Definition 136

Die Klasse der μ-rekursiven Funktionen ist die kleinste Klasse von (nicht notwendigerweise totalen) Funktionen, die die Basisfunktionen (konstante Funktionen, Nachfolgerfunktion, Projektionen) enthält und alle Funktionen, die man hieraus durch (evtl. wiederholte) Anwendung von Komposition, primitiver Rekursion und/oder des μ-Operators gewinnen kann.

Satz 137

f μ-rekursiv $\Longleftrightarrow$ f WHILE-berechenbar.

Beweis:

Der Beweis ist elementar. □

2. Entscheidbarkeit, Halteproblem

Wir wollen nun zeigen, dass es keinen Algorithmus geben kann, der als Eingabe ein (beliebiges) Programm P und Daten x für P erhält und (für jedes solches Paar (P, x)!) entscheidet, ob P hält, wenn es mit Eingabe x gestartet wird.

Wir erinnern uns:

1. Eine Sprache A ist rekursiv gdw die charakteristische Funktion χ_A berechenbar ist.
2. Eine Sprache A ist rekursiv aufzählbar (r.a.) gdw die semi-charakteristische Funktion χ'_A berechenbar ist.

2.1 Rekursive Aufzählbarkeit

Definition 138

Eine Sprache $A \subseteq \Sigma^*$ heißt rekursiv auflistbar, falls es eine berechenbare Funktion $f : \mathbb{N}_0 \to \Sigma^*$ gibt, so dass

$$A = \{f(0), f(1), f(2), \ldots\}\,.$$

Bemerkung: Es ist nicht verlangt, dass die Auflistung in einer gewissen Reihenfolge (z.B. lexikalisch) erfolgt!

Beispiel 139

Σ^* (mit $\Sigma = \{0, 1\}$) ist rekursiv auflistbar. Wir betrachten dazu etwa folgende Funktion:

alle nullstelligen Wörter $f(0) = \epsilon$

alle einstelligen Wörter
$$\begin{cases} f(1) = 0 \\ f(2) = 1 \end{cases}$$

alle zweistelligen Wörter
$$\begin{cases} f(3) = 00 \\ f(4) = 01 \\ f(5) = 10 \\ f(6) = 11 \end{cases}$$

alle dreistelligen Wörter
$$\begin{cases} f(7) = 000 \\ \vdots \end{cases}$$

Beispiel 139

Eine weitere Möglichkeit, eine Funktion f anzugeben, die alle Wörter $\in \{0,1\}^*$ auflistet, ist:

$$f(n) = \text{Binärkodierung von } n+1 \text{ ohne die führende } 1$$

Also:

$$\begin{aligned} f(0) &= 1\epsilon \\ f(1) &= 10 \\ f(2) &= 11 \\ f(3) &= 100 \\ \vdots &\qquad \vdots \end{aligned}$$

Beispiel 139

$L_{TM} = \{w \in \{0,1\}^*;\ w$ ist Codierung einer TM$\}$ ist rekursiv auflistbar:

Wir listen $\{0,1\}^*$ rekursiv auf, prüfen jedes erzeugte Wort, ob es eine syntaktisch korrekte Codierung einer Turing-Maschine ist, und verwerfen es, falls nicht.

Wir wählen stattdessen die kanonische Auflistung von $\{0,1\}^*$ und ersetzen jedes dabei erzeugte Wort, das keine korrekte Codierung darstellt, durch den Code einer Standard-TM, die $\emptyset$ akzeptiert.

Satz 140

Eine Sprache A ist genau dann rekursiv auflistbar, wenn sie rekursiv aufzählbar (semi-entscheidbar) ist.

Beweis:

Wir zeigen zunächst „$\Rightarrow$“.

Sei $f : \mathbb{N}_0 \to \Sigma^*$ eine berechenbare Funktion, die A auflistet.

Betrachte folgenden Algorithmus:

lies die Eingabe $w \in \Sigma^*$
$x := 0$
while true do
  if $w = f(x)$ **then return** („ja“); **halt fi**
  $x := x + 1$
od

Beweis:

Wir zeigen nun „$\Leftarrow$“.

Sei P eine WHILE-Programm, das die semi-charakteristische Funktion χ'_A berechnet, und sei f eine berechenbare Funktion, die Σ^* auflistet.

Betrachte folgenden Algorithmus:

lies die Eingabe $n \in \mathbb{N}_0$
$count := -1$; $k := -1$
repeat
  $k := k + 1$
  $w := f(c_1(k))$; $m := c_2(k)$
  if P hält bei Eingabe w in genau m Schritten **then**
    $count := count + 1$
until $count = n$
return w

Hier sind c_1 und c_2 die Umkehrfunktionen einer Paarfunktion. □

2.2 Halteproblem

Definition 141

Unter dem speziellen Halteproblem H_s versteht man die folgende Sprache:

$$H_s = \{w \in \{0,1\}^*;\ M_w \text{ angesetzt auf } w \text{ hält}\}$$

Hierbei ist $(M_\epsilon, M_0, M_1, \ldots)$ eine berechenbare Auflistung der Turing-Maschinen.

Wir definieren weiter

Definition 142

$$L_d = \{w \in \Sigma^*;\ M_w \text{ akzeptiert } w \text{ nicht}\}$$

Satz 143

L_d ist nicht rekursiv aufzählbar.

Beweis:

Wäre L_d r.a., dann gäbe es ein w, so dass $L_d = L(M_w)$.
Dann gilt:

$$\begin{aligned} M_w \text{ akzeptiert } w \text{ nicht} &\Leftrightarrow w \in L_d \\ &\Leftrightarrow w \in L(M_w) \\ &\Leftrightarrow M_w \text{ akzeptiert } w \end{aligned}$$

„$\Longrightarrow$“ Widerspruch! □

Korollar 144

L_d ist nicht entscheidbar.

Satz 145

H_s ist nicht entscheidbar.

Beweis:

Angenommen, es gäbe eine Turing-Maschine M, die H_s entscheidet. Indem man i.W. die Antworten von M umdreht, erhält man eine TM, die L_d entscheidet. Widerspruch! □

2.3 Unentscheidbarkeit

Definition 146

Unter dem (allgemeinen) Halteproblem H versteht man die Sprache

$$H = \{\langle x, w\rangle \in \{0,1\}^*;\ M_x \text{ angesetzt auf } w \text{ hält}\}$$

Satz 147

Das Halteproblem H ist nicht entscheidbar.

Beweis:

Eine TM, die H entscheidet, könnten wir benutzen, um eine TM zu konstruieren, die H_s entscheidet. □

Bemerkung: H und H_s sind beide rekursiv aufzählbar!

Definition 148

Seien A, $B \subseteq \Sigma^*$. Dann heißt A (effektiv) reduzierbar auf B gdw $\exists f : \Sigma^* \to \Sigma^*$, f total und berechenbar mit

$$(\forall w \in \Sigma^*)[w \in A \Leftrightarrow f(w) \in B]\,.$$

Wir schreiben auch

$$A \hookrightarrow_f B \text{ bzw. } A \hookrightarrow B\,.$$

bzw. manchmal

$$A \leq B \text{ oder auch } A \preceq_f B\,.$$

Ist A mittels f auf B reduzierbar, so gilt insbesondere

$$f(A) \subseteq B \text{ und } f(\bar{A}) \subseteq \bar{B}\,.$$

Satz 149

Sei $A \hookrightarrow_f B$.

(i) B *rekursiv* $\Rightarrow$ A *rekursiv.*

(ii) B *rekursiv aufzählbar* $\Rightarrow$ A *rekursiv aufzählbar.*

Beweis:

(i) $\chi_A = \chi_B \circ f$.

(ii) $\chi'_A = \chi'_B \circ f$.

□

Definition 150

Das Halteproblem auf leerem Band H_0 ist

$$H_0 = \{w \in \{0,1\}^*;\ M_w \text{ hält auf leerem Band}\}\,.$$

Satz 151

H_0 ist unentscheidbar (nicht rekursiv).

Beweis:

Betrachte die Abbildung f, die definiert ist durch:

$$\{0,1\}^* \ni w \mapsto f(w),$$

$f(w)$ ist die Gödelnummer einer TM, die, auf leerem Band angesetzt, zunächst $c_2(w)$ auf das Band schreibt und sich dann wie $M_{c_1(w)}$ (angesetzt auf $c_2(w)$) verhält. Falls das Band nicht leer ist, ist es unerheblich, wie sich $M_{f(w)}$ verhält.

f ist total und berechenbar.

Es gilt: $w \in H \Leftrightarrow M_{c_1(w)}$ angesetzt auf $c_2(w)$ hält
$\Leftrightarrow M_{f(w)}$ hält auf leerem Band
$\Leftrightarrow f(w) \in H_0$

also $H \hookrightarrow_f H_0$ und damit H_0 unentscheidbar. □

Bemerkung

Es gibt also keine allgemeine algorithmische Methode, um zu entscheiden, ob ein Programm anhält.

Satz 152 (Rice)

Sei $\mathcal{R}$ die Menge aller (TM)-berechenbaren Funktionen und $\mathcal{S}$ eine nichttriviale Teilmenge von $\mathcal{R}$ (also $\mathcal{S} \neq \mathcal{R}$, $\mathcal{S} \neq \emptyset$). Dann ist

$$G(\mathcal{S}) := \{w \in \{0,1\}^*; \quad \textit{die von } M_w \textit{ berechnete Funktion ist in } \mathcal{S}\}$$

unentscheidbar.

Beweis:
Sei Ω die total undefinierte Funktion.

1. Fall: $\Omega \in \mathcal{S}$

Sei $q \in \mathcal{R} - \mathcal{S}$ (es gibt ein derartiges q, da $\mathcal{S}$ nichttrivial), Q eine TM für q.
Zu $w \in \{0,1\}^*$ sei $f(w) \in \{0,1\}^*$ Gödelnummer einer TM, für die gilt:

1. bei Eingabe x ignoriert $M_{f(w)}$ diese zunächst und verhält sich wie M_w auf leerem Band.
2. wenn obige Rechnung hält, dann verhält sich $M_{f(w)}$ wie Q auf der Eingabe x.

f ist total und berechenbar.

$w \in H_0 \Leftrightarrow M_w$ hält auf leerem Band
$\Leftrightarrow M_{f(w)}$ berechnet die Funktion q $(\neq \Omega)$
$\Leftrightarrow$ die von $M_{f(w)}$-berechnete Funktion ist nicht in $\mathcal{S}$
$\Leftrightarrow f(w) \notin G(\mathcal{S})$

Beweis:

Sei Ω die total undefinierte Funktion.

2. Fall: $\Omega \notin \mathcal{S}$

Seien $q \in \mathcal{S}, Q, f$ wie im Fall 1.

$$\begin{aligned} w \in H_0 \quad &\Leftrightarrow M_{f(w)} \text{ berechnet die Funktion } q \ (\neq \Omega) \\ &\Leftrightarrow \text{die von } M_{f(w)} \text{ berechnete Funktion ist in } \mathcal{S} \\ &\Leftrightarrow f(w) \in G(\mathcal{S}) \end{aligned}$$

Also: $H_0 \hookrightarrow_f G(\mathcal{S})$.

H_0 unentscheidbar $\Rightarrow G(\mathcal{S})$ unentscheidbar.

□

Wir zeigen hier nur diesen Satz. Setzt man weitere Eigenschaften von $\mathcal{S}$ voraus, kann man sogar zeigen, dass $G(\mathcal{S})$ nicht einmal rekursiv aufzählbar ist.