdot \chi_A(x_n).$$

Wiederholung: Chomsky-Hierarchie

Definition

- Eine Sprache $A \subseteq \Sigma^*$ ist genau dann vom Typ $i \in \{0, 1, 2, 3\}$, wenn es eine Typ- i -Grammatik G gibt mit $L(G) = A$.
- Die *Chomsky-Hierarchie* besteht aus den vier Sprachklassen:

$$\mathcal{L}_i = \{L(G) \mid G \text{ ist Typ-}i\text{-Grammatik}\},$$

wobei $i \in \{0, 1, 2, 3\}$. Übliche Bezeichnungen:

- \mathcal{L}_0 ist die Klasse aller Sprachen, die durch eine Grammatik erzeugt werden können;
- $\mathcal{L}_1 = \text{CS}$ ist die *Klasse der kontextsensitiven Sprachen*;
- $\mathcal{L}_2 = \text{CF}$ ist die *Klasse der kontextfreien Sprachen*;
- $\mathcal{L}_3 = \text{REG}$ ist die *Klasse der regulären Sprachen*.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Theorem

$$\text{CS} \subset \text{REC} \subset \mathfrak{L}_0.$$

Beweis:

$$\text{REC} \subseteq \mathfrak{L}_0.$$

Es sei L entscheidbar. Dann gibt es eine Turingmaschine M , die χ_L berechnet und somit entscheidet, ob $x \in L$ oder $x \notin L$ gilt.

Also ist

$$L \in \{L(M) \mid M \text{ ist eine Turingmaschine}\} = \mathfrak{L}_0.$$

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

$CS \subseteq REC$.

Sei $L \in CS$, und sei $G = (\Sigma, N, S, P)$ eine Grammatik für L mit nur nichtverkürzenden Regeln. Sei $x \in \Sigma^*$ ein gegebenes Wort mit $|x| = n$.

Idee:

- Die „Zwischenergebnisse“ x_i in einer beliebigen Ableitung

$$S \vdash_G x_1 \vdash_G x_2 \vdash_G \cdots \vdash_G x_k = x$$

haben alle die Länge $|x_i| \leq n$.

- Da es in $(\Sigma \cup N)^*$ nur endlich viele Wörter der Länge $\leq n$ gibt, kann man durch systematisches Durchprobieren entscheiden, ob $x \in L$ oder $x \notin L$ gilt, d.h., L ist entscheidbar.
- (Somit ist das Wortproblem für Typ-1 Sprachen entscheidbar.)

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Formal: Wir geben einen Algorithmus an, der diese Entscheidung trifft. Dieser kann z.B. durch eine TM oder sonstwie implementiert werden.

Definiere für $m, n \in \mathbb{N}$ die Mengen

$$T_m^n = \{w \in (\Sigma \cup N)^* \mid |w| \leq n \text{ und } S \vdash_G^m w\}.$$

Diese lassen sich, für festes $n \geq 1$, wie folgt induktiv über m definieren:

$$\begin{aligned} T_0^n &= \{S\} \\ T_{m+1}^n &= \text{Abl}^n(T_m^n), \end{aligned}$$

wobei für ein beliebiges X der Hüllenoperator Abl^n definiert ist durch

$$\text{Abl}^n(X) = X \cup \left\{ w \in (\Sigma \cup N)^* \mid \begin{array}{l} |w| \leq n \text{ und es existiert ein} \\ v \in X \text{ mit } v \vdash_G w \end{array} \right\}.$$

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Dies:

$$\text{Abl}^n(X) = X \cup \left\{ w \in (\Sigma \cup N)^* \mid \begin{array}{l} |w| \leq n \text{ und es existiert ein} \\ v \in X \text{ mit } v \vdash_G w \end{array} \right\}.$$

ist nur für Typ-1-Grammatiken korrekt. Bei Typ-0-Grammatiken könnte ein w mit $|w| \leq n$ aus v mit $|v| > n$ ableitbar sein.)

Der folgende Algorithmus liefert die Entscheidung des Wortproblems für Typ-1-Grammatiken. Offenbar läuft dieser Algorithmus in Exponentialzeit.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Algorithmus-Typ-1(G, x) {

// G ist Typ-1-Grammatik und $x \in \Sigma^*$ ein Wort mit $|x| = n$

$T := \{S\}; T_1 := \emptyset;$

while ($x \notin T$ und $T \neq T_1$) { $T_1 := T; T := \text{Abl}^n(T_1);$ }

if ($x \in T$) return „ $x \in L$ “

else return „ $x \notin L$ “ }

Abbildung: Algorithmus zur Entscheidung des Typ-1-Wortproblems

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

- Da es in $(\Sigma \cup N)^*$ nur endlich viele Wörter der Länge $\leq n$ gibt, folgt für jedes n , dass $\bigcup_{m \geq 0} T_m^n$ eine endliche Menge ist (nämlich mit $2^{c \cdot n}$ Elementen für ein von G abhängiges c).

- Folglich existiert ein m_0 mit

$$T_{m_0}^n = T_{m_0+1}^n = T_{m_0+2}^n = \dots = \bigcup_{m \geq 0} T_m^n.$$

- Ist $x \in L$, so ist $x \in \bigcup_{m \geq 0} T_m^n = T_{m_0}^n$.
- Ist jedoch $x \notin L$, so ist $x \notin T_{m_0}^n$.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

CS \neq REC.

Sei $\Sigma = \{a, b\}$. Wie definieren eine Sprache $L \subseteq \Sigma^*$ mit

- $L \in \text{REC}$, aber
- $L \notin \text{CS}$.

Dazu brauchen wir eine

Gödelisierung aller Typ-1-Grammatiken mit $\Sigma = \{a, b\}$:

- Die Nichtterminale seien durchnummeriert:

$$X_0, X_1, X_2, \dots,$$

wobei das Nichtterminal X_i dargestellt werde als $X \underbrace{|\dots|}_{i\text{-mal}}$.

- X_0 sei das Startsymbol.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

- Regeln der Form

$$p_1 \rightarrow q_1, \quad p_2 \rightarrow q_2, \quad \dots, \quad p_n \rightarrow q_n$$

können durch das Wort

$$p_1 \rightarrow q_1 \# p_2 \rightarrow q_2 \# \dots \# p_n \rightarrow q_n$$

über dem Alphabet $\Gamma = \{X, |, a, b, \rightarrow, \#\}$ dargestellt werden.

- Nun codieren wir Wörter, die solche Typ-1-Grammatiken beschreiben, durch den Homomorphismus $\varphi : \Gamma^* \rightarrow \Sigma^*$:

$$\varphi(\lambda) = \lambda$$

$$\varphi(a) = bab$$

$$\varphi(b) = ba^2b$$

$$\varphi(\rightarrow) = ba^3b$$

$$\varphi(\#) = ba^4b$$

$$\varphi(\underbrace{X || \dots ||}_{i\text{-mal}}) = ba^{5+i}b.$$

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

- Zur Erinnerung: Ein Homomorphismus ist eine Abbildung $h : \Gamma^* \rightarrow \Sigma^*$ mit

$$h(xy) = h(x)h(y),$$

$$h(\lambda) = \lambda.$$

Sind etwa die Regeln der Grammatik gegeben durch

$$X_0 \rightarrow \lambda, \quad X_0 \rightarrow X_1 X_2, \quad X_2 \rightarrow a, \quad X_1 X_2 \rightarrow b X_2,$$

so wird sie codiert durch das Wort

$$\begin{aligned} &ba^5 bba^3 bba^4 bba^5 bba^3 bba^6 bba^7 bba^4 bba^7 b \\ &ba^3 bbabba^4 bba^6 bba^7 bba^3 bba^2 bba^7 b. \end{aligned}$$

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

- Eine Ordnung auf Σ (z.B. $a < b$) induziert eine quasilexikographische Ordnung w_0, w_1, w_2, \dots auf Σ^* , wobei w_i mit $i \in \mathbb{N}$ das i -te Wort ist:

Σ^*	λ	a	b	aa	ab	ba	bb	aaa	\dots
i -tes Wort	w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	\dots

- Entfernen wir nun alle Wörter, die keine syntaktisch korrekte Typ-1-Grammatik codieren, so erhalten wir die gesuchte Gödelisierung von CS: G_0, G_1, G_2, \dots

Es ist dabei möglich, dass $G_i = G_j$ für $i \neq j$, da eine Permutation der Regeln verschiedene Wörter w_i und w_j induziert.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Wie bei jeder Gödelisierung gilt:

- Jeder Typ-1-Grammatik G entspricht ein Wort $w_G \in \Sigma^*$.
- Für jedes Wort $w \in \Sigma^*$ können wir algorithmisch entscheiden,
 - ob es eine (syntaktisch korrekte) Typ-1-Grammatik codiert,
 - und wenn ja, welche.

Nun definieren wir die Sprache $L \subseteq \Sigma^*$ durch

$$L = \{w_i \mid i \in \mathbb{N} \text{ und } w_i \notin L(G_i)\}.$$

Beispielsweise ist $\lambda \notin L$, denn $w_0 = \lambda$, aber G_0 ist gegeben durch die eine Regel $X_0 \rightarrow \lambda$, so dass $\lambda \in L(G_0)$ gilt.

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Dass $L \in \text{REC}$ gilt, folgt unmittelbar aus der oben gezeigten Inklusion $\text{CS} \subseteq \text{REC}$. Für gegebenes $x \in \Sigma^*$:

- berechne das i mit $x = w_i$;
- berechne die Grammatik G_i durch den Aufbau der Gödelisierung bis zur Nr. i ;
- entscheide mit dem Algorithmus aus der obigen Abbildung, ob $x = w_i \notin L(G_i)$.

Um zu zeigen, dass $L \notin \text{CS}$, nehmen wir für einen Widerspruch an, dass $L \in \text{CS}$. Dann existiert ein $j \in \mathbb{N}$ mit $L = L(G_j)$. Daraus folgt

$$w_j \in L \iff w_j \notin L(G_j) = L,$$

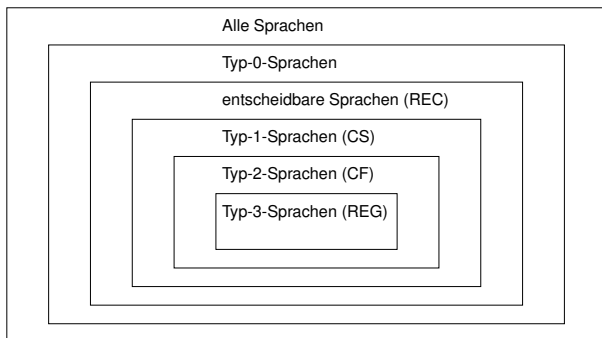
ein Widerspruch.

$\text{REC} \neq \mathcal{L}_0$. Diese Aussage zeigen wir später. □

Wie ordnet sich REC in die Chomsky-Hierarchie ein?

Aus dem letzten Satz folgt insbesondere die Echtheit der letzten (hier noch nicht betrachteten) Inklusion aus dem Fakt:

$$\text{REG} \subseteq \text{CF} \subseteq \text{CS} \subseteq \mathfrak{L}_0.$$



Rekursive Aufzählbarkeit

Definition

- Eine Menge A heißt *rekursiv aufzählbar* (kurz: *A ist r.e.*, nach dem englischen *recursively enumerable*), falls
 - entweder $A = \emptyset$
 - oder $A = W_f$ für ein $f \in \mathbb{R}$.

Dabei bezeichnet W_f den Wertebereich von f , und f heißt *Aufzählfunktion von A* .

- RE bezeichne die *Klasse aller rekursiv aufzählbaren Mengen*.

Rekursive Aufzählbarkeit

Bemerkung:

- Eine Aufzählfunktion f für eine Menge $A \in \text{RE}$ schöpft also den ganzen Wertebereich A aus, d.h.,

$$A = \{f(i) \mid i \in \mathbb{N}\}.$$

- Rekursive Aufzählbarkeit darf nicht verwechselt werden mit dem Begriff der Abzählbarkeit. Diese verlangt keine effektive (algorithmische) Machbarkeit.
- Umgekehrt verlangt die rekursive Aufzählbarkeit keine 1-1-Zuordnung. Es ist möglich, dass $f(i) = f(j)$ für $i \neq j$.

Rekursive Aufzählbarkeit

Bemerkung:

- Jede endliche Menge ist rekursiv aufzählbar.

Eine Aufzählfunktion von $A = \{a_0, \dots, a_n\}$ ist gegeben durch

$$f(i) = \begin{cases} a_i & \text{falls } 0 \leq i \leq n \\ a_n & \text{falls } i > n. \end{cases}$$

- Jede Teilmenge einer abzählbaren Menge ist abzählbar.
- Jedoch ist *nicht* jede Teilmenge einer rekursiv aufzählbaren Menge auch rekursiv aufzählbar.

REC versus RE

Theorem

$$\text{REC} \subseteq \text{RE}.$$

Beweis: Sei $A \in \text{REC}$. Ist $A = \emptyset$, so ist $A \in \text{RE}$ bereits gezeigt.

Ist $A \neq \emptyset$, so wählen wir ein festes $a \in A$ und konstruieren die Aufzählfunktion f für A so:

$$f(i) = \begin{cases} i & \text{falls } \chi_A(i) = 1 \text{ (d.h., } i \in A) \\ a & \text{sonst.} \end{cases}$$

Offenbar ist f total und berechenbar:

Da $A \in \text{REC}$, ist $\chi_A \in \mathbb{R}$ und somit auch $f \in \mathbb{R}$.

REC versus RE

Es gilt

$$A = W_f,$$

denn:

- **$A \subseteq W_f$:**

Ist $j \in A$, so ist $\chi_A(j) = 1$, also $f(j) = j$, woraus $j \in W_f$ folgt.

- **$W_f \subseteq A$:**

Ist $j \in W_f$, so ist

- entweder $\chi_A(j) = 1$, also $j \in A$,
- oder $\chi_A(j) = 0$, also $j = a \in A$.

Der Satz ist bewiesen. □

REC versus RE

Theorem

$$A \in \text{REC} \iff (A \in \text{RE} \wedge \bar{A} \in \text{RE}).$$

Beweis: (\Rightarrow)

- Sei $A \in \text{REC}$.
- Nach dem vorigen Satz ist $A \in \text{RE}$.
- Da REC unter Komplement abgeschlossen ist, ist $\bar{A} \in \text{REC}$.
- Wieder nach dem vorigen Satz ist $\bar{A} \in \text{RE}$.

REC versus RE

(\Leftarrow) Seien A und \bar{A} in RE.

- Ist $A = \emptyset$ oder $\bar{A} = \emptyset$ (d.h., $A = \Sigma^*$), so ist $A \in \text{REC}$.
- Sei also $\emptyset \neq A \neq \Sigma^*$, und seien $f, g \in \mathbb{R}$ Aufzählfunktionen mit

$$A = W_f \quad \text{und} \quad \bar{A} = W_g.$$

Der folgende Algorithmus berechnet χ_A bei Eingabe x :

- Berechne

$$f(0), g(0), f(1), g(1), f(2), \dots$$

solange, bis $x = f(i)$ oder $x = g(i)$ für ein i gilt.

(Alle diese Berechnungen terminieren wegen $f, g \in \mathbb{R}$.)

- Gilt $x = f(i)$ für ein i , so gib 1 aus;
- gilt $x = g(i)$ für ein i , so gib 0 aus.

REC versus RE

- Da
 - entweder $x \in A = W_f$
 - oder $x \in \bar{A} = W_g$,

kann sich x nicht beliebig lange vor dieser Entscheidung drücken, sondern muss sich irgendwann „outen“.

- Folglich terminiert diese Prozedur in endlicher Zeit, und es gilt

$$\chi_A \in \mathbb{R}.$$

- Es folgt $A \in \text{REC}$. □

Abschlusseigenschaften von RE

Theorem

RE *ist abgeschlossen unter Schnitt und Vereinigung.* **ohne Beweis**

Bemerkung: Wir werden später sehen, dass RE *nicht* komplementabgeschlossen ist.

Charakterisierungen von RE

Theorem

$A \in \text{RE} \iff A$ ist semi-entscheidbar.

ohne Beweis

Theorem

Sei $\varphi_0, \varphi_1, \varphi_2, \dots$ eine fixierte Gödelisierung der Klasse \mathbb{P} , und $D_i = D_{\varphi_i}$ bzw. $W_i = W_{\varphi_i}$ bezeichne den Definitions- bzw. den Wertebereich der i -ten Funktion $\varphi_i \in \mathbb{P}$. Dann gilt:

$$\text{RE} = \{D_i \mid i \in \mathbb{N}\} = \{W_i \mid i \in \mathbb{N}\}.$$

Charakterisierungen von RE

Beweis: 1. Sei $A \in \text{RE}$.

Wir zeigen $A = D_i$ und $A = W_j$ für geeignete $i, j \in \mathbb{N}$.

Da $A \in \text{RE}$, gilt $\chi'_A \in \mathbb{IP}$ nach dem vorigen Satz. Genauer:

Fall 1: $A = \emptyset$. Dann ist χ'_A die nirgends definierte Funktion, die natürlich in \mathbb{IP} liegt.

Fall 2: $A \neq \emptyset$. Dann gibt es ein $f \in \mathbb{R}$ mit $W_f = A$. Der folgende Algorithmus berechnet χ'_A bei Eingabe x :

- Berechne $f(0), f(1), f(2), \dots$ (also die Aufzählung von A) solange, bis $x = f(i)$ für ein i gilt.
(Alle diese Berechnungen terminieren wegen $f \in \mathbb{R}$.)
- Gilt $x = f(i)$ für ein i , so gib 1 aus.

Falls x nie vorkommt, weil $x \notin A$, so terminiert der obige Algorithmus nie.

Charakterisierungen von RE

Da $\chi'_A \in \mathbb{P}$, existiert eine Nr. i mit $\chi'_A = \varphi_i$. Somit ist gezeigt:

$$A = D_{\chi'_A} = D_i.$$

Definiere

$$g(x) = x \cdot \chi'_A(x).$$

Es gilt:

- $g(x) = x$, falls $x \in A$, und
- $g(x)$ ist nicht definiert, falls $x \notin A$.

Da $\chi'_A \in \mathbb{P}$, ist auch $g \in \mathbb{P}$, und es gibt ein j mit $\varphi_j = g$.

Somit gilt

$$A = W_g = W_j.$$

Charakterisierungen von RE

Was haben wir denn jetzt gezeigt??



Charakterisierungen von RE

Wir haben also gezeigt, dass

$$A \in \{D_i \mid i \in \mathbb{N}\} \quad \text{und} \quad A \in \{W_i \mid i \in \mathbb{N}\}$$

gilt. Somit:

$$\text{RE} \subseteq \{D_i \mid i \in \mathbb{N}\} \quad \text{und} \quad \text{RE} \subseteq \{W_i \mid i \in \mathbb{N}\}.$$

Charakterisierungen von RE

Ach so.



Charakterisierungen von RE

2. Wir zeigen $D_i \in \text{RE}$ für jedes $i \in \mathbb{N}$;
der Fall $W_i \in \text{RE}$ wird analog bewiesen.

Ist $D_i = \emptyset$, so gilt trivialerweise $D_i \in \text{RE}$.

Sei also $D_i \neq \emptyset$. Wähle ein festes $a \in D_i$.

Definiere eine injektive Paarungsfunktion $\pi : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ mit den folgenden Eigenschaften:

- π ist berechenbar.
- Die Umkehrfunktionen $\pi_1, \pi_2 : \mathbb{N} \rightarrow \mathbb{N}$, die für $\pi(x, y) = n$ definiert sind durch $\pi_1(n) = x$ und $\pi_2(n) = y$, sind ebenfalls berechenbar.
- $W_\pi \in \text{REC}$.

Charakterisierungen von RE

Ein solches π kann so definiert werden:

$$\pi(x, y) = 2^{x+y} + x.$$

Anschaulich bedeutet dies:

x	0	1	2	3	4	...
y						
0	$2^0 = 1$	3	6	11	20	...
1	$2^1 = 2$	5	10	19	\ddots	\ddots
2	$2^2 = 4$	9	18	\ddots	\ddots	\ddots
3	$2^3 = 8$	17	\ddots	\ddots	\ddots	\ddots
4	$2^4 = 16$	\ddots	\ddots	\ddots	\ddots	\ddots
\vdots	\vdots	\ddots	\ddots	\ddots	\ddots	\ddots

Charakterisierungen von RE

Klar ist, dass $\pi \in \mathbb{R}$ und $W_\pi \in \text{REC}$, denn es gilt

$$n \in W_\pi \iff 2^k + k \geq n,$$

wobei k die größte Zahl in \mathbb{N} mit $2^k \leq n$ ist.

Wir zeigen $\pi_1, \pi_2 \in \mathbb{P}$:

- Für $n \in W_\pi$ sei $\pi(x, y) = n$.
- Bestimme das größte $k \in \mathbb{N}$ mit $2^k \leq n$ und berechne

$$x = n - 2^k \quad \text{und} \quad y = k - x.$$

Charakterisierungen von RE

Wir suchen nun ein $f \in \mathbb{R}$ mit $W_f = D_i$. Dieses f werde durch den folgenden Algorithmus bei Eingabe $n \in \mathbb{N}$ berechnet:

- Berechne $x = \pi_1(n)$ und $y = \pi_2(n)$, falls $n \in W_\pi$.
Andernfalls setze $x = y = 0$.
- Berechne aus der fixierten Nr. i das Programm der i -ten TM M_i in der fixierten Gödelisierung von \mathbb{P} .
- Simuliere die Berechnung von M_i auf Eingabe x für y Takte.
- Terminiert die Simulation, so gib x aus, sonst das fest gewählte Element $a \in D_i$.

Charakterisierungen von RE

Da der Algorithmus nur Elemente von D_i ausgibt, gilt

$$W_f \subseteq D_i.$$

Umgekehrt gilt auch

$$D_i \subseteq W_f,$$

denn wenn $j \in D_i$, so ist $\varphi_i(j)$ definiert und $M_i(j)$ hält nach t Takten an, für ein geeignetes t .

Setze $n = \pi(j, t)$. Dann ist $f(n) = j$ für ein geeignetes n , also $j \in W_f$.

Somit gilt $W_f = D_i$ für $f \in \mathbb{R}$.

Es folgt $D_i \in \text{RE}$.



Charakterisierungen von RE

Theorem

$A \in \mathcal{L}_0 \iff A$ ist semi-entscheidbar.

ohne Beweis

Folgerung: Die folgenden Aussagen sind paarweise äquivalent.

- 1 $A \in \text{RE}$.
- 2 A ist semi-entscheidbar.
- 3 A ist vom Typ 0, d.h., $A \in \mathcal{L}_0$.
- 4 $\chi'_A \in \mathbb{P}$.
- 5 $A = L(M)$ für eine deterministische TM M .
- 6 $A = L(M)$ für eine nichtdeterministische TM M .
- 7 $A = D_f$ für ein $f \in \mathbb{P}$.
- 8 $A = W_f$ für ein $f \in \mathbb{P}$.

ohne Beweis

Das Halteproblem



Das Halteproblem

- Wir wissen: $REC \subseteq RE$.
- Um zu zeigen, dass diese Inklusion echt ist, definieren wir nun das **Halteproblem**.
- Bei diesem Problem kommen Turingmaschinen als Eingaben vor.
- Dazu erläutern wir zunächst, wie man Turingmaschinen

$$M = (\Sigma, \Gamma, Z, \delta, z_0, \square, F)$$

als ein Wort über dem Alphabet $\{0, 1\}$ schreiben kann (Gödelisierung).

Das Halteproblem: Gödelisierung von TMs

Wir nummerieren zunächst die Elemente aus Z und Γ , also

$$Z = \{z_0, z_1, z_2, \dots, z_n\}$$

$$\Gamma = \{a_0, a_1, a_2, \dots, a_k\}.$$

Jeder δ -Regel

$$\delta(z_i, a_j) = (z_{i'}, a_{j'}, r)$$

ordnen wir ein Wort $w_{i,j,i',j',r}$ zu, wobei

$$w_{i,j,i',j',r} = \#\#\text{bin}(i)\#\text{bin}(j)\#\text{bin}(i')\#\text{bin}(j')\#\text{bin}(r)$$

$$r = \begin{cases} 0 & \text{falls } r = L \\ 1 & \text{falls } r = R \\ 2 & \text{falls } r = N. \end{cases}$$

Das Halteproblem: Gödelisierung von TMs

Für alle Regeln aus δ schreiben wir diese Wörter in beliebiger Reihenfolge hintereinander und erhalten so eine Codierung von M über dem Alphabet $\{0, 1, \#\}$.

Um M über dem Alphabet $\{0, 1\}$ zu codieren, nehmen wir die folgende Ersetzung vor:

0 \mapsto 00

1 \mapsto 01

\mapsto 11

Das so erhaltene Wort zu Turingmaschine M bezeichnen wir mit

$\text{code}(M)$.

Das Halteproblem: Gödelisierung von TMs

Beispiel: Wir geben eine Codierung der Turingmaschine

$$M = (\{0, 1\}, \{0, 1, \square\}, \{z_0, z_1, z_2, z_e\}, \delta, z_0, \square, \{z_e\})$$

für die Nachfolgerfunktion $f : \mathbb{N} \rightarrow \mathbb{N}$ mit

$$f : n \rightarrow n + 1$$

aus einem früheren Beispiel an:

$(z_0, 0) \mapsto (z_0, 0, R)$	$(z_1, 0) \mapsto (z_2, 1, L)$	$(z_2, 0) \mapsto (z_2, 0, L)$
$(z_0, 1) \mapsto (z_0, 1, R)$	$(z_1, 1) \mapsto (z_1, 0, L)$	$(z_2, 1) \mapsto (z_2, 1, L)$
$(z_0, \square) \mapsto (z_1, \square, L)$	$(z_1, \square) \mapsto (z_e, 1, N)$	$(z_2, \square) \mapsto (z_e, \square, R)$

Tabelle: Liste δ der Turingbefehle von M für die Funktion $f(n) = n + 1$

Das Halteproblem: Gödelisierung von TMs

Nummerierung der Zustände

z_0	z_1	z_2	z_e
z_0	z_1	z_2	z_3

und des Arbeitsalphabets

0	1	\square
a_0	a_1	a_2

Das Halteproblem: Gödelisierung von TMs

ergibt die folgende Codierung der Funktion δ :

$\delta(z_i, a_j) = (z_{i'}, a_{j'}, r)$	$w_{i,j,i',j',r}$
$\delta(z_0, 0) = (z_0, 0, R)$	##0#0#0#0#1
$\delta(z_0, 1) = (z_0, 1, R)$	##0#1#0#1#1
$\delta(z_0, \square) = (z_1, \square, L)$	##0#10#1#10#0
$\delta(z_1, 0) = (z_2, 1, L)$	##1#0#10#1#0
$\delta(z_1, 1) = (z_1, 0, L)$	##1#1#1#0#0
$\delta(z_1, \square) = (z_e, 1, N)$	##1#10#11#1#10
$\delta(z_2, 0) = (z_2, 0, L)$	##10#0#10#0#0
$\delta(z_2, 1) = (z_2, 1, L)$	##10#1#10#1#0
$\delta(z_2, \square) = (z_e, \square, R)$	##10#10#11#10#1

Das Halteproblem: Gödelisierung von TMs

Dies ergibt die Codierung:

##0#0#0#0#1##0#1#0#1#1##0#10#1#10#0
 ##1#0#10#1#0##1#1#1#0#0##1#10#11#1#10
 ##10#0#10#0#0##10#1#10#1#0##10#10#11#10#1
 von M über dem Alphabet $\{0, 1, \#\}$.

Mit der Ersetzung

0 \mapsto 00

1 \mapsto 01

\mapsto 11

erhalten wir $\text{code}(M) = 1111001100110011001101 \dots$

Das Halteproblem: Gödelisierung von TMs

- Offensichtlich ist nicht jedes Wort über dem Alphabet $\{0, 1\}$ eine so definierte Codierung einer Turingmaschine.
- Um die Umkehrabbildung der obigen Codierung anzugeben, sei M_0 eine beliebige feste Turingmaschine.

Dann ist für jedes Wort $w \in \{0, 1\}^*$ eine Turingmaschine M_w definiert durch:

$$w \in \{0, 1\}^* \mapsto M_w = \begin{cases} M & \text{falls } w = \text{code}(M) \\ M_0 & \text{sonst} \end{cases}$$

Das spezielle Halteproblem

Definition

Die Sprache

$$\begin{aligned} K &= \{w \in \{0, 1\}^* \mid M_w(w) \text{ hält nach endlich vielen Schritten}\} \\ &= \{i \in \mathbb{N} \mid i \in D_i\} \end{aligned}$$

wird als das *spezielle Halteproblem* bezeichnet.

Theorem

Das spezielle Halteproblem ist rekursiv aufzählbar, aber nicht entscheidbar, d.h.,

$$K \in \text{RE} \quad \text{und} \quad K \notin \text{REC}.$$

Das spezielle Halteproblem

Beweis: Der Beweis von $K \in \text{RE}$ ist analog zum Beweis von

$$\text{RE} = \{\mathbf{D}_i \mid i \in \mathbb{N}\} = \{\mathbf{W}_i \mid i \in \mathbb{N}\}.$$

Um zu zeigen, dass $K \notin \text{REC}$, nehmen wir für einen Widerspruch $K \in \text{REC}$ an.

Dann ist die charakteristische Funktion χ_K berechenbar mittels einer Turingmaschine M .

Das spezielle Halteproblem

Wir modifizieren M wie folgt zu einer TM M' :

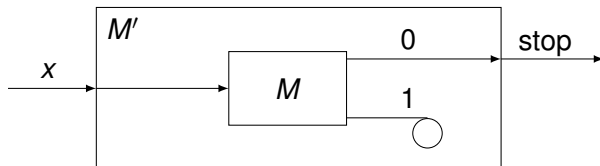


Abbildung: Zum Beweis von $K \notin \text{REC}$

- M' hält, falls M eine 0 ausgibt, und
- M' geht in eine Endlosschleife, falls M eine 1 ausgibt, siehe die obige Abbildung.

Das spezielle Halteproblem

Es sei nun $w' \in \{0, 1\}^*$ mit $M_{w'} = M'$, d.h., w' sei das Codewort der Turingmaschine M' . Dann gilt:

M' angesetzt auf w' hält

$\Leftrightarrow M$ angesetzt auf w' gibt 0 aus (nach Def. von M')

$\Leftrightarrow \chi_K(w') = 0$ (nach Def. von M)

$\Leftrightarrow w' \notin K$ (nach Def. von χ_K)

$\Leftrightarrow M_{w'}$ angesetzt auf w' hält nicht (nach Def. von K)

$\Leftrightarrow M'$ angesetzt auf w' hält nicht (nach Def. von $M_{w'}$).

Das spezielle Halteproblem

Damit haben wir die Aussage

M' angesetzt auf w' hält $\Leftrightarrow M'$ angesetzt auf w' hält nicht

hergeleitet, die offenbar einen Widerspruch darstellt.

Damit ist die Annahme, dass K entscheidbar ist, falsch. □

Folgerung: REC ist echt in RE enthalten. Insgesamt haben wir damit gezeigt, dass

$$\text{REG} \subset \text{DCF} \subset \text{CF} \subset \text{CS} \subset \text{REC} \subset \text{RE}$$

und die Chomsky-Hierarchie echt ist:

$$\mathfrak{L}_3 \subset \mathfrak{L}_2 \subset \mathfrak{L}_1 \subset \mathfrak{L}_0.$$

Das spezielle Halteproblem

Folgerung: RE ist nicht komplementabgeschlossen.

Beweis: Nach dem obigen Satz ist $K \in \text{RE}$, aber $\bar{K} \notin \text{RE}$, denn sonst wäre $K \in \text{REC}$.

Behauptung: Es gibt Funktionen in \mathbb{P} , die nicht zu Funktionen in \mathbb{R} fortgesetzt werden können. **ohne Beweis**

Der Projektionssatz

Definition

Seien $A \subseteq \mathbb{N}$ und $B \subseteq \mathbb{N} \times \mathbb{N}$ Mengen.

A ist *Projektion von B* , falls für alle $x \in \mathbb{N}$ gilt:

$$x \in A \iff (\exists y \in \mathbb{N}) [(x, y) \in B].$$

Theorem (Projektionssatz)

$A \in \text{RE} \iff A$ ist Projektion einer Menge $B \in \text{REC}$.

Beweis des Projektionssatzes

Beweis: (\Rightarrow) Sei $A \in \text{RE}$. Wegen

$$\text{RE} = \{D_i \mid i \in \mathbb{N}\} = \{W_i \mid i \in \mathbb{N}\}.$$

gibt es ein $i \in \mathbb{N}$ mit $A = D_i$. Daraus folgt:

$$x \in A \iff x \in D_i$$

$$\iff \varphi_i(x) \text{ ist definiert, d.h., } (\exists y) [\varphi_i(x) = y]$$

$$\iff M_i(x) \text{ hält, d.h., } (\exists y) [M_i(x) \text{ hält mit Ausgabe } y]$$

Beweis des Projektionssatzes



Zwischenfrage: Ist „ $M_i(x)$ hält mit Ausgabe y “ in

$$(\exists y) [M_i(x) \text{ hält mit Ausgabe } y]$$

denn schon ein entscheidbares Prädikat?

Beweis des Projektionssatzes



Nein?

Ach so.

Und wie weiter?

Beweis des Projektionssatzes

(\Rightarrow) Sei $A \in \text{RE}$.

Wegen

$$\text{RE} = \{D_i \mid i \in \mathbb{N}\} = \{W_i \mid i \in \mathbb{N}\}.$$

gibt es ein $i \in \mathbb{N}$ mit $A = D_i$.

Daraus folgt:

$$\begin{aligned} x \in A &\iff x \in D_i \\ &\iff \varphi_i(x) \text{ ist definiert, d.h., } (\exists y) [\varphi_i(x) = y] \\ &\iff M_i(x) \text{ hält, d.h., } (\exists y) [M_i(x) \text{ hält mit Ausgabe } y] \\ &\iff (\exists y) (\exists t) [M_i(x) \text{ hält nach } t \text{ Takten mit Ausgabe } y] \\ &\iff (\exists z) [z = \pi(y, t) \text{ und } T_i(x, z) \text{ ist erfüllt}] \\ &\iff (\exists z) [(x, z) \in B_i]. \end{aligned}$$

Beweis des Projektionssatzes

Dabei ist das so genannte Turingprädikat $T_i(x, z)$ genau dann erfüllt, wenn $M_i(x)$ nach t Takten mit Ausgabe y hält, wobei $z = \pi(y, t)$.

Die Menge B_i ist für festes i so definiert:

$$B_i = \{(x, z) \mid z = \pi(y, t) \text{ und } T_i(x, z) \text{ ist erfüllt}\}.$$

Noch zu zeigen ist $B_i \in \text{REC}$.

Dies leistet der folgende Algorithmus bei Eingabe (x, z) :

- Berechne $y = \pi_1(z)$ und $t = \pi_2(z)$; falls $z \notin W_\pi$, so setze $y = t = 0$;
- berechne aus der bekannten Nr. i das Programm der i -ten TM M_i ;
- simuliere $M_i(x)$ für $\leq t$ Takte und teste, ob sie mit Ausgabe y anhält;
- falls ja, gib 1 aus; andernfalls gib 0 aus.

Beweis des Projektionssatzes

(\Leftarrow) Sei A Projektion einer Menge $B \in \text{REC}$. Das heißt, für alle $x \in \mathbb{N}$ gilt:

$$x \in A \iff (\exists y \in \mathbb{N}) [(x, y) \in B].$$

Wir suchen eine Aufzählfunktion $f \in \mathbb{R}$ für A , d.h.,

$$W_f = A.$$

Man muss von jedem x argwöhnen, dass es in A liegt.

Ist das der Fall, so wird dies durch ein

$$y \in \mathbb{N} \quad \text{mit} \quad (x, y) \in B$$

bezeugt. Die Entscheidung, ob $(x, y) \in B$ oder $(x, y) \notin B$, lässt sich mit dem Algorithmus für B treffen, das ja entscheidbar ist.

Beweis des Projektionssatzes

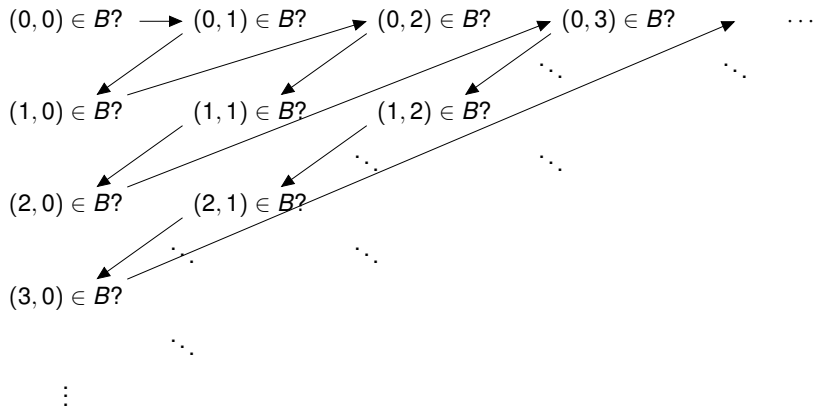
Wie findet man für jedes $x \in A$ einen Zeugen?



Beweis des Projektionssatzes

Und wie findet man für jedes $x \in A$ einen Zeugen?

Die folgende Abbildung:



Beweis des Projektionssatzes

... zeigt, in welcher Reihenfolge die rekursive Aufzählfunktion f durch alle Paare geht und jeweils den Entscheidungsalgorithmus für B simuliert.

- Sagt dieser „ja“ für ein Paar (x, y) , so gibt f das Argument x aus,
- andernfalls einen fest gewählten Lückenbüßer $a \in A$. Dieser existiert unter der Annahme, dass $A \neq \emptyset$.

Somit ist $A \in \text{RE}$ via $f \in \text{R}$.

Ist $A = \emptyset$, so ist A nach Definition sowieso in RE. □

Anwendung des Projektionssatzes

Nachweis der rekursiven Aufzählbarkeit, z.B.:

① K ist in RE:

$$\begin{aligned}i \in K &\iff \varphi_i(i) \text{ ist definiert} \\ &\iff (\exists t) [M_i(i) \text{ hält nach } t \text{ Takten}] \\ &\iff (\exists t) [(i, t) \in B],\end{aligned}$$

wobei die Menge $B \in \text{REC}$ so definiert ist:

$$B = \{(i, t) \mid M_i(i) \text{ hält nach } t \text{ Takten}\}.$$

Anwendung des Projektionssatzes

2 $X = \{i \in \mathbb{N} \mid D_i \neq \emptyset\}$ ist in RE:

$$\begin{aligned}
 i \in X &\iff D_i \neq \emptyset \\
 &\iff (\exists j) [j \in D_i] \\
 &\iff (\exists j) [\varphi_i(j) \text{ ist definiert}] \\
 &\iff (\exists j) (\exists t) [M_i(j) \text{ hält nach } t \text{ Takten}] \\
 &\iff (\exists z) [(i, z) \in B],
 \end{aligned}$$

wobei die Menge $B \in \text{REC}$ so definiert ist:

$$B = \{(i, z) \mid z = \pi(j, t) \text{ und } M_i(j) \text{ hält nach } t \text{ Takten}\}.$$

Anwendung des Projektionssatzes

③ $Y = \{i \in \mathbb{N} \mid 17 \in W_i\}$ ist ebenso in RE.

④ ...

REC und RE in der Chomsky-Hierarchie

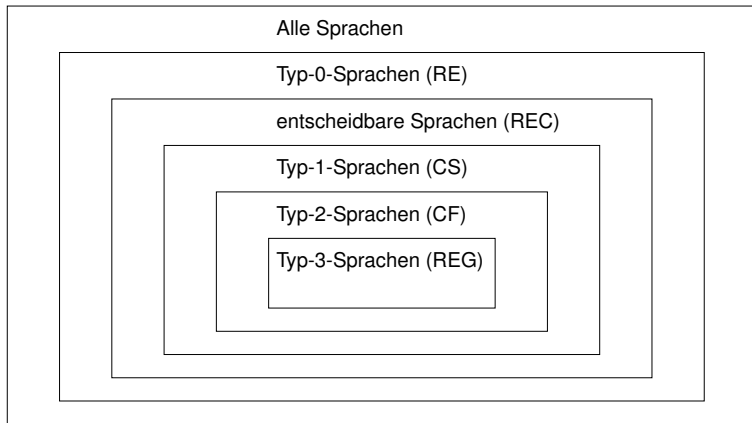


Abbildung: Einordnung von REC und RE in die Chomsky-Hierarchie