5. Universelle Maschinen und uniform rekursive Klassen

Die Existenz universeller Maschinen erlaubt effektive Aufzählungen der Klasse der r.a. Mengen, der Klasse der partiell rekursiven Funktionen und der üblichen Komplexitätsklassen.

Zur Konstruktion universeller Turingmaschinen nehmen wir zunächst gewisse Normierungen an den Turingmaschinen vor und beschreiben diese normierten Turingmaschinen dann mit Hilfe von Binärwörtern (Gödelisierung).

5.1 DEFINITION. Eine Turingmaschine M ist *normiert*, wenn die Zustandsmenge $Z = \{0,\dots,m\}$ von M ein endliches Anfangsstück der natürlichen Zahlen ist, wobei 0 der Startzustand ist und $1,2$ die einzigen Stoppzustände sind. Ist M ein Akzeptor, so ist weiter 1 der einzige Endzustand von M. Das Bandalphabet $\Gamma = \{a_0,\dots,a_{p+1}\}$ beginnt mit dem Eingabealphabet $\Sigma = \{a_0,\dots,a_p\}$ (das bei einem Transduktor zugleich das Ausgabealphabet ist) und enthält zusätzlich nur das Blankzeichen a_{p+1}.

5.2 LEMMA. *Zu jeder Turingmaschine M zur Erkennung einer n-stelligen Sprache über Σ^* mit $\Sigma_2 \subseteq \Sigma$ oder zur Berechnung einer n-stelligen Funktion über Σ^* kann man effektiv eine äquivalente normierte Turingmaschine M' des gleichen Typs angeben, für die*

$$\mathrm{time}_{M'}(\vec{x}) \in O(\mathrm{time}_M(\vec{x}))$$

und

$$\mathrm{space}_{M'}(\vec{x}) \in O(\mathrm{space}_M(\vec{x}))$$

gilt.

BEWEIS. Zur Normierung der Zustände genügt es, diese geeignet umzubenennen und um die beiden Stoppzustände zu erweitern. Eine M-Stopp- oder -Endkonfiguration wird dann in einem zusätzlichen Rechenschritt in den entsprechenden M'-Stoppzustand 1 (Akzeptieren) bzw. 2 (Verwerfen) überführt. Zur Normierung des Bandalphabets ersetzt man die zu eliminierenden Buchstaben durch ihre Binärkodierung. Genauer: Ist $\Gamma = \{a_0,\dots,a_q\}$ das Bandalphabet von M, so werden auf den Arbeitsbändern von M' die Buchstaben a_i durch Bitfolgen 1^i0^{q-i} ersetzt. Dies ist für den linearen Verlust bei Platz und Zeit verantwortlich. □

Betrachten wir normierte Turingmaschinen festen Typs, so findet sich alle spezifische Information über die Maschine in ihrem Programm. Wir können also eine solche Maschine M durch ein Binärwort wie folgt eindeutig beschreiben:

Die Buchstaben des Bandalphabets $\Gamma = \{a_0,\dots,a_{p+1}\}$, die Bewegungen $\{-1,0,1\}$ und die Zustände $Z = \{0,\dots,m\}$ von M kodieren wir hierzu zunächst unär:

$$\begin{aligned} \ulcorner a_i \urcorner &= 1^{i+1} \qquad (0 \leq i \leq p+1) \\ \ulcorner -1 \urcorner &= 1, \quad \ulcorner 0 \urcorner = 11, \quad \ulcorner +1 \urcorner = 111 \\ \ulcorner z \urcorner &= 1^{z+1} \qquad (0 \leq z \leq m) \end{aligned}$$

Das Programm δ fassen wir dann als endliche Menge von Instruktionen $\delta = \{I_1, \ldots I_r\}$ auf. Einer Instruktion $I = (z, b_1, \ldots, b_k, \hat{b}_1, \ldots, \hat{b}_k, B_1, \ldots, B_k, \hat{z}) \in \delta$ ordnen wir das Binärwort

$$\ulcorner I \urcorner = \ulcorner z \urcorner 0 \ulcorner b_1 \urcorner 0 \ldots \ulcorner b_k \urcorner 0 \ulcorner \hat{b}_1 \urcorner 0 \ldots \ulcorner \hat{b}_k \urcorner 0 \ulcorner B_1 \urcorner 0 \ldots \ulcorner B_k \urcorner 0 \ulcorner \hat{z} \urcorner$$

zu und dem Programm δ das Wort

$$\ulcorner \delta \urcorner = 00 \ulcorner I_1 \urcorner 00 \ulcorner I_2 \urcorner 00 \ldots \ulcorner I_r \urcorner.$$

Schließlich identifizieren wir

$$\ulcorner M \urcorner = \ulcorner \delta \urcorner$$

und nennen das Binärwort $\ulcorner M \urcorner$ den *Index* oder die *Gödelnummer* der Maschine M.

Eine universelle Turingmaschine U interpretiert (simuliert) nun alle normierten Turingmaschinen eines gegebenen Typs. D.h. bei Eingabe (w, x) fasst U das erste Eingabewort w als Index einer Maschine M und die zweite Eingabe als Eingabe von M auf.

5.3 SATZ. *Zu jedem Typ T von Turingmaschinen mit k (Halb-)Bändern ($k \geq 1$ fest) und mit Eingabealphabet $\Sigma \supseteq \Sigma_2$ zur Berechnung n-stelliger Funktionen über Σ^* oder zur Erkennung n-stelliger Sprachen über Σ^* kann man effektiv eine $k+2$ (Halb-)Band-Turingmaschine U_T mit einer zusätzlichen Eingabe vom ansonsten gleichen Typ T angeben, sodass folgendes gilt: Ist M eine normierte Turingmaschine vom Typ T, so hat U_T bei Eingabe $(\ulcorner M \urcorner, \vec{w})$ dasselbe Ausgabe- bzw. Akzeptanzverhalten wie M bei Eingabe $\vec{w}$ und es gilt*

$$\begin{aligned} \mathrm{time}_{U_T}(\ulcorner M \urcorner, \vec{w}) &\leq O(|\ulcorner M \urcorner| \cdot \mathrm{time}_M(\vec{w})) \\ \mathrm{space}_{U_T}(\ulcorner M \urcorner, \vec{w}) &\leq O(|\ulcorner M \urcorner| + \mathrm{space}_M(\vec{w})). \end{aligned}$$

BEWEISIDEE. Bei Eingabe $(x, \vec{w})$ schreibt U_T die erste Eingabe x auf Band $k+1$ und die Kodierung des Startzustandes normierter Turingmaschinen $\ulcorner 0 \urcorner = 1$ auf Band $k+2$. (Im Falle von on-line Vollband-TMs löscht U_T dabei gleichzeitig die erste Eingabe x auf Band 1. Sonst hält sich U_T im folgenden auf dem Eingabeband immer rechts von x auf. Im Falle von on-line Transduktoren ist Band k wiederum das Ausgabeband, d.h. die Bänder von U_T sind geeignet umzunummerieren!) U_T interpretiert dann x als Index einer TM M vom Typ T und simuliert M Schritt-für-Schritt mit Hilfe des folgenden Zyklus.

Zu Beginn und Ende jeden Zyklus stimmen Inschriften und Positionen der Arbeitsfelder von M und U_T auf Band 1 bis k überein, und der aktuelle (kodierte) M-Zustand steht auf Band $k+2$. U_T merkt sich zunächst die Inschriften $b_1, \ldots, b_k$ der ersten k Arbeitsfelder (in binärkodierter Form) in seinem Zustand (da k fest ist, ist dies möglich). Durch Mustervergleiche findet U_T dann die (Kodierung der) anzuwendende(n) M-Instruktion, indem es den ersten (im nd. Fall einen beliebigen) Block der Form $00z0 \ulcorner b_1 \urcorner 0 \ldots \ulcorner b_k \urcorner$ auf Band $k+1$ sucht, wobei z die aktuelle Inschrift von Band $k+2$ ist. U_T simuliert dann diese Instruktion durch Aktualisierung der Inschriften und Positionen der Arbeitsfelder auf den ersten k Bändern und durch Aktualisierung des kodierten M-Zustandes auf Band $k+2$. Der Zeitaufwand für solch einen Zyklus ist linear in der Länge des kodierten Programmes $\ulcorner M \urcorner$. □

Um die gewünschten Aufzählungssätze hieraus abzuleiten, führen wir zunächst folgende Notation ein.

5.4 DEFINITION. Sei F eine Klasse von (partiellen) Funktionen über Σ^* und sei C eine Klasse von Sprachen über dem Alphabet Σ.

(i) Eine (partielle) $(n+1)$-stellige Funktion φ über Σ^* ist *schwach n-universell für* F, falls $\{\varphi_v : v \in \Sigma^*\}$ gerade die Menge der n-stelligen (partiellen) Funktionen in F ist, wobei φ_v durch $\varphi_v(\vec{w}) = \varphi(v,\vec{w})$ definiert ist. Gilt zusätzlich $\varphi \in$ F, so heißt φ *n-universell* für F.

(ii) Eine $(n+1)$-stellige Sprache L über dem Alphabet Σ ist *schwach n-universell* für C, falls $\{L_v : v \in \Sigma^*\}$ gerade die Menge der n-stelligen Sprachen in C ist, wobei $L_v = \{\vec{w} : (v,\vec{w}) \in L\}$. Gilt zusätzlich $L \in$ C, so heißt L *n-universell*.

Wir werden in der Regel nur 1-stellige Funktionen und Sprachenklassen betrachten. Wir sprechen dann einfacher von (schwacher) Universalität. Um hier überhaupt universelle Funktionen bzw. Sprachen erhalten zu können, benutzen wir geeignete Paarfunktionen. Möglich ist hier z.B. die in der Vorlesung "Einführung in die Theoretische Informatik" eingeführte Funktion τ, die sich durch Durchlaufen der Nebendiagonalen ergibt. Verzichten wir auf die Surjektivität, so können wir (für das binäre Alphabet) die Kodierung

$$\langle x,y\rangle := 1^{|x|}0xy$$

benutzen. (Für 1-stelliges φ und L bezeichnen entsprechend $\varphi_v(w) = \varphi(\langle v,w\rangle)$ und $L_v = \{w : \langle v,w\rangle \in L\}$.)

Unmittelbar aus Satz 5.3 können wir die Existenz n-universeller Funktionen bzw. Sprachen für die Klasse der partiell-rekursiven Funktionen und die Klasse der r.a. Mengen ableiten.

5.5 KOROLLAR. *Die Klasse* PFREK *der partiell rekursiven Funktionen besitzt n-universelle Funktionen. Ebenso besitzt die Klasse* RA *der r.a. Mengen n-universelle Sprachen.*

BEWEISIDEE. Da eine n-stellige Funktion ψ über Σ_2^* genau dann partiell rekursiv ist, wenn sie von einer deterministischen Turingmaschine berechnet wird, folgt aus dem 1. Bandreduktionssatz und dem Normierungslemma (Lemma 5.2), dass jede solche Funktion von einer deterministischen normierten on-line 1-Band-Turingmaschine zur Berechnung n-stelliger Funktionen über Σ_2^* berechnet wird. Für die universelle Turingmaschine U_T des entsprechenden Typs T ist die von U_T berechnete Funktion φ daher n-universell für PFREK.

Eine n-universelle Sprache L für RA erhält man entsprechend, indem man die akzeptierte Sprache des universellen Akzeptors U_T für den Typ T der deterministischen normierten on-line 1-Band-Turingakzeptoren zur Erkennung n-stelliger Sprachen über dem Alphabet Σ_2 betrachtet. □

Ist φ eine universelle Funktion für PFREK und gilt $\varphi_v = \psi$, so nennen wir v einen (φ-)*Index* für ψ. Die universelle Funktion φ für PFREK, die wir im Beweis von Korollar 5.5 über die zugehörige universelle Turingmaschine erhalten haben, hat die folgende interessante Eigenschaft: Ist φ' eine andere universelle Funktion für PFREK, so kann man aus einem φ'-Index v einer partiell rekursiven Funktion ψ effektiv einen φ-Index v' für ψ berechnen.

5.6 DEFINITION. Eine universelle Funktion φ für PFREK heisst *Standardaufzählung* oder *Gödelnummerierung* von PFREK, falls es zu jeder 2-stelligen partiell rekursiven Funktion φ' über Σ_2^* eine total rekursive Funktion h gibt, sodass $\varphi_{h(v)} = \varphi_v$ für alle Binärwörter v gilt. Die Funktion h bezeichnet man als *Übersetzungsfunktion* von φ' nach φ.

5.7 KOROLLAR. *Die im Beweis von Korollar 5.5 durch Gödelisierung der normierten 1-Band-Turingmaschinen erhaltene universelle Funktion* φ *für* PFREK *ist eine Gödelnummerierung.*

BEWEISIDEE. Sei φ' eine 2-stellige Funktion über Σ_2^* und sei M eine normierte 1-Band-Turingmaschine, die φ' berechne. Dann lässt sich effektiv für jedes Binärwort v eine normierte Turingmaschine M_v angeben, die φ'_v berechnet, d.h. sich bei Eingabe w wie M bei Eingabe (v,w) verhält: Nämlich M_v schreibt zunächst das Wort v vor die Eingabe w und simuliert dann M. Wegen der Effektivität der Gödelisierung der normierten Maschinen ist daher die Funktion h mit $h(v) = \ulcorner M_v \urcorner$ berechenbar und damit nach Churchscher These rekursiv. Nach Definition von φ gilt jedoch gerade $\varphi'_v = \varphi_{h(v)}$, weshalb h die gesuchte Übersetzungsfunktion von φ' nach φ ist. □

Man beobachtet allgemein, dass "natürliche" universelle Maschinen für die üblichen formalen Berechnungsmodelle zu universellen Funktionen führen, die Gödelnummerierungen sind. Im nächsten Kapitel, in dem wir abstrakte Komplexitätsmaße betrachten, werden wir hierauf zurückkommen.

Im Gegensatz zu Korollar 5.5 besitzen die total rekursiven Funktionen bzw. die rekursiven Mengen keine universellen Objekte. Dies zeigt man durch ein einfaches Diagonalargument.

5.8 LEMMA. *Die Klassen* FREK *und* REK *der total rekursiven Funktionen und rekursiven Sprachen besitzen keine universellen Funktionen bzw. Sprachen.*

BEWEIS. Wir betrachten den Fall 1-universeller Sprachen für REK. Die anderen Fälle sind ähnlich. Wir gehen von der Widerspruchsannahme aus, dass U 1-universell für REK sei. Dann gilt für die durch

$$x \in D \Leftrightarrow \langle x,x\rangle \notin U$$

definierte Sprache, dass D rekursiv ist (da U rekursiv ist). Nach Definition gilt jedoch $D \neq U_x$ (da $x \in D$ g.d.w. $x \notin U_x$) für alle x, was der Universalität von U widerpricht. □

Eine Variante des Beweises von Lemma 5.8 zeigt, dass das Halteproblem für Turingmaschinen, d.h. die Frage, ob eine Turingmaschine bei einer Eingabe terminiert, unentscheidbar ist. Formal ist das *Halteproblem* definiert durch $K = \{x : \varphi_x(x)\downarrow\}$, wobei φ eine Standardaufzählung der 1-stelligen partiell rekursiven Funktionen über dem binären Alphabet ist, z.B. die Aufzählung, die man durch Gödelisierung der normierten 1-Band-TMs erhält. Wäre K rekursiv, so wäre die durch

$$\psi(x) = \begin{cases} 1 - \varphi_x(x) & \text{falls } \varphi_x(x)\downarrow \\ 0 & \text{sonst} \end{cases}$$

definierte partielle Funktion ψ partiell berechenbar, also nach Churchscher These partiell rekursiv. Nach Definition unterscheidet sich ψ aber von jedem Zweig φ_x von φ (nämlich gerade an der Stelle x). Dies widerspricht aber der Universalität von φ. Das Halteproblem ist jedoch rekursiv aufzählbar, da die Funktion $\psi(x) = \varphi_x(x)$ partiell rekursiv ist und eine Maschine M, die ψ berechnet, gerade auf den Eingaben x aus K terminiert, sich also leicht in einen Akzeptor für K übersetzen lässt. K ist also ein Beispiel für eine r.a. Sprache, die nicht rekursiv ist.

Lemma 5.8 lässt sich verallgemeinern (s. “Theoretische Informatik”). So besitzen z.B. die üblichen Komplexitätsklassen ebenfalls keine universellen Mengen. Im Gegensatz zu REK können wir hier jedoch schwach universelle rekursive Mengen angeben. Klassen mit dieser Eigenschaft heissen uniform rekursiv.

5.9 DEFINITION. Eine Klasse F totaler Funktionen (bzw. eine Klasse C von Sprachen) heißt *uniform rekursiv* oder *rekursiv präsentierbar* (r.p.), wenn F (C) eine rekursive schwach universelle Funktion (Sprache) besitzt oder leer ist.

Ist $\mathrm{C} \neq \emptyset$ uniform rekursiv, so gibt es eine rekursive Sprache U mit $\mathrm{C} = \{U_x : x \in \Sigma^*\}$. Die Klasse C enthält daher nur rekursive Mengen, ist insbesondere also abzählbar. Nach Lemma 5.8 ist REK ein Beispiel für eine Klasse, die nur rekursive Sprachen enthält, aber nicht uniform rekursiv ist. Zum Nachweis der uniformen Rekursivität sind folgende Beobachtungen nützlich. Hierbei sagen wir, dass eine Funktionsklasse F *gegen endliche Varianten abgeschlossen* ist, wenn

$$\forall f \in \mathrm{F}(f \stackrel{*}{=} f' \;\&\; f' \text{ total } \Rightarrow f' \in \mathrm{F})$$

gilt, wobei $f \stackrel{*}{=} f'$ ausdrückt, dass sich f und f' nur in endlich vielen Argumenten unterscheiden, d.h. $f = f'$ fast überall gilt (entsprechend für Sprachen).

5.10 LEMMA. *(a) Sind* C *und* D *uniform rekursiv, so sind auch* $\mathrm{C} \cup \mathrm{D}$ *und* $\text{CO-C} = \{\overline{L} : L \in \mathrm{C}\}$ *uniform rekursiv. Sind* C *und* D *zusätzlich gegen endliche Varianten abgeschlossen, so ist auch* $\mathrm{C} \cap \mathrm{D}$ *uniform rekursiv.*

(b) Ist C *uniform rekursiv, so auch der Abschluss von* C *unter endlichen Varianten:* $\mathrm{Fin}(\mathrm{C}) = \{L : \exists L' \in \mathrm{C}(L \stackrel{*}{=} L')\}$.

(c) Jede endliche Klasse ist uniform rekursiv.

BEWEIS. Seien U und V rekursive schwach universelle Mengen für C bzw. D. Weiter nehmen wir an, dass C und D Sprachen über dem binären Alphabet entalten und wir identifizieren Σ_2^* mit $\mathbb{N}$ ($x = \underline{x}$). Dann ist W mit

$$\begin{aligned} \langle 2x, y\rangle \in W \quad &:\Leftrightarrow \quad \langle x, y\rangle \in U \\ \langle 2x+1, y\rangle \in W \quad &:\Leftrightarrow \quad \langle x, y\rangle \in V \end{aligned}$$

rekursiv und schwach universell für $\mathrm{C} \cup \mathrm{D}$. Ähnlich ist W' mit

$$\langle x, y\rangle \in W' :\Leftrightarrow \langle x, y\rangle \notin U$$

eine rekursive schwache universelle Menge für CO-C. Zum Nachweis der uniformen Rekursivität von $\mathrm{C} \cap \mathrm{D}$ können wir o.B.d.A. $\mathrm{C} \cap \mathrm{D} \neq \emptyset$ annehmen, also eine rekursive Sprache $L \in \mathrm{C} \cap \mathrm{D}$ wählen. Wegen des angenommenen Abschlusses von C und D gegen endliche Varianten, liegen dann auch alle endlichen Varianten von L im Durchschnitt von C und D. Definieren wir also W'' durch

$$\langle\langle x,y\rangle,z\rangle \in W'' \;\Leftrightarrow\; \langle x,z\rangle \in U \;\;\&\;\; \langle y,z\rangle \in V$$

falls

$$\forall z' \leq z(\langle x,z'\rangle \in U \Leftrightarrow \langle y,z'\rangle \in V)$$

und durch

$$\langle\langle x,y\rangle,z\rangle \in W'' \Leftrightarrow z \in L$$

sonst (kodiert u kein Paar, so setzen wir $W''_u = L$), so ist W'' (nach Churchscher These bzw. formal mit Hilfe elementarer Abschlusseigenschaften von REK) rekursiv. Zu zeigen bleibt

$$\{W''_{\langle x,y\rangle} : x,y \in \Sigma_2^*\} = \mathrm{C} \cap \mathrm{D}.$$

Zum Nachweis der Inklusion "$\subseteq$" beobachtet man hierzu, dass für jedes Paar $\langle x,y\rangle$ entweder $W''_{\langle x,y\rangle} = U_x = V_y$ oder $W''_{\langle x,y\rangle} \stackrel{*}{=} L$ gilt. Zum Nachweis von "$\supseteq$" wählt man ein beliebiges $L' \in \mathrm{C} \cap \mathrm{D}$. Dann gibt es aber wegen der Universalitätseigenschaft von U und V Wörter x und y mit $L' = U_x$ und $L' = V_y$, also $L' = W''_{\langle x,y\rangle}$.

Zum Nachweis von (b) definiert man $\tilde{W}$ durch

$$\langle\langle x,y\rangle,z\rangle \in \tilde{W} \Leftrightarrow (z \leq |y| \;\;\&\;\; y(z) = 1) \;\vee\; (|y| < z \;\;\&\;\; z \in U_x).$$

D.h. $\tilde{W}_{\langle x,y\rangle}$ ist auf den ersten $|y|$ Argumenten durch y festgelegt und stimmt auf den größeren Argumenten mit U_x überein. D.h. $\{\tilde{W}_{\langle x,y\rangle} : y \in \Sigma_2^*\} = \mathrm{Fin}(\{U_x\})$, woraus die Behauptung leicht folgt.

Zum Beweis von (c) sei $\mathrm{E} = \{L_0, \ldots, L_n\}$ endlich aber o.B.d.A. nicht leer. Dann ist $\hat{W}$ mit

$$\langle x,y\rangle \in \hat{W} \Leftrightarrow (x \leq n \;\;\&\;\; y \in L_x) \;\vee\; (x > n \;\;\&\;\; y \in L_n)$$

schwach universell für E. □

Im folgenden nennen wir die Vereinigung von Komplexitätsklassen desselben Typs, die durch uniform rekursive Schrankenfunktionen gegeben sind, eine *allgemeine Komplexitätsklasse*. Beispiele solcher allgemeiner Komplexitätsklassen, die wir im folgenden studieren werden, sind:

$$\mathrm{P} = \mathrm{PTIME} = \mathrm{DTIME}(\mathrm{poly}) = \bigcup_{p \text{ Polynom}} \mathrm{DTIME}(p(n))$$

((Det.) Polynomialzeit)

$$\mathrm{NP} = \mathrm{NPTIME} = \mathrm{NTIME}(\mathrm{poly}) = \bigcup_{p \text{ Polynom}} \mathrm{NTIME}(p(n))$$

(Nd. Polynomialzeit)

$$\mathrm{PSPACE} = \mathrm{DSPACE}(\mathrm{poly}) = \bigcup_{p \text{ Polynom}} \mathrm{DSPACE}(p(n))$$

((Det.) Polynomialplatz)

$$\mathrm{E} = \mathrm{DTIME}(2^{\mathrm{lin}}) = \bigcup_{k \geq 1} \mathrm{DTIME}(2^{k \cdot n})$$

((Det.) Linear-Exponentialzeit)

$$\mathrm{EXP} = \mathrm{DTIME}(2^{\mathrm{poly}}) = \bigcup_{p \text{ Polynom}} \mathrm{DTIME}(2^{p(n)})$$

((Det.) Polynomiell-Exponentialzeit)

(Man mache sich klar, dass die Schranken jeweils uniform rekursiv sind!)

Die uniforme Rekursivität der Schranken überträgt sich auf die Komplexitätsklassen.

5.11 LEMMA. *Sei* $\mathrm{C} \in \{\mathrm{DTIME}, \mathrm{DSPACE}, \mathrm{NTIME}, \mathrm{NSPACE}\}$ *und sei* F *uniform rekursiv, wobei* F *eine nichttriviale Schranke* f*, d.h.* $f \geq \log(n)$ *für Platz- und* f *hyperlinear für Zeitklassen, enthalte. Dann ist*

$$\mathrm{C}(\mathrm{F}) = \bigcup_{f \in \mathrm{F}} \mathrm{C}(f)$$

ebenfalls uniform rekursiv und gegen endliche Varianten abgeschlossen.

BEWEIS. Wir betrachten den Fall $\mathrm{C} = \mathrm{DTIME}$. Der Abschluss von $\mathrm{DTIME}(f)$ und damit von $\mathrm{DTIME}(\mathrm{F})$ gegen endliche Varianten folgt aus der Tatsache, dass die Schranken nur asymptotisch gelten müssen. Zum Nachweis der uniformen Rekursivität von $\mathrm{DTIME}(\mathrm{F})$ wählen wir zunächst φ total rekursiv mit $\mathrm{F} = \{\varphi_x : x \in \Sigma_2^*\}$.

Aus dem linearen Beschleunigungssatz und Lemma 5.2 folgt dann:

$$\begin{aligned} L \in \mathrm{DTIME}(\mathrm{F}) \quad &\Leftrightarrow \quad \exists f \in \mathrm{F}(L \in \mathrm{DTIME}(f)) \\ &\Leftrightarrow \quad \exists y (L \in \mathrm{DTIME}(\varphi_y)) \\ &\Leftrightarrow \quad \exists y \exists k \exists M \exists c (M \text{ normierter } (c\varphi_y + c)\text{-zeitbeschränkter} \\ &\qquad \text{on-line } k\text{-Band-TA und } L = L(M)) \end{aligned}$$

wobei wir in der letzten Charakterisierung davon ausgehen dürfen, dass die Zeitschranke $c \cdot \varphi_y + c$ für alle Eingabenlängen gilt (sonst c geeignet vergrößern!). Jede Sprache $L \in \mathrm{DTIME}(\mathrm{F})$ wird daher durch ein Tupel $\langle y, k, w, c \rangle$ beschrieben, wobei w die Gödelnummer eines normierten det. k-Band-TA M ist, der überall $c \cdot \varphi_y(n) + c$ zeitbeschränkt ist (und umgekehrt beschreibt jedes Tupel $\langle y, k, w, c \rangle$ mit diesen Eigenschaften eine Sprache in $\mathrm{DTIME}(\mathrm{F})$). Definieren wir also V durch

$$\langle \langle y, k, w, c \rangle, x \rangle \in V \Leftrightarrow x \in L(M)$$

falls w die Gödelnummer eines normierten det. k-Band-TA M ist, dessen Rechenzeit für alle Eingaben der Länge $n \leq |x|$ durch $c \cdot \varphi_y(n) + c$ beschränkt ist, und

$$\langle \langle y, k, w, c \rangle, x \rangle \notin V$$

andernfalls, so ist V schwach universell für $\mathrm{DTIME}(\mathrm{F})$. (Die Klasse der Zweige $V_{\langle y,k,w,c \rangle}$ von V, die gemäß dem ersten Fall definiert sind, ist nämlich gerade $\mathrm{DTIME}(\mathrm{F})$; die gemäß Fall 2 definierten Zweige $V_{\langle y,k,w,c \rangle}$ sind endlich und liegen damit ebenfalls in

DTIME(F).) Weiter ist V entscheidbar, also nach Churchscher These rekursiv. (Gegeben ein Wort z können wir nämlich effektiv feststellen, ob das Wort die Gestalt $z = \langle\langle y,k,w,c\rangle,x\rangle$ hat, im positiven Fall die Komponenten y,k,w,c,x bestimmen, prüfen, ob w die Gödelnummer eines normierten on-line k-Band-TA M ist, und gegebenenfalls M angeben. Weiter können wir in diesem Fall wegen der Rekursivität von φ, $t(n) = c \cdot \varphi_y(n) + c$ für $n \leq |x|$ berechnen, und schließlich für jedes Wort v der Länge $n \leq |x|$ prüfen, ob die Laufzeit von M bei Eingabe v durch $t(n)$ beschränkt ist. Dass diese Laufzeitanalyse effektiv durchführbar ist, ergibt sich aus der Tatsache, dass der Graph der Rechenzeitfunktion rekursiv ist; vgl. Lemma 3.2.) $\square$