
10. Der Äquivalenzsatz

In diesem Abschnitt zeigen wir, dass die von uns betrachteten verschiedenen Formalisierungen des Berechenbarkeitsbegriffs äquivalent sind, d.h. alle zu derselben Klasse (partiell) berechenbarer Funktionen über \mathbb{N} führen.

10.1 SATZ. (Äquivalenzsatz) Für eine (partielle) Funktion $f^{(n)} : \mathbb{N}^n \rightarrow \mathbb{N}$ sind folgende Aussagen äquivalent

- (i) f ist (partiell) Turing-berechenbar.
- (ii) f wird von einer k -Band-Turingmaschine berechnet ($k \geq 1$).
- (iii) f wird von einem Turingoperator (konservativ) berechnet.
- (iv) f wird von einer Registermaschine berechnet.
- (v) f wird von einem Registeroperator (konservativ) berechnet.
- (vi) f ist (partiell) rekursiv.

D.h. es gilt

$$\begin{aligned} F(\text{TM}) &= F(k\text{-TM}) = \bigcup_{k \geq 1} F(k\text{-TM}) = F(\text{TO}) = F_{\text{kon}}(\text{TO}) \\ &= F(\text{RM}) = F(\text{RO}) = F_{\text{kon}}(\text{RO}) = F(\text{REK}). \end{aligned}$$

In den vorangegangenen Abschnitten haben wir bereits die Inklusionen

$$\begin{aligned} F(\text{REK}) \subseteq F_{\text{kon}}(\text{RO}) = F(\text{RO}) \subseteq F_{\text{kon}}(\text{TO}) \subseteq F(\text{TO}) \subseteq F(\text{TM}) \\ = F(k\text{-TM}) = \bigcup_{k \geq 1} F(k\text{-TM}) \end{aligned}$$

gezeigt und in den Übungen wurde

$$F(\text{RO}) \subseteq F(\text{RM}) \subseteq F(k\text{-TM})$$

bewiesen. Zum Beweis des Äquivalenzsatzes genügt es also zu zeigen:

10.2 LEMMA. $F(\text{TM}) \subseteq F(\text{REK})$.

BEWEIS. Wir werden zeigen, dass man die Arbeitsweise von Turingmaschinen rekursiv beschreiben kann. Hierzu *gödelisiert* (oder *arithmetisiert*) man die gegebenen Maschinen M , d.h. kodiert die M -Konfigurationen durch Zahlen (*Gödelnummern*), sodass Eigenschaften der Rechnungen von M in Prädikate über \mathbb{N} übergehen. Insbesondere wird das Prädikat $\text{RECHNUNG}(\vec{x}, y, z)$, das besagt, dass y die terminierende Rechnung von M bei Eingabe \vec{x} beschreibt und z die zugehörige Ausgabe ist, primitiv rekursiv sein. Die von M berechnete Funktion erhalten wir dann hieraus durch Anwendung des μ -Operators.

Sei nun $\varphi^{(n)} \in F(TM)$ gegeben und sei $M = (B, P)$ eine Turingmaschine, die φ berechnet. Es ist dann B die Basismaschine

$$B = TB(\{1\}, \{1\}, \Gamma, n),$$

wobei wir o.B.d.A.

$$\Gamma = \{0, 1, \dots, p\}$$

annehmen können, wobei 0 das Blankzeichen ist. Ohne Einschränkung hat das Programm $P = \{I_1, \dots, I_{q-1}\}$ genau einen Stoppzustand, sodass wir als Zustandsmenge $Z = \{1, \dots, q\}$ annehmen können, wobei 1 der Startzustand und q der Stoppzustand ist.

Zur Gödelisierung der Konfigurationen benutzen wir die im letzten Abschnitt eingeführte Kodierungsfunktion für endliche Zahlenfolgen. Ein Wort $w = a_1 \dots a_n$ über dem Bandalphabet Γ beschreiben wir durch die kodierte Folge $\langle a_1, \dots, a_n \rangle$. Eine Konfiguration c wird dann durch ein kodierte Tripel $\tilde{c} = \langle z(c), l(c), r(c) \rangle$ beschrieben, wobei $z(c)$ der Zustand von c ist und $l(c)$ und $r(c)$ den relevanten Teil der Inschrift auf der linken bzw. rechten Bandhälfte (jeweils vom Arbeitsfeld aus gelesen) beschreiben. Ist also c die Konfiguration

$$c = \dots a_{-l} \dots a_{-1} a_0 \dots a_r \dots$$

\uparrow
 z

wobei $l, r \geq 1$, $a_i \in \Gamma$, $z \in Z$ (und links von a_{-l} bzw. rechts von a_r nur Blanks stehen), dann ist die Gödelnummer $\tilde{c} = \langle z(c), l(c), r(c) \rangle$ von c gegeben durch

$$\begin{aligned} z(c) &= z \\ l(c) &= \langle a_{-1}, \dots, a_{-l} \rangle \\ r(c) &= \langle a_0, \dots, a_r \rangle. \end{aligned}$$

Wir zeigen nun, dass man die Arbeitsweise von M bzgl. dieser Kodierung durch primitiv rekursive Funktionen und Prädikate beschreiben kann, wobei wir zunächst die einzelnen Komponenten von M betrachten.

Eingabefunktion. Die Funktion $\text{start}^{(n)}$, die einer Eingabe $\vec{x} = (x_1, \dots, x_n) \in \mathbb{N}^n$ die Gödelnummer \tilde{c} der zugehörigen Startkonfiguration

$$c = \dots 00 \underset{\uparrow}{x_1} 0 \underset{\uparrow}{x_2} 0 \dots 0 \underset{\uparrow}{x_r} 0 \dots$$

zuordnet, ist wie folgt definiert. Wir führen zunächst die Hilfsfunktion $\text{unär}^{(1)}$ ein, die eine Zahl m auf die kodierte Folge $\langle m \rangle = \langle 1, \dots, 1 \rangle$ der Länge $m+1$ zur Beschreibung der Unärdarstellung abbildet. Diese Funktion lässt sich durch die primitive Rekursion

$$\begin{aligned} \text{unär}(0) &= \langle 1 \rangle \\ \text{unär}(m+1) &= \text{unär}(m) \circ \langle 1 \rangle \end{aligned}$$

definieren, weshalb $\text{unär} \in F(\text{PRIM})$.

Wir definieren dann

$$\text{start}(\vec{x}) = \langle 1, \text{lbhstart}(\vec{x}), \text{rbhstart}(\vec{x}) \rangle$$

explizit über die Funktionen $\text{lbhstart}(\vec{x})$ und $\text{rbhstart}(\vec{x})$, die die Inschriften der beiden Bandhälften der Startkonfiguration beschreiben. Diese wiederum sind explizit definiert über den Funktionen $\langle \rangle$, \circ und unär durch

$$\begin{aligned}\text{lbhstart}(\vec{x}) &= \langle 0 \rangle \\ \text{rbhstart}(\vec{x}) &= \langle 0 \rangle \circ \text{unär}(x_1) \circ \langle 0 \rangle \circ \dots \circ \text{unär}(x_n) \circ \langle 0 \rangle\end{aligned}$$

woraus $\text{start} \in F(\text{PRIM})$ folgt.

Einschrittfunktion. Als Nächstes beschreiben wir die einzelnen Schritte von M , d.h. definieren die Funktion $\text{nfk}^{(1)}$, die die Gödelnummer \tilde{c} einer Konfiguration c auf die Gödelnummer $\text{nfk}(\tilde{c})$ der Nachfolgekongfiguration abbildet. (Ist c Stoppkonfiguration, so setzen wir $\text{nfk}(\tilde{c}) = \tilde{c}$.) Zur Definition von nfk benutzen wir die primitiv rekursiven Funktionen

$$z(x) = (x)_1, \text{lbh}(x) = (x)_2, \text{rbh}(x) = (x)_3 \text{ und } \text{af}(x) = (x)_{3,1}$$

als Hilfsfunktionen, die den Zustand bzw. die Inschriften der linken und rechten Bandhälfte sowie des Arbeitsfeldes einer Konfiguration aus deren Gödelnummer berechnen. Die Funktion nfk spalten wir in die 3 Teile

$$\text{nfk}(x) = \langle \text{nfz}(x), \text{nflbh}(x), \text{nfrbh}(x) \rangle$$

auf, die die 3 Komponenten der Gödelnummer der Nachfolgekongfiguration beschreiben. Diese sind in Abhängigkeit vom aktuellen Zustand in Entsprechung zu der anzuwendenden Instruktion $I = I_{z(x)}$ (für $z(x) < q$) definiert. Wir beschreiben hierzu zunächst die Wirkung der Instruktion I_i bei aktueller Inschrift a des Arbeitsfeldes durch die Funktionen

$$\text{nz}(i, a) = \begin{cases} j & \text{falls } i < q \ \& \ I_i = (i, 0, j) \\ j & \text{falls } i < q \ \& \ I_i = (i, t_{a'}, j, k) \ \& \ a' \neq a \\ k & \text{falls } i < q \ \& \ I_i = (i, t_a, j, k) \\ i & \text{sonst} \end{cases}$$

$$\text{no}(i) = \begin{cases} a & \text{falls } i < q \ \& \ I_i = (i, a, j) \ \& \ a \leq p \\ p+1 & \text{falls } i < q \ \& \ I_i = (i, L, j) \\ p+2 & \text{falls } i < q \ \& \ I_i = (i, R, j) \\ p+3 & \text{sonst} \end{cases}$$

Bei der „nächste-Operation“-Funktion no kodieren dabei die Zahlen $p+1$, $p+2$, $p+3$ die Bewegungen L , R , S . Da $\text{nz}(i, a) = U_1^2(i, a)$ f.ü. und $\text{no}(i) = C_{p+3}^1(i)$ f.ü., sind diese Funktionen nach Satz 9.22 primitiv rekursiv. Hiermit können wir nun definieren

$$\text{nfz}(x) = \text{nz}(z(x), \text{af}(x))$$

$$\text{nflbh}(x) = \begin{cases} \text{lbh}(x) & \text{falls } \text{no}(z(x)) \neq p+1 \ \& \ \text{no}(z(x)) \neq p+2 \\ \text{tail}(\text{lbh}(x)) \circ \langle 0 \rangle & \text{falls } \text{no}(z(x)) = p+1 \\ \langle \text{af}(x) \rangle \circ \text{lbh}(x) & \text{falls } \text{no}(z(x)) = p+2 \end{cases}$$

$$\text{nfrbh}(x) = \begin{cases} \langle p \rangle \circ \text{tail}(\text{rbh}(x)) & \text{falls } \text{no}(z(x)) \leq p \\ \text{head}(\text{lbh}(x)) \circ \text{rbh}(x) & \text{falls } \text{no}(z(x)) = p+1 \\ \text{tail}(\text{rbh}(x)) \circ \langle 0 \rangle & \text{falls } \text{no}(z(x)) = p+2 \\ \text{rbh}(x) & \text{sonst} \end{cases}$$

(Bei der Definition von $nfbh$ erweitern wir hierbei den relevanten Teil der linken Bandhälfte links um ein Blank, falls diese durch den L -Befehl rechts um ein Feld gekürzt wird (s. Fall 2). Analog bei $nfrbh$, weshalb die relevanten Bandteile nie leer sind.) Aus den im letzten Abschnitt gezeigten Abschlusseigenschaften von $F(\text{PRIM})$ folgt, dass diese Funktionen primitiv rekursiv sind. Damit gilt auch $nfk(x) \in F(\text{PRIM})$.

Ausgabefunktion. Wir geben eine Funktion $\text{wert}^{(1)}$ an, die der Gödelnummer \tilde{c} einer Konfiguration

$$c = \dots a_{-1} \dots a_{-1} a_0 \overset{\uparrow}{m} a_{m+2} \dots a_r \dots$$

mit $a_{m+2} \neq 1$ den Ausgabewert m zuordnet. Da $m+2$ durch die Länge der relevanten Bandhälfte beschränkt ist und a_{m+2} der erste rechts des Arbeitsfeldes stehende von 1 verschiedene Buchstabe ist, lässt sich der Ausgabewert durch

$$\text{wert}(x) = \mu y \leq l(\text{rbh}(x)) (y \geq 2 \ \& \ (\text{rbh}(x))_y \neq 1) - 2$$

definieren, woraus $\text{wert} \in F(\text{PRIM})$ folgt.

M-Rechnungen. Mit Hilfe der Funktion nfk können wir das Prädikat

$$\text{TKF}(x) \Leftrightarrow x \text{ kodiert eine Folge von Gödelnummern} \\ \text{von Konfigurationen, die eine terminierende} \\ M\text{-Rechnung bilden}$$

als primitiv rekursiv nachweisen:

$$\text{TKF}(x) \Leftrightarrow l(x) \geq 1 \ \& \\ \forall i < l(x) (i \geq 1 \Rightarrow nfk((x)_i) = (x)_{i+1}) \ \& \\ z((x)_{l(x)}) = q$$

Durch Hinzunahme der die Ein- und Ausgabe beschreibenden Funktionen sehen wir damit, dass das Prädikat

$$\text{RECHNUNG}(\vec{x}, y, z) \Leftrightarrow y \text{ kodiert die terminierende Rechnung} \\ \text{von } M \text{ bei Eingabe } \vec{x} \\ \text{und diese liefert Ausgabe } z$$

ebenfalls primitiv rekursiv ist:

$$\text{RECHNUNG}(\vec{x}, y, z) \Leftrightarrow \text{TKF}(y) \ \& \\ \text{start}(\vec{x}) = (y)_1 \ \& \\ \text{wert}((y)_{l(y)}) = z$$

Resultatsfunktion. Die von M berechnete Funktion können wir nun noch mit Hilfe des μ -Operators aus dem primitiv rekursiven Rechnungsprädikat extrahieren:

$$\varphi(\vec{x}) = (\mu y (\text{RECHNUNG}(\vec{x}, (y)_1, (y)_2)))_2$$

Hiermit ist $\varphi \in F(\text{REK})$ bewiesen. □

Die im Beweis von Satz 10.1 gefundene Darstellung von φ zeigt, dass jede partiell rekursive Funktion eine Darstellung hat, in der der μ -Operator nur einmal angewendet wird. Darüberhinaus sind die anderen Bestandteile der Darstellung sogar primitiv rekursiv, d.h.

$$\varphi^{(n)}(\vec{x}) = f^{(1)}(\mu y (P^{(n+1)}(\vec{x}, y)))$$

wobei $f, P \in F(\text{PRIM})$. (Wir werden hierauf bei der sog. Kleeneschen Normalform zurückkommen.) Dies zeigt insbesondere, dass eine Turingmaschine zur Berechnung einer zahlentheoretischen Funktion $\varphi : \mathbb{N}^n \rightarrow \mathbb{N}$, deren Laufzeit primitiv rekursiv beschränkt ist, eine primitiv rekursive Funktion berechnet, da hier der μ -Operator in der Darstellung beschränkt werden kann.

10.3 KOROLLAR. Sei $f^{(n)}$ eine totale Funktion, die von einer Turingmaschine M berechnet wird, deren Laufzeit $\text{time}_M(\vec{x})$ durch eine primitiv rekursive Funktion $s(\vec{x})$ beschränkt ist. Dann ist f selbst primitiv rekursiv.

BEWEISIDEE. Nach den vorangehenden Bemerkungen genügt es zu zeigen, dass die kodierte Rechnung einer Turingmaschine primitiv rekursiv in der Eingabe und in der Rechenzeit beschränkt ist. Dies ergibt sich leicht aus der Definition der kodierten Rechnung. (Übung!) \square

Wir wollen noch zeigen, dass die primitiv rekursiven Funktionen gerade die von primitiven Registeroperatoren berechneten Funktionen sind.

10.4 SATZ. Eine Funktion $f : \mathbb{N}^n \rightarrow \mathbb{N}$ ist genau dann primitiv rekursiv, wenn sie von einem primitiven Registeroperator berechnet wird. D.h. $F(\text{PRO}) = F(\text{PRIM})$.

Da wir bereits $F(\text{PRIM}) \subseteq F(\text{PRO}) = F_{\text{kon}}(\text{PRO})$ gezeigt haben, genügt es folgendes Lemma zu beweisen.

10.5 LEMMA. $F_{\text{kon}}(\text{PRO}) \subseteq F(\text{PRIM})$.

BEWEIS. Wir zeigen zunächst, dass es zu jedem k -PRO $P : \mathbb{N}^k \rightarrow \mathbb{N}^k$ eine primitiv rekursive Funktion $\langle P \rangle : \mathbb{N} \rightarrow \mathbb{N}$ gibt, sodass

$$\forall \vec{x} \in \mathbb{N}^k (\langle P \rangle(\langle \vec{x} \rangle) = \langle P(\vec{x}) \rangle).$$

Hierbei definieren wir $\langle P \rangle$ durch Induktion nach dem Aufbau von P :

1. $P = a_i$. Dann ist

$$\langle P \rangle(x) = \langle (x)_1, \dots, (x)_{i-1}, (x)_i + 1, (x)_{i+1}, \dots, (x)_k \rangle$$

2. $P = s_i$. Dann ist $\langle P \rangle$ entsprechend mit $(x)_i - 1$ statt $(x)_i + 1$ definiert.
3. $P = P_1 P_2$. Dann ist $\langle P \rangle = \langle P_2 \rangle(\langle P_1 \rangle)$ für die nach I.V. existierenden Funktionen $\langle P_1 \rangle$ und $\langle P_2 \rangle$.
4. $P = [s_i P_1]_i$, wobei P_1 weder a_i noch s_i als Teiloperator enthält. Für die nach I.V. gegebene Funktion $\langle P_1 \rangle$ und für $x = \langle \vec{x} \rangle = \langle x_1, \dots, x_k \rangle$ gilt dann

$$(\langle P \rangle(x))_j = U_j^k(P(\vec{x})) = U_j^k(P_1^{x_i}(\vec{x})) = (\langle P_1 \rangle^{(x)_i}(x))_j$$

für $j \neq i$ und

$$(\langle P \rangle(x))_i = U_i^k(P(\vec{x})) = 0$$

Wir können also $\langle P \rangle$ mit Hilfe der primitiv rekursiven Hilfsfunktion

$$f(x) = \langle P_1 \rangle^{(x)_i}(x) = \text{Iter}(\langle P_1 \rangle)(x, (x)_i)$$

darstellen durch

$$\langle P \rangle(x) = \langle (f(x))_1, \dots, (f(x))_{i-1}, 0, (f(x))_{i+1}, \dots, (f(x))_k \rangle.$$

Damit ist $\langle P \rangle \in F(\text{PRIM})$ für jeden PRO P gezeigt.

Wir können nun $F_{kon}(\text{PRO}) \subseteq F(\text{PRIM})$ wie folgt zeigen. Sei $f^{(n)} \in F_{kon}(\text{PRO})$. Dann gibt es einen k -PRO P (mit $k \geq n + 1$), der f konservativ berechnet. Es gilt also

$$f(\vec{x}) = U_{n+1}^k(P(\vec{x}, 0^{k-n})).$$

Mit der oben definierten primitiv rekursiven Kodierung $\langle P \rangle$ von P gilt also

$$f(\vec{x}) = (\langle P \rangle(\langle \vec{x}, 0^{k-n} \rangle))_{n+1}$$

D.h. f ist über die primitiv rekursiven Funktionen $\langle P \rangle$, τ^* und π^* explizit definierbar und damit selbst primitiv rekursiv. \square

BEMERKUNG. Entsprechend können wir $F(\text{RO}) \subseteq F(\text{REK})$ direkt zeigen (was jedoch schon aus dem Äquivalenzsatz folgt). Im Beweis von Lemma 10.5 müssen wir dann zu jedem k -RO P eine passende rekursive Funktion $\langle P \rangle$ definieren. Dies geschieht wiederum induktiv nach dem Aufbau von P , wobei im 4. Fall nun $P = [P_1]_i$, wobei $\langle P_1 \rangle$ nach I.V. bereits gegeben ist. Hier ist die Anzahl der iterativen Aufführungen von P_1 nun nicht a priori beschränkt. Die Hilfsfunktion f hat hier entsprechend die Gestalt

$$f(\langle \vec{x} \rangle) = P_1^y(\vec{x}),$$

wobei y die kleinste Zahl z mit $U_i^k(P_1^z(\vec{x})) = 0$ ist. D.h.

$$f(x) = \text{Iter}(\langle P_1 \rangle)(x, g(x))$$

wobei

$$g(x) = \mu y((\text{Iter}(\langle P_1 \rangle))_i)(x).$$

Nach der Church-Turing-These und dem Äquivalenzsatz stimmen die (partiell) berechenbaren Funktionen mit den (partiell) rekursiven Funktionen überein. Da die entscheidbaren Mengen nach Satz 2.10 gerade die Mengen sind, deren charakteristische Funktion berechenbar ist, und die rekursiven Mengen analog über die Rekursivität der charakteristischen Funktion definiert werden, impliziert die Church-Turing-These auch die Äquivalenz von entscheidbaren und rekursiven Mengen. Benutzen wir die aus Satz 2.11 folgende Charakterisierung der aufzählbaren Mengen, so können wir mit Hilfe der folgenden Definition diesen Begriff ebenfalls formalisieren:

10.6 DEFINITION. Eine Menge $M \subseteq \mathbb{N}^n$ ist *rekursiv aufzählbar*, wenn M der Definitionsbereich einer partiell rekursiven Funktion $\varphi : \mathbb{N}^n \rightarrow \mathbb{N}$ ist.