

Einführung in die Theoretische Informatik

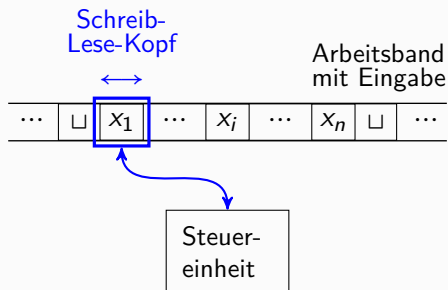
Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

WS 2011/12

Die Turingmaschine



- Eine Turingmaschine erhält ihre Eingabe auf einem nach links und rechts unbegrenzten Band.
- Während ihrer Rechnung kann sie den Schreib-Lese-Kopf auf dem Band in beide Richtungen bewegen und dabei die besuchten Bandfelder lesen sowie die gelesenen Zeichen gegebenenfalls überschreiben.

Entscheidbare und semi-entscheidbare Sprachen

Definition

- Eine NTM M **hält** bei Eingabe x , falls alle Rechnungen von $M(x)$ eine endliche Länge haben.
- Eine NTM M **entscheidet** eine Eingabe x , falls $M(x)$ hält oder eine Konfiguration mit einem Endzustand erreichen kann.
- Eine Sprache $L \subseteq \Sigma^*$ heißt **entscheidbar**, falls eine DTM M mit $L(M) = L$ existiert, die jede Eingabe $x \in \Sigma^*$ entscheidet.
- Jede von einer DTM M erkannte Sprache heißt **semi-entscheidbar**.

Bemerkung

- Die von M akzeptierte Sprache $L(M)$ heißt semi-entscheidbar, da M zwar alle Eingaben $x \in L$ entscheidet (aber eventuell nicht alle $x \in \bar{L}$).
- Später werden wir sehen, dass genau die Typ-0 Sprachen semi-entscheidbar sind.

Berechnung von Funktionen

Definition

- Eine k -DTM $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ **berechnet** eine Funktion $f : \Sigma^* \rightarrow \Gamma^*$, falls M bei jeder Eingabe $x \in \Sigma^*$ in einer Konfiguration

$$K = (q, u_1, a_1, v_1, \dots, u_k, a_k, v_k) \in Z \times (\Gamma^* \times \Gamma \times \Gamma^*)^k$$

hält (d.h. $K_x \vdash^* K$ und K hat keine Folgekonfiguration) mit

$$u_k = f(x).$$

- Hierfür sagen wir auch, M gibt bei Eingabe x das Wort $f(x)$ aus und schreiben $M(x) = f(x)$.
- f heißt **Turing-berechenbar** (oder einfach **berechenbar**), falls es eine k -DTM M mit $M(x) = f(x)$ für alle $x \in \Sigma^*$ gibt.
- Aus historischen Gründen werden berechenbare Funktionen auch **rekursiv** genannt.

Berechenbarkeit von partiellen Funktionen

Definition

- Eine **partielle Funktion** hat die Form $f : \Sigma^* \rightarrow \Gamma^* \cup \{\uparrow\}$.
- Für $f(x) = \uparrow$ sagen wir auch $f(x)$ ist **undefiniert**.
- Der **Definitionsbereich** (engl. *domain*) von f ist

$$\text{dom}(f) = \{x \in \Sigma^* \mid f(x) \neq \uparrow\}.$$

- Das **Bild** (engl. *image*) von f ist

$$\text{img}(f) = \{f(x) \mid x \in \text{dom}(f)\}.$$

- f heißt **total**, falls $f(x) \neq \uparrow$ für alle $x \in \Sigma^*$ ist.
- Eine k -DTM $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ **berechnet** f , falls $M(x)$ für alle $x \in \text{dom}(f)$ das Wort $f(x)$ ausgibt und für alle $x \notin \text{dom}(f)$ keine Ausgabe berechnet (d.h. $M(x)$ darf im Fall $x \notin \text{dom}(f)$ nicht halten).

Berechen- und Entscheidbarkeit

Wir fassen die entscheidbaren Sprachen und die (partiellen) berechenbaren Funktionen in folgenden Klassen zusammen:

$REC = \{L(M) \mid M \text{ ist eine DTM, die jede Eingabe entscheidet}\},$

$FREC = \{f \mid f \text{ ist eine berechenbare (totale) Funktion}\},$

$FREC_p = \{f \mid f \text{ ist eine berechenbare partielle Funktion}\}.$

Dann gilt:

- $FREC \not\subseteq FREC_p$ und
- $REG \not\subseteq DCFL \not\subseteq CFL \not\subseteq DCSL \subseteq CSL \not\subseteq REC \not\subseteq RE.$

Berechenbarkeit von partiellen Funktionen

Beispiel

- Bezeichne x^+ den **lexikografischen Nachfolger** von $x \in \Sigma^*$.
- Für $\Sigma = \{0, 1\}$ ergeben sich beispielsweise folgende Werte:

x	ε	0	1	00	01	10	11	000	...
x^+	0	1	00	01	10	11	000	001	...

- Betrachte die auf Σ^* definierten partiellen Funktionen f_1, f_2, f_3, f_4 mit

$$\begin{aligned} f_1(x) &= 0, \\ f_2(x) &= x, \\ f_3(x) &= x^+ \end{aligned} \quad \text{und} \quad f_4(x) = \begin{cases} \uparrow, & x = \varepsilon, \\ y, & x = y^+. \end{cases}$$

- Da f_1, f_2, f_3, f_4 berechenbar sind, gehören die totalen Funktionen f_1, f_2, f_3 zu FREC und die partielle Funktion f_4 zu FREC_p . \triangleleft

Berechenbarkeit von charakteristischen Funktionen

Satz

- Eine Sprache $A \subseteq \Sigma^*$ ist genau dann entscheidbar, wenn ihre **charakteristische Funktion** $\chi_A : \Sigma^* \rightarrow \{0, 1\}$ berechenbar ist:

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

- Eine Sprache $A \subseteq \Sigma^*$ ist genau dann semi-entscheidbar, falls ihre **partielle charakteristische Funktion** $\hat{\chi}_A : \Sigma^* \rightarrow \{0, 1, \uparrow\}$ berechenbar ist:

$$\hat{\chi}_A(x) = \begin{cases} 1, & x \in A, \\ \uparrow, & x \notin A. \end{cases}$$

Beweis

Siehe Übungen.

Charakterisierung der rekursiv aufzählbaren Sprachen

Definition

Eine Sprache $A \subseteq \Sigma^*$ heißt **rekursiv aufzählbar**, falls $A = \emptyset$ oder das Bild $\text{img}(f)$ einer berechenbaren Funktion $f : \Gamma^* \rightarrow \Sigma^*$ ist.

Satz

Folgende Eigenschaften sind äquivalent:

- 1 A ist semi-entscheidbar (d.h. A wird von einer DTM akzeptiert),
- 2 A wird von einer 1-DTM akzeptiert,
- 3 A wird von einer 1-NTM akzeptiert,
- 4 A ist vom Typ 0,
- 5 A wird von einer NTM akzeptiert,
- 6 A ist rekursiv aufzählbar.

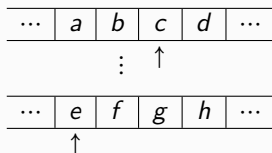
Beweis

Die Implikation 2 \Rightarrow 3 ist klar. Die Implikationen 3 \Rightarrow 4 \Rightarrow 5 werden in den Übungen gezeigt. Hier zeigen wir 1 \Rightarrow 2 und 5 \Rightarrow 6 \Rightarrow 1.

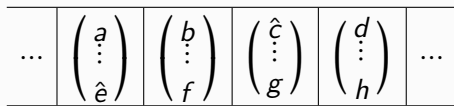
Simulation einer k -DTM durch eine 1-DTM

Beweis von ① \Rightarrow ②: $\{L(M) \mid M \text{ ist eine DTM}\} \subseteq \{L(M) \mid M \text{ ist eine 1-DTM}\}$

- Sei $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ eine k -DTM mit $L(M) = A$.
- Wir konstruieren eine 1-DTM $M' = (Z', \Sigma, \Gamma', \delta', z_0, E)$ für A .
- M' simuliert M , indem sie jede Konfiguration K von M der Form



durch eine Konfiguration K' folgender Form nachbildet:



Simulation einer k -DTM durch eine 1-DTM

Beweis von ① \Rightarrow ②: $\{L(M) \mid M \text{ ist eine DTM}\} \subseteq \{L(M) \mid M \text{ ist eine 1-DTM}\}$

- Das heißt, M' arbeitet mit dem Alphabet

$$\Gamma' = \Gamma \cup (\Gamma \cup \{\hat{a} \mid a \in \Gamma\})^k$$

- und erzeugt bei Eingabe $x = x_1 \dots x_n \in \Sigma^*$ zuerst die der Startkonfiguration

$$K_x = (q_0, \varepsilon, x_1, x_2 \dots x_n, \varepsilon, \sqcup, \varepsilon, \dots, \varepsilon, \sqcup, \varepsilon)$$

von M bei Eingabe x entsprechende Konfiguration

$$K'_x = q'_0 \begin{pmatrix} \hat{x}_1 \\ \hat{\sqcup} \\ \vdots \\ \hat{\sqcup} \end{pmatrix} \begin{pmatrix} x_2 \\ \sqcup \\ \vdots \\ \sqcup \end{pmatrix} \dots \begin{pmatrix} x_n \\ \sqcup \\ \vdots \\ \sqcup \end{pmatrix}.$$

Simulation einer k -DTM durch eine 1-DTM

Beweis von ① \Rightarrow ②: $\{L(M) \mid M \text{ ist eine DTM}\} \subseteq \{L(M) \mid M \text{ ist eine 1-DTM}\}$

- Dann simuliert M' jeweils einen Schritt von M durch folgende Sequenz von Rechenschritten:
 - Zuerst geht M' solange nach rechts, bis sie alle mit \wedge markierten Zeichen (z.B. $\hat{a}_1, \dots, \hat{a}_k$) gefunden hat.
 - Diese Zeichen speichert M' in ihrem Zustand.
 - Anschließend geht M' wieder nach links und realisiert dabei die durch $\delta(q, a_1, \dots, a_k)$ vorgegebene Anweisung von M .
 - Den aktuellen Zustand q von M speichert M' ebenfalls in ihrem Zustand.
- Sobald M in einen Endzustand übergeht, wechselt M' ebenfalls in einen Endzustand und hält.
- Nun ist leicht zu sehen, dass $L(M') = L(M)$ ist. □

Charakterisierung der rekursiv aufzählbaren Sprachen

Beweis von ⑤ \Rightarrow ⑥: $\{L(M) \mid M \text{ ist eine NTM}\} \subseteq \{L \mid L \text{ ist rek. aufzählbar}\}$

- Sei $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ eine k -NTM und sei $A = L(M) \neq \emptyset$.
- Sei $\tilde{\Gamma}$ das Alphabet $Z \cup \Gamma \cup \{\#\}$.
- Wir kodieren eine Konfiguration $K = (q, u_1, a_1, v_1, \dots, u_k, a_k, v_k)$ durch das Wort

$$\text{code}(K) = \#q\#u_1\#a_1\#v_1\#\dots\#u_k\#a_k\#v_k\#$$

und eine Rechnung $K_0 \vdash \dots \vdash K_t$ durch $\text{code}(K_0) \dots \text{code}(K_t)$.

- Dann lassen sich die Wörter von A durch folgende Funktion $f : \tilde{\Gamma}^* \rightarrow \Sigma^*$ aufzählen (dabei ist x_0 ein beliebiges Wort in A):

$$f(x) = \begin{cases} y, & x \text{ kodiert eine Rechnung } K_0 \vdash \dots \vdash K_t \text{ von } M \text{ mit} \\ & K_0 = K_y \text{ und } K_t \in E \times (\Gamma^* \times \Gamma \times \Gamma^*)^k \\ x_0, & \text{sonst.} \end{cases}$$

- Da f berechenbar ist, ist $A = \text{img}(f)$ rekursiv aufzählbar. □

Charakterisierung der rekursiv aufzählbaren Sprachen

Beweis von ⑥ \Rightarrow ①: $\{L \mid L \text{ ist rek. aufzählbar}\} \subseteq \{L(M) \mid M \text{ ist eine DTM}\}$

- Sei $f : \Gamma^* \rightarrow \Sigma^*$ eine Funktion mit $A = \text{img}(f)$ und sei M eine k -DTM, die f berechnet.
- Betrachte folgende $(k + 1)$ -DTM M' , die bei Eingabe x
 - auf dem 2. Band der Reihe nach alle Wörter y in Γ^* erzeugt,
 - für jedes y den Wert $f(y)$ durch Simulation von $M(y)$ berechnet, und
 - ihre Eingabe x akzeptiert, sobald $f(y) = x$ ist. □

Charakterisierung der entscheidbaren Sprachen

Satz

A ist genau dann entscheidbar, wenn A und \bar{A} semi-entscheidbar sind, d.h.
 $\text{REC} = \text{RE} \cap \text{co-RE}$.

Beweis.

- Falls A entscheidbar ist, ist auch \bar{A} entscheidbar, d.h. A und \bar{A} sind dann auch semi-entscheidbar.
- Für die Rückrichtung seien $f_1, f_2 : \Gamma^* \rightarrow \Sigma^*$ Turing-berechenbare Funktionen mit $\text{img}(f_1) = A$ und $\text{img}(f_2) = \bar{A}$.
- Wir betrachten folgende DTM M , die bei Eingabe x für jedes $y \in \Gamma^*$ die beiden Werte $f_1(y)$ und $f_2(y)$ bestimmt und im Fall
 - $f_1(y) = x$ in einem Endzustand hält,
 - $f_2(y) = x$ in einem Nichtendzustand hält.
- Da jede Eingabe x entweder in $\text{img}(f_1) = A$ oder in $\text{img}(f_2) = \bar{A}$ enthalten ist, hält M bei allen Eingaben. □

Kodierung (Gödelisierung) von Turingmaschinen

- Sei $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ eine 1-DTM mit
 - Zustandsmenge $Z = \{q_0, \dots, q_m\}$ (o.B.d.A. sei $E = \{q_m\}$),
 - Eingabealphabet $\Sigma = \{0, 1, \#\}$ und
 - Arbeitsalphabet $\Gamma = \{a_0, \dots, a_l\}$, wobei wir o.B.d.A. $a_0 = \sqcup$, $a_1 = 0$, $a_2 = 1$ und $a_3 = \#$ annehmen.
- Dann können wir jede Anweisung der Form $q_i a_j \rightarrow q_{i'} a_{j'} D$ durch das Wort

$$\# \text{bin}(i) \# \text{bin}(j) \# \text{bin}(i') \# \text{bin}(j') \# b_D \#$$

kodieren.

- Dabei ist $\text{bin}(n)$ die Binärdarstellung von n und

$$b_D = \begin{cases} 0, & D = N, \\ 1, & D = L, \\ 10, & D = R. \end{cases}$$

Kodierung von Turingmaschinen

- M lässt sich nun als ein Wort über dem Alphabet $\{0, 1, \#\}$ kodieren, indem wir die Anweisungen von M in kodierter Form auflisten.
- Kodieren wir die Zeichen $0, 1, \#$ binär (z.B. $0 \mapsto 00, 1 \mapsto 11, \# \mapsto 10$), so gelangen wir zu einer Binärkodierung w_M von M .
- Die Binärzahl w_M wird auch die **Gödel-Nummer** von M genannt.
- M_w ist durch Angabe von w bis auf die Benennung ihrer Zustände und Arbeitszeichen eindeutig bestimmt.
- Ganz analog lassen sich auch DTMs mit einer beliebigen Anzahl von Bändern (sowie NTMs, Konfigurationen oder Rechnungen von TMs) binär kodieren.
- Umgekehrt können wir jedem Binärstring $w \in \{0, 1\}^*$ eine DTM M_w wie folgt zuordnen:

$$M_w = \begin{cases} M, & \text{falls eine DTM } M \text{ mit } w_M = w \text{ existiert,} \\ M_0, & \text{sonst (dabei sei } M_0 \text{ eine beliebige DTM).} \end{cases}$$

Unentscheidbarkeit des Halteproblems

Definition

- Das **Halteproblem** ist die Sprache

$$H = \left\{ w \# x \mid \begin{array}{l} w, x \in \{0, 1\}^* \text{ und} \\ \text{die DTM } M_w \text{ h\"alt} \\ \text{bei Eingabe } x \end{array} \right\}$$

- Das **spezielle Halteproblem** ist

$$K = \left\{ w \in \{0, 1\}^* \mid \begin{array}{l} \text{die DTM } M_w \\ \text{h\"alt bei Eingabe } w \end{array} \right\}$$

χ_H	x_1	x_2	x_3	\dots
w_1	0	1	0	\dots
w_2	0	1	1	\dots
w_3	1	1	0	\dots
\vdots	\vdots	\vdots	\vdots	\ddots

χ_K				
w_1	0			
w_2		1		
w_3			0	
\vdots				\ddots

Satz

$K \in \text{RE} \setminus \text{REC}$.

Semi-Entscheidbarkeit des speziellen Halteproblems

Beweis von $K \in RE$

- Sei w_0 die Kodierung einer DTM, die bei jeder Eingabe (sofort) hält und betrachte die Funktion

$$f(x) = \begin{cases} w, & x \text{ ist Kodierung einer haltenden Rechnung einer} \\ & \text{DTM } M_w \text{ bei Eingabe } w, \\ w_0, & \text{sonst.} \end{cases}$$

- Da f berechenbar und $\text{img}(f) = K$ ist, folgt $K \in RE$. □

Bemerkung

Ganz ähnlich lässt sich $H \in RE$ zeigen.

Unentscheidbarkeit des speziellen Halteproblems

Beweisidee

- Da $K \in RE$ ist, gibt es in der Matrixdarstellung von χ_H eine Zeile (sprich DTM M_w), die mit der Diagonalen der Matrix übereinstimmt.
- Beispielsweise können wir für M_w eine DTM wählen, die die partielle charakteristische Funktion $\hat{\chi}_K$ von K berechnet.
- Wäre K entscheidbar, so könnten wir eine DTM $M_{\hat{w}}$ konstruieren, so dass die zugehörige Zeile in der Matrix invers zur Zeile von M_w und damit zur Diagonalen ist.
- Da eine solche Matrix nicht existieren kann, führt dies auf den gewünschten Widerspruch.

Unentscheidbarkeit des speziellen Halteproblems

Beweis von $K \notin \text{REC}$

- Angenommen, die Sprache

$$K = \{w \mid M_w(w) \text{ h\"alt}\} \quad (*)$$

w\"are durch eine DTM M_K entscheidbar.

- Betrachte die DTM \hat{M} , die bei Eingabe $w \in \{0,1\}^*$ die DTM $M_K(w)$ simuliert und genau dann h\"alt, wenn $M_K(w)$ verwirft:

$$\hat{M}(w) \text{ h\"alt} \Leftrightarrow w \notin K \quad (**)$$

- F\"ur die Kodierung \hat{w} von \hat{M} folgt dann aber

$$\hat{w} \in K \stackrel{(*)}{\Leftrightarrow} M_{\hat{w}}(\hat{w}) \text{ h\"alt} \stackrel{(**)}{\Leftrightarrow} \hat{w} \notin K \quad \downarrow \text{ (Widerspruch!)} \quad \square$$

χ_H	x_1	x_2	x_3	\dots
w_1	0	1	0	\dots
w_2	0	1	1	\dots
w_3	1	1	0	\dots
\vdots	\vdots	\vdots	\vdots	\ddots

\hat{w}	1	0	1	\dots
-----------	---	---	---	---------

Unentscheidbarkeit des speziellen Halteproblems

Korollar

- $\text{REC} \not\subseteq \text{RE}$,
- $K \in \text{RE} \setminus \text{co-RE}$ (d.h. $\text{RE} \neq \text{co-RE}$),
- $\bar{K} \in \text{co-RE} \setminus \text{RE}$.

Beweis

- $\text{REC} \not\subseteq \text{RE}$: klar, da $K \in \text{RE} - \text{REC}$.
- $K \notin \text{co-RE}$: Aus der Annahme $K \in \text{co-RE}$ würde wegen $K \in \text{RE}$ folgen, dass K entscheidbar ist (Widerspruch).
- $\bar{K} \in \text{co-RE} \setminus \text{RE}$: klar, da $K \in \text{RE} \setminus \text{co-RE}$.



Der Reduktionsbegriff

Definition

Eine Sprache $A \subseteq \Sigma^*$ heißt auf $B \subseteq \Gamma^*$ **reduzierbar** (kurz: $A \leq B$), falls eine berechenbare Funktion $f : \Sigma^* \rightarrow \Gamma^*$ ex., so dass gilt:

$$\forall x \in \Sigma^* : x \in A \Leftrightarrow f(x) \in B.$$

Beispiel

Es gilt $K \leq H$ mittels $f : w \mapsto w\#w$, da für alle $w \in \{0,1\}^*$ gilt:

$$\begin{aligned} w \in K &\Leftrightarrow M_w \text{ ist eine DTM, die bei Eingabe } w \text{ hält} \\ &\Leftrightarrow w\#w \in H. \end{aligned}$$



Abschluss von REC unter \leq

Definition

- Eine Sprachklasse \mathcal{C} heißt **unter \leq abgeschlossen**, wenn für alle Sprachen A, B gilt:

$$A \leq B \wedge B \in \mathcal{C} \Rightarrow A \in \mathcal{C}.$$

Satz

Die Klassen REC und RE sind unter \leq abgeschlossen.

Beweis

- Gelte $A \leq B$ mittels f und sei $B \in \text{REC}$.
- Dann ex. eine DTM M , die χ_B berechnet.
- Betrachte folgende DTM M' :
 - M' berechnet bei Eingabe x zuerst den Wert $f(x)$ und
 - simuliert dann M bei Eingabe $f(x)$.

Abschluss von REC und RE unter \leq

Satz

Die Klasse REC ist unter \leq abgeschlossen.

Beweis.

- Gelte $A \leq B$ mittels f und sei $B \in \text{REC}$.
- Dann ex. eine DTM M , die χ_B berechnet.
- Betrachte folgende DTM M' :
 - M' berechnet bei Eingabe x zuerst den Wert $f(x)$ und
 - simuliert dann M bei Eingabe $f(x)$.
- Wegen $x \in A \Leftrightarrow f(x) \in B$ ist $\chi_A(x) = \chi_B(f(x))$ und daher folgt
$$M'(x) = M(f(x)) = \chi_B(f(x)) = \chi_A(x).$$
- Also berechnet M' die Funktion χ_A , d.h. $A \in \text{REC}$. □

Bemerkung

Der Abschluss von RE unter \leq folgt analog (siehe Übungen).

Der Vollständigkeitsbegriff

Definition

- Eine Sprache A heißt **hart** für eine Sprachklasse \mathcal{C} (kurz: **\mathcal{C} -hart** oder **\mathcal{C} -schwer**), falls jede Sprache $L \in \mathcal{C}$ auf A reduzierbar ist:

$$\forall L \in \mathcal{C} : L \leq A.$$

- Eine \mathcal{C} -harte Sprache A , die zu \mathcal{C} gehört, heißt **\mathcal{C} -vollständig**.

Beispiel

Das Halteproblem H ist RE-vollständig. Es gilt nämlich

- $H \in \text{RE}$ und
- $\forall L \in \text{RE} : L \leq H$

mittels der Reduktionsfunktion $x \mapsto w\#x$, wobei w die Kodierung einer DTM M_w ist, die $\hat{\chi}_L$ berechnet.



Bemerkung

Auch das spezielle Halteproblem K ist RE-vollständig (siehe Übungen).

H ist nicht entscheidbar

Korollar

- $A \leq B \wedge A \notin \text{REC} \Rightarrow B \notin \text{REC}$.
- $A \leq B \wedge A \notin \text{RE} \Rightarrow B \notin \text{RE}$.

Beweis

Aus der Annahme, dass B entscheidbar (bzw. semi-entscheidbar) ist, folgt wegen $A \leq B$, dass dies auch auf A zutrifft (Widerspruch). \square

Bemerkung

Wegen $K \leq H$ überträgt sich somit die Unentscheidbarkeit von K auf H .

Korollar

$H \notin \text{REC}$.

Das Halteproblem bei leerem Band

Definition

Das **Halteproblem bei leerem Band** ist die Sprache

$$H_0 = \left\{ w \in \{0, 1\}^* \mid \begin{array}{l} \text{die DTM } M_w \\ \text{hält bei Eingabe } \varepsilon \end{array} \right\}$$

Satz

H_0 ist RE-vollständig.

Beweis

- $H_0 \in \text{RE}$ folgt wegen $H_0 \leq H \in \text{RE}$ mittels der Reduktionsfunktion $w \mapsto w\#\varepsilon$.

χ_H	x_1	x_2	x_3	\dots
w_1	0	1	0	\dots
w_2	0	1	1	\dots
w_3	1	1	0	\dots
\vdots	\vdots	\vdots	\vdots	\ddots

χ_{H_0}	$x_1 (= \varepsilon)$
w_1	0
w_2	0
w_3	1
\vdots	\vdots

H_0 ist RE-vollständig

Beweis

- $H_0 \in \text{RE}$ folgt wegen $H_0 \leq H \in \text{RE}$ mittels der Reduktionsfunktion $w \mapsto w\#\epsilon$.
- Sei $A \in \text{RE}$ und sei w die Kodierung einer DTM, die $\hat{\chi}_A$ berechnet. Um A auf H_0 zu reduzieren, transformieren wir $x \in \{0,1\}^*$ auf die Kodierung einer DTM M_{w_x} , die zunächst ihre Eingabe durch x ersetzt und dann $M_w(x)$ simuliert. Dann gilt

$$x \in A \iff w_x \in H_0$$

und somit $A \leq H_0$ mittels der Reduktionsfunktion $x \mapsto w_x$.

Korollar

$H_0 \notin \text{REC}$.

Der Satz von Rice

Frage

- Kann man einer beliebig vorgegebenen DTM ansehen, ob die von ihr berechnete Funktion eine gewisse Eigenschaft hat?
- Kann man beispielsweise entscheiden, ob eine gegebene DTM eine totale Funktion berechnet?

Antwort

Nein (es sei denn, die fragliche Eigenschaft ist trivial, d.h. keine oder jede DTM berechnet eine Funktion mit dieser Eigenschaft).

Der Satz von Rice

Definition

- Zu einer Klasse \mathcal{F} von Funktionen definieren wir die Sprache

$$L_{\mathcal{F}} = \{w \in \{0,1\}^* \mid \text{die DTM } M_w \text{ berechnet eine Funktion in } \mathcal{F}\}.$$

- Die Eigenschaft \mathcal{F} heißt **trivial**, wenn $L_{\mathcal{F}} = \emptyset$ oder $L_{\mathcal{F}} = \{0,1\}^*$ ist.

Der Satz von Rice besagt, dass $L_{\mathcal{F}}$ nur für triviale Eigenschaften entscheidbar ist.

Satz (Satz von Rice)

Für jede nicht triviale Eigenschaft \mathcal{F} ist $L_{\mathcal{F}}$ unentscheidbar.

Der Satz von Rice

Beispiel

- Die Sprache

$$L = \{w \in \{0, 1\}^* \mid M_w(0^n) = 0^{n+1} \text{ für alle } n \geq 0\}$$

ist unentscheidbar.

- Dies folgt aus dem Satz von Rice, da die Eigenschaft

$$\mathcal{F} = \{f \mid \{0\}^* \subseteq \text{dom}(f) \wedge f(0^n) = 0^{n+1} \text{ für alle } n \geq 0\}$$

nicht trivial und $L = L_{\mathcal{F}}$ ist.

- \mathcal{F} ist nicht trivial, da z.B. die berechenbare partielle Funktion

$$f(x) = \begin{cases} 0^{n+1}, & x = 0^n \text{ für ein } n \geq 0 \\ \uparrow, & \text{sonst} \end{cases}$$

in \mathcal{F} und die konstante Funktion $g(x) = 0$ nicht in \mathcal{F} enthalten ist.

Der Satz von Rice

Satz (Satz von Rice)

Für jede nicht triviale Eigenschaft \mathcal{F} ist die Sprache $L_{\mathcal{F}}$ unentscheidbar.

Beweis

- Die Idee besteht darin, H_0 (oder $\overline{H_0}$) auf $L_{\mathcal{F}}$ zu reduzieren, indem wir für eine gegebene DTM M_w eine DTM $M_{w'}$ konstruieren mit
$$w \in H_0 \Leftrightarrow M_{w'} \text{ berechnet (k)eine Funktion in } \mathcal{F}.$$
- Hierzu lassen wir $M_{w'}$ bei Eingabe x zunächst einmal die DTM M_w bei Eingabe ε simulieren.
- Falls $w \notin H_0$ ist, berechnet $M_{w'}$ also die überall undefinierte Funktion u mit $u(x) = \uparrow$ für alle $x \in \{0, 1, \#\}^*$.
- Damit die Reduktion gelingt, müssen wir nur noch dafür sorgen, dass $M_{w'}$ im Fall $w \in H_0$ eine Funktion f berechnet, die genau dann die Eigenschaft \mathcal{F} hat, wenn u sie nicht hat.
- Da \mathcal{F} nicht trivial ist, gibt es eine DTM M , die eine solche Funktion f berechnet.

Der Satz von Rice

Beweis (Schluss)

- Da \mathcal{F} nicht trivial ist, gibt es eine DTM M , die eine solche Funktion f berechnet.
- Betrachte die Reduktionsfunktion
$$h(w) = w', \text{ wobei } w' \text{ die Kodierung einer DTM ist, die bei Eingabe } x \text{ zunächst die DTM } M_w(\varepsilon) \text{ simuliert und im Fall, dass } M_w(\varepsilon) \text{ h\u00e4lt, mit der Simulation von } M(x) \text{ fortf\u00e4hrt.}$$
- Dann ist $h : w \mapsto w'$ eine totale berechenbare Funktion und es gilt
$$w \in H_0 \Rightarrow M_{w'} \text{ berechnet } f$$
$$w \notin H_0 \Rightarrow M_{w'} \text{ berechnet } u.$$
- Dies zeigt, dass h das Problem H_0 (oder $\overline{H_0}$) auf $L_{\mathcal{F}}$ reduziert, und da H_0 und $\overline{H_0}$ unentscheidbar sind, muss auch $L_{\mathcal{F}}$ unentscheidbar sein. \square

Der Satz von Rice für Akzeptoren

Der Satz von Rice gilt auch für Eigenschaften, die das Akzeptanzverhalten einer gegebenen Turingmaschine betreffen.

Satz (Satz von Rice für Spracheigenschaften)

Für eine beliebige Sprachklasse \mathcal{S} sei

$$L_{\mathcal{S}} = \{w \in \{0, 1\}^* \mid L(M_w) \in \mathcal{S}\}.$$

Dann ist $L_{\mathcal{S}}$ unentscheidbar, außer wenn $L_{\mathcal{S}} \in \{\emptyset, \{0, 1\}^*\}$ ist.

Beweis

Siehe Übungen.

Das Postsche Korrespondenzproblem (PCP)

Definition

- Sei Σ ein beliebiges Alphabet mit $\# \notin \Sigma$.
- Das **Postsche Korrespondenzproblem über Σ** (kurz PCP_Σ) ist:
gegeben: k Paare $(x_1, y_1), \dots, (x_k, y_k)$ von Wörtern über Σ .
gefragt: Gibt es eine Folge $\alpha = (i_1, \dots, i_n)$, $n \geq 1$, von Indizes $i_j \in \{1, \dots, k\}$ mit $x_{i_1} \dots x_{i_n} = y_{i_1} \dots y_{i_n}$?
- Das **modifizierte PCP über Σ** (kurz MPCP_Σ) fragt nach einer Lösung $\alpha = (i_1, \dots, i_n)$ mit $i_1 = 1$.
- Wir notieren eine PCP-Instanz meist in Form einer Matrix $\begin{pmatrix} x_1 \dots x_k \\ y_1 \dots y_k \end{pmatrix}$ und kodieren sie durch das Wort $x_1 \# y_1 \# \dots \# x_k \# y_k$.

Beispiel

Die Instanz $I = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} a & ab & caa \\ aca & bc & aa \end{pmatrix}$ besitzt wegen

$$x_1 x_3 x_2 x_3 = acaabcaa$$

$$y_1 y_3 y_2 y_3 = acaabcaa$$

die PCP-Lösung $\alpha = (1, 3, 2, 3)$, die auch eine MPCP-Lösung ist.

Das Postsche Korrespondenzproblem

Lemma

Für jedes Alphabet Σ gilt $\text{PCP}_\Sigma \leq \text{PCP}_{\{a,b\}}$.

Beweis

- Sei $\Sigma = \{a_1, \dots, a_m\}$. Für ein Zeichen $a_i \in \Sigma$ sei $\hat{a}_i = ab^{i-1}$ und für ein Wort $w = w_1 \dots w_n \in \Sigma^*$ mit $w_i \in \Sigma$ sei $\hat{w} = \hat{w}_1 \dots \hat{w}_n$.
- Dann folgt $\text{PCP}_\Sigma \leq \text{PCP}_{\{a,b\}}$ mittels der Reduktionsfunktion

$$f : \begin{pmatrix} x_1 \dots x_k \\ y_1 \dots y_k \end{pmatrix} \mapsto \begin{pmatrix} \hat{x}_1 \dots \hat{x}_k \\ \hat{y}_1 \dots \hat{y}_k \end{pmatrix}.$$

□

Beispiel

Sei $\Sigma = \{0, 1, 2\}$. Dann ist $\hat{0} = a$, $\hat{1} = ab$ und $\hat{2} = abb$. Somit ist

$$f \left(\begin{pmatrix} 0 & 01 & 200 \\ 020 & 12 & 00 \end{pmatrix} \right) = \begin{pmatrix} a & aab & abbaa \\ aabba & ababb & aa \end{pmatrix}.$$

Das Postsche Korrespondenzproblem

Im Folgenden lassen wir im Fall $\Sigma = \{a, b\}$ den Index weg und schreiben einfach PCP (bzw. MPCP).

Satz

$\text{MPCP} \leq \text{PCP}$.

Beweis

- Wir zeigen $\text{MPCP} \leq \text{PCP}_\Sigma$ für $\Sigma = \{a, b, \langle, |, \rangle\}$.
- Für ein Wort $w = w_1 \dots w_n$ sei

$$\begin{array}{cccc} \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} \\ \langle w_1 | \dots | w_n \rangle & \langle w_1 | \dots | w_n \rangle & | w_1 | \dots | w_n & w_1 | \dots | w_n \rangle \end{array}$$

Das Postsche Korrespondenzproblem

Beweis von $\text{MPCP} \leq \text{PCP}_\Sigma$

- Wir zeigen $\text{MPCP} \leq \text{PCP}_\Sigma$ für $\Sigma = \{a, b, \langle, |, \rangle\}$.
- Für ein Wort $w = w_1 \dots w_n$ sei

$$\begin{array}{cccc} \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} \\ \hline \langle w_1 | \dots | w_n & w_1 | \dots | w_n & \langle w_1 | \dots | w_n & | w_1 | \dots | w_n \end{array}$$

- Wir reduzieren MPCP mittels folgender Funktion f auf PCP_Σ :

$$f : \begin{pmatrix} x_1 & \dots & x_k \\ y_1 & \dots & y_k \end{pmatrix} \mapsto \begin{pmatrix} \overleftarrow{x_1} & \overleftarrow{x_1} & \dots & \overleftarrow{x_k} & \rangle \\ \overleftarrow{y_1} & \overleftarrow{y_1} & \dots & \overleftarrow{y_k} & | \rangle \end{pmatrix}$$

Beispiel

$$f : \begin{pmatrix} aa & b & bab & bb \\ aab & bb & a & b \end{pmatrix} \mapsto \begin{pmatrix} \langle a|a & a|a & b| & b|a|b| & b|b| \rangle \\ \langle a|a|b & |a|a|b & |b|b & |a & |b \rangle \end{pmatrix}$$

Das Postsche Korrespondenzproblem

Beweis von $\text{MPCP} \leq \text{PCP}$

- Wir zeigen $\text{MPCP} \leq \text{PCP}_\Sigma$ für $\Sigma = \{a, b, \langle, |, \rangle\}$.
- Für ein Wort $w = w_1 \dots w_n$ sei

$$\begin{array}{cccc} \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} & \overleftarrow{w} \\ \hline \langle w_1 | \dots | w_n \rangle & w_1 | \dots | w_n & \langle w_1 | \dots | w_n & | w_1 | \dots | w_n \rangle \end{array}$$

- Wir reduzieren MPCP mittels folgender Funktion f auf PCP_Σ :

$$f : \begin{pmatrix} x_1 & \dots & x_k \\ y_1 & \dots & y_k \end{pmatrix} \mapsto \begin{pmatrix} \overleftarrow{x_1} & \overleftarrow{x_1} & \dots & \overleftarrow{x_k} & \rangle \\ \overleftarrow{y_1} & \overleftarrow{y_1} & \dots & \overleftarrow{y_k} & | \rangle \end{pmatrix}$$

- Da jede MPCP-Lösung $\alpha = (1, i_2, \dots, i_n)$ für I auf eine PCP-Lösung $\alpha' = (1, i_2 + 1, \dots, i_n + 1, k + 2)$ für $f(I)$ führt, folgt

$$I \in \text{MPCP} \Rightarrow f(I) \in \text{PCP}_\Sigma.$$

Das Postsche Korrespondenzproblem

Beweis von MPCP \leq PCP

- Für die umgekehrte Implikation sei $\alpha' = (i_1, \dots, i_n)$ eine PCP-Lösung für

$$f(I) = \left(\begin{array}{cccc} \overleftarrow{x_1} & \overline{x_1} & \dots & \overline{x_k} & \rangle \\ \overleftarrow{y_1} & \overline{y_1} & \dots & \overline{y_k} & | \rangle \end{array} \right).$$

- Dann muss $i_1 = 1$ sein, da $(\overleftarrow{x_1}, \overleftarrow{y_1})$ das einzige Paar in $f(I)$ ist, bei dem beide Komponenten mit demselben Buchstaben anfangen.
- Zudem muss $i_n = k + 2$ sein, da nur das Paar $(\rangle, | \rangle)$ mit demselben Buchstaben aufhört.
- Wählen wir α' von minimaler Länge, so ist $i_j \in \{2, \dots, k + 1\}$ für $j = 2, \dots, n - 1$.
- Dann ist aber

$$\alpha = (i_1, i_2 - 1, \dots, i_{n-1} - 1)$$

eine MPCP-Lösung für I .

Das Postsche Korrespondenzproblem

Satz

PCP ist RE-vollständig und damit unentscheidbar.

Beweis.

- Es ist leicht zu sehen, dass $PCP \in RE$ ist.
- Sei $A \in RE$ und sei $M = (Z, \Sigma, \Gamma, \delta, z_0, E)$ eine 1-DTM für A .
- Wir zeigen $A \leq MPCP_{\Sigma'}$ für $\Sigma' = \Gamma \cup Z \cup \{\{, |, \}\}$.
- Wegen $MPCP_{\Sigma'} \leq PCP$ folgt hieraus $A \leq PCP$.

Beweisidee für die Reduktion $A \leq MPCP_{\Sigma'}$:

Transformiere eine Eingabe $w \in \Sigma^*$ in eine Instanz $f(w) = \begin{pmatrix} x_1 \dots x_k \\ y_1 \dots y_k \end{pmatrix}$, so dass $\alpha = (i_1, \dots, i_n)$ genau dann eine MPCP-Lösung für $f(w)$ ist, wenn das zugehörige **Lösungswort** $x_{i_1} \dots x_{i_n} = y_{i_1} \dots y_{i_n}$ eine akzeptierende Rechnung von $M(w)$ kodiert. Dann gilt

$$w \in A \Leftrightarrow f(w) \in MPCP_{\Sigma'}.$$

Das Postsche Korrespondenzproblem

Beweis von $A \leq \text{MPCP}_{\Sigma'}$

Wir bilden $f(w)$ aus folgenden Wortpaaren:

- 1 $(\langle, \langle | z_0 w)$, „Startregel“
- 2 für alle $a \in \Gamma \cup \{\sqcup\}$: (a, a) , „Kopierregeln“
- 3 für alle $a, a', b \in \Gamma, z, z' \in Z$:
 - $(za, z'a')$, falls $\delta(z, a) = (z', a', N)$,
 - $(za, a'z')$, falls $\delta(z, a) = (z', a', R)$,
 - $(bza, z'ba')$, falls $\delta(z, a) = (z', a', L)$,
 - $(|za, |z' \sqcup a')$, falls $\delta(z, a) = (z', a', L)$,
 - $(bz|, z'ba'|)$, falls $\delta(z, \sqcup) = (z', a', L)$,
 - $(z|, z'a'|)$, falls $\delta(z, \sqcup) = (z', a', N)$,
 - $(z|, a'z'|)$, falls $\delta(z, \sqcup) = (z', a', R)$,„Überführungsregeln“
- 4 für alle $e \in E, a \in \Gamma$: $(ae, e), (ea, e)$, „Löschregeln“
- 5 sowie für alle $e \in E$: $(e|), (|e)$. „Abschlussregeln“

Das Postsche Korrespondenzproblem

Beispiel

- Betrachte die 1-DTM $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ mit $Z = \{q_0, \dots, q_4\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, A, B, \sqcup\}$, $E = \{q_4\}$ und den Anweisungen

$$\begin{aligned} \delta: & q_0 a \rightarrow q_1 A R \quad (1) \quad q_1 a \rightarrow q_1 a R \quad (2) \quad q_1 B \rightarrow q_1 B R \quad (3) \quad q_1 b \rightarrow q_2 B L \quad (4) \\ & q_2 a \rightarrow q_2 a L \quad (5) \quad q_2 B \rightarrow q_2 B L \quad (6) \quad q_2 A \rightarrow q_0 A R \quad (7) \quad q_0 B \rightarrow q_3 B R \quad (8) \\ & q_3 B \rightarrow q_3 B R \quad (9) \quad q_3 \sqcup \rightarrow q_4 \sqcup N \quad (10) \end{aligned}$$

- M akzeptiert die Eingabe ab wie folgt:

$$q_0 ab \underset{(1)}{\vdash} Aq_1 b \underset{(4)}{\vdash} q_2 AB \underset{(7)}{\vdash} Aq_0 B \underset{(8)}{\vdash} ABq_3 \sqcup \underset{(10)}{\vdash} ABq_4 \sqcup$$

- Die MPCP-Instanz $f(ab)$ enthält für $u \in \Gamma$ die Wortpaare

Startregel	Kopierregeln	Löschregeln	Abschlussregel
$(\langle, \langle z_0 ab)$	$(u, u), (,)$	$(q_4 u, q_4), (u q_4, q_4)$	$(q_4 \rangle, \rangle)$

sowie folgende Überführungsregeln:

Das Postsche Korrespondenzproblem

Beispiel

Anweisung	zugehörige Überführungsregeln
$q_0 a \rightarrow q_1 A R$ (1)	$(q_0 a, A q_1)$
$q_1 a \rightarrow q_1 a R$ (2)	$(q_1 a, a q_1)$
$q_1 B \rightarrow q_1 B R$ (3)	$(q_1 B, B q_1)$
$q_1 b \rightarrow q_2 B L$ (4)	$(u q_1 b, q_2 u B), (q_1 b, q_2 \sqcup B)$
$q_2 a \rightarrow q_2 a L$ (5)	$(u q_2 a, q_2 u a), (q_2 a, q_2 \sqcup a)$
$q_2 B \rightarrow q_2 B L$ (6)	$(u q_2 B, q_2 u B), (q_2 B, q_2 \sqcup B)$
$q_2 A \rightarrow q_0 A R$ (7)	$(q_2 A, A q_0)$
$q_0 B \rightarrow q_3 B R$ (8)	$(q_0 B, B q_3)$
$q_3 B \rightarrow q_3 B R$ (9)	$(q_3 B, B q_3)$
$q_3 \sqcup \rightarrow q_4 \sqcup N$ (10)	$(q_3 \sqcup, q_4 \sqcup), (q_3 , q_4 \sqcup)$

- Die MPCP-Instanz $f(ab)$ enthält für $u \in \Gamma$ die Wortpaare

Startregel	Kopierregeln	Löschregeln	Abschlussregel
$(\langle, \langle z_0 ab)$	$(u, u), (,)$	$(q_4 u, q_4), (u q_4, q_4)$	(q_4 , \rangle)

sowie u.a. folgende **Überführungsregeln**:

$q_0 a \rightarrow q_1 AR$	(1)	$(q_0 a, Aq_1)$
$q_1 b \rightarrow q_2 BL$	(4)	$(uq_1 b, q_2 uB), (q_1 b, q_2 \sqcup B)$
$q_2 A \rightarrow q_0 AR$	(7)	$(q_2 A, Aq_0)$
$q_0 B \rightarrow q_3 BR$	(8)	$(q_0 B, Bq_3)$
$q_3 \sqcup \rightarrow q_4 \sqcup N$	(10)	$(q_3 \sqcup, q_4 \sqcup), (q_3 , q_4 \sqcup)$

- Der akzeptierenden Rechnung

$$q_0 ab \stackrel{(1)}{\vdash} Aq_1 b \stackrel{(4)}{\vdash} q_2 AB \stackrel{(7)}{\vdash} Aq_0 B \stackrel{(8)}{\vdash} ABq_3 \sqcup \stackrel{(10)}{\vdash} ABq_4 \sqcup$$

von $M(ab)$ entspricht dann das MPCP-Lösungswort

$$\begin{aligned} & \langle | q_0 ab | Aq_1 b | q_2 AB | Aq_0 B | ABq_3 | ABq_4 \sqcup | Aq_4 \sqcup | q_4 \sqcup | q_4 | \rangle \\ & \langle | q_0 ab | Aq_1 b | q_2 AB | Aq_0 B | ABq_3 | ABq_4 \sqcup | Aq_4 \sqcup | q_4 \sqcup | q_4 | \rangle \end{aligned}$$

Das Postsche Korrespondenzproblem

Beweis von $A \leq \text{MPCP}_{\Sigma'}$

- Nun lässt sich leicht aus einer akzeptierenden Rechnung

$$K_0 = z_0 w \vdash K_1 \vdash \dots \vdash K_t = uev$$

mit $e \in E$ und $u, v \in \Gamma^*$ eine MPCP-Lösung mit einem Lösungswort der Form

$$\langle |K_0|K_1|\dots|K_t|K_{t+1}|\dots|K_{t+|K_t|-1}| \rangle$$

angeben, wobei K_{t+i} aus K_t durch Löschen von i Zeichen in der Nachbarschaft von e entsteht.

- Umgekehrt lässt sich aus jeder MPCP-Lösung auch eine akzeptierende Rechnung von M bei Eingabe w gewinnen, womit

$$w \in L(M) \Leftrightarrow f(w) \in \text{MPCP}_{\Sigma'}$$

gezeigt ist.



Das Schnittproblem für CFL ist unentscheidbar

Das Schnittproblem für kontextfreie Grammatiken

Gegeben: Zwei kontextfreie Grammatiken G_1 und G_2 .

Gefragt: Ist $L(G_1) \cap L(G_2) \neq \emptyset$?

Satz

Das Schnittproblem für kontextfreie Grammatiken ist RE-vollständig.

Beweis

- Es ist leicht zu sehen, dass das Problem semi-entscheidbar ist.
- Wir reduzieren eine PCP-Instanz $I = \begin{pmatrix} x_1 \dots x_k \\ y_1 \dots y_k \end{pmatrix}$ auf ein Grammatikpaar (G_1, G_2) , so dass gilt: $I \in \text{PCP} \Leftrightarrow L(G_1) \cap L(G_2) \neq \emptyset$.
- Für $i = 1, 2$ sei $G_i = (\{S\}, \{a, b, 1, \dots, k\}, P_i, S)$ mit
$$P_1: S \rightarrow 1Sx_1, \dots, kSx_k, 1x_1, \dots, kx_k,$$
$$P_2: S \rightarrow 1Sy_1, \dots, kSy_k, 1y_1, \dots, ky_k.$$

Das Schnittproblem für CFL ist unentscheidbar

Beispiel

- Die PCP-Instanz

$$I = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} = \begin{pmatrix} a & aab & abbaa \\ aabba & ababb & aa \end{pmatrix}$$

wird auf das Grammatikpaar (G_1, G_2) mit folgenden Regeln reduziert:

$$P_1: S_1 \rightarrow 1Sa, 2S_{aab}, 3S_{abbaa}, \\ 1a, 2aab, 3abbaa,$$

$$P_2: S_2 \rightarrow 1S_{aabba}, 2S_{ababb}, 3S_{aa}, \\ 1aabba, 2ababb, 3aa.$$

- Der PCP-Lösung $\alpha = (1, 3, 2, 3)$ entspricht dann das Wort

$$\begin{aligned} 3231x_1x_3x_2x_3 &= 3231aabbaa aababbaa \\ &= 3231aabbaa aababbaa = 3231y_1y_3y_2y_3 \end{aligned}$$

im Schnitt $L(G_1) \cap L(G_2)$.

Das Schnittproblem für CFL ist unentscheidbar

Reduktion von PCP auf das Schnittproblem für CFL

- Für $i = 1, 2$ sei $G_i = (\{S\}, \{a, b, 1, \dots, k\}, P_i, S)$ mit

$$P_1: S \rightarrow 1Sx_1, \dots, kSx_k, 1x_1, \dots, kx_k,$$

$$P_2: S \rightarrow 1Sy_1, \dots, kSy_k, 1y_1, \dots, ky_k.$$

- Dann gilt

$$L(G_1) = \{i_n \dots i_1 x_{i_1} \dots x_{i_n} \mid 1 \leq n, 1 \leq i_1, \dots, i_n \leq k\},$$

$$L(G_2) = \{i_n \dots i_1 y_{i_1} \dots y_{i_n} \mid 1 \leq n, 1 \leq i_1, \dots, i_n \leq k\}.$$

- Somit ist $L(G_1) \cap L(G_2)$ die Sprache

$$\{i_n \dots i_1 x_{i_1} \dots x_{i_n} \mid 1 \leq n, x_{i_1} \dots x_{i_n} = y_{i_1} \dots y_{i_n}\}.$$

- Folglich ist $\alpha = (i_1, \dots, i_n)$ genau dann eine Lösung für I , wenn $i_n \dots i_1 x_{i_1} \dots x_{i_n} \in L(G_1) \cap L(G_2)$ ist.
- Also vermittelt $f : I \mapsto (G_1, G_2)$ eine Reduktion von PCP auf das Schnittproblem für CFL.

Das Schnitt- und das Inklusionsproblem für DCFL sind unentscheidbar

Korollar

- 1 Das Schnittproblem für DPDAs ist RE-vollständig.
- 2 Das Inklusionsproblem für DPDAs ist co-RE-vollständig.

Beweis.

- 1 Die kontextfreien Grammatiken G_1 und G_2 in obigem Beweis lassen sich leicht in äquivalente DPDAs M_1 und M_2 verwandeln (siehe Übungen).
- 2 Wir reduzieren das Komplement des Schnittproblems für DPDAs auf das Inklusionsproblem für DPDAs. Wegen

$$L_1 \cap L_2 = \emptyset \Leftrightarrow L_1 \subseteq \overline{L_2}.$$

berechnet die Funktion $f : (M_1, M_2) \mapsto (M_1, \overline{M_2})$ die gewünschte Reduktion. □

Weitere Unentscheidbarkeitsresultate für CFL

Korollar

Für kontextfreie Grammatiken sind folgende Probleme unentscheidbar:

- 1 Ist $L(G) = \Sigma^*$? (Ausschöpfungsproblem)
- 2 Ist $L(G_1) = L(G_2)$? (Äquivalenzproblem)
- 3 Ist G mehrdeutig? (Mehrdeutigkeitsproblem)

1 Das Ausschöpfungsproblem für kf. Grammatiken ist co-RE-vollständig

Wir reduzieren das Komplement des Schnittproblems für DPDAs auf das Ausschöpfungsproblem für kontextfreie Grammatiken. Es gilt

$$L_1 \cap L_2 = \emptyset \Leftrightarrow \overline{L_1} \cup \overline{L_2} = \Sigma^*.$$

Daher vermittelt die Funktion $f : (M_1, M_2) \mapsto G$, wobei G eine kontextfreie Grammatik mit

$$L(G) = \overline{L(M_1)} \cup \overline{L(M_2)}$$

ist, die gewünschte Reduktion. □

Weitere Unentscheidbarkeitsresultate für CFL

Korollar

Für kontextfreie Grammatiken sind folgende Probleme unentscheidbar:

- 1 Ist $L(G) = \Sigma^*$? (Ausschöpfungsproblem)
- 2 Ist $L(G_1) = L(G_2)$? (Äquivalenzproblem)
- 3 Ist G mehrdeutig? (Mehrdeutigkeitsproblem)

- 2 Das Äquivalenzproblem für kontextfreie Grammatiken ist co-RE-vollständig

Wir reduzieren das Ausschöpfungsproblem für CFL auf das Äquivalenzproblem für CFL.

Dies leistet beispielsweise die Reduktionsfunktion

$$f : G \mapsto (G, G_{all}),$$

wobei G_{all} eine kontextfreie Grammatik mit $L(G_{all}) = \Sigma^*$ ist. □

Weitere Unentscheidbarkeitsresultate für CFL

3 Das Mehrdeutigkeitsproblem ist RE-vollständig

- Wir reduzieren PCP auf das Mehrdeutigkeitsproblem.
- Betrachte die Reduktionsfunktion $f : \binom{x_1 \dots x_k}{y_1 \dots y_k} \mapsto G$ mit

$$G = (\{S, A, B\}, \{a, b, 1, \dots, k\}, P_1 \cup P_2 \cup \{S \rightarrow A, S \rightarrow B\}, S)$$

und den Regeln

$$P_1: A \rightarrow 1Ax_1, \dots, kAx_k, 1x_1, \dots, kx_k,$$

$$P_2: B \rightarrow 1By_1, \dots, kBy_k, 1y_1, \dots, ky_k.$$

- Da alle von A oder B ausgehenden Ableitungen eindeutig sind, ist G genau dann mehrdeutig, wenn es ein Wort $w \in L(G)$ gibt mit

$$S \Rightarrow A \Rightarrow^* w \quad \text{und} \quad S \Rightarrow B \Rightarrow^* w.$$

- Wie wir im Beweis der Unentscheidbarkeit des Schnittproblems für CFL gesehen haben, ist dies genau dann der Fall, wenn die PCP-Instanz $I = \binom{x_1 \dots x_k}{y_1 \dots y_k}$ eine PCP-Lösung hat. □

Ein Unentscheidbarkeitsresultat für DCSL

Das Leerheitsproblem für DLBAs

Gegeben: Ein DLBA M .

Gefragt: Ist $L(M) = \emptyset$?

Satz

Das Leerheitsproblem für DLBAs ist co-RE-vollständig.

Beweis.

- Wir reduzieren das Ausschöpfungsproblem für CFL auf das Leerheitsproblem für DLBAs.
- Eine kontextfreie Grammatik G lässt sich wie folgt in einen DLBA M mit $L(M) = \overline{L(G)}$ überführen (siehe Übungen):
 - Bestimme zunächst einen DLBA M mit $L(M) = L(G)$.
 - Konstruiere daraus einen DLBA \overline{M} mit $L(\overline{M}) = \overline{L(M)}$.
- Dann gilt $L(G) = \Sigma^* \Leftrightarrow L(M) = \emptyset$, d.h. die Funktion $f : G \mapsto \overline{M}$ berechnet die gewünschte Reduktion. □

Entscheidbare Probleme

Dagegen ist es nicht schwer,

- für eine kontextfreie Grammatik G zu entscheiden, ob mindestens ein Wort in G ableitbar ist (Leerheitsproblem für CFL), und
- für eine kontextsensitive Grammatik G und ein Wort x zu entscheiden, ob x in G ableitbar ist (Wortproblem für CSL).

Satz

- Das Leerheitsproblem für CFL ist entscheidbar.
- Das Wortproblem für CSL ist entscheidbar.

Beweis.

Siehe Übungen.



Überblick der gezeigten (Un-)Entscheidbarkeitsresultate

In folgender Tabelle fassen wir nochmals zusammen, wie schwierig die betrachteten Entscheidungsprobleme für die verschiedenen Stufen der Chomsky-Hierarchie sind.

	Wort- problem $x \in L?$	Leerheits- problem $L = \emptyset?$	Aus- schöpfung $L = \Sigma^*?$	Äquivalenz- problem $L_1 = L_2?$	Inklusions- problem $L_1 \subseteq L_2$	Schnitt- problem $L_1 \cap L_2 \neq \emptyset?$
REG	ja	ja	ja	ja	ja	ja
DCFL	ja	ja	ja	ja ^a	nein	nein
CFL	ja	ja	nein	nein	nein	nein
DCSL	ja	nein	nein	nein	nein	nein
CSL	ja	nein	nein	nein	nein	nein
RE	nein	nein	nein	nein	nein	nein

^aBewiesen in 1997 von Géraud Sénizergues (Univ. Bordeaux).

Die arithmetische Hierarchie

Frage

Wie kann man den Grad der Unentscheidbarkeit von unentscheidbaren Problemen messen?

Definition

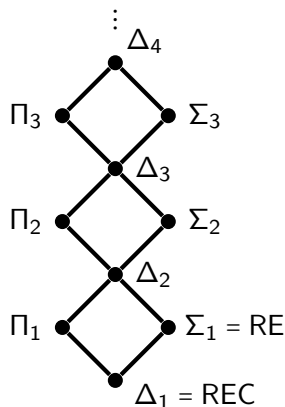
- Sei $A \subseteq \Sigma^*$ eine Sprache und \mathcal{K} eine Sprachklasse. Dann ist

$$\exists A = \{x \in \Sigma^* \mid \exists y \in \{0,1\}^* : x\#y \in A\}$$

und

$$\exists \mathcal{K} = \mathcal{K} \cup \{\exists A \mid A \in \mathcal{K}\}.$$

- Weiter sei $\Sigma_0 = \text{REC}$, sowie $\Pi_i = \text{co-}\Sigma_i$, $\Delta_i = \Sigma_i \cap \Pi_i$ und $\Sigma_{i+1} = \exists \Pi_i$ für $i \geq 0$.
- Die Sprachklassen $\Sigma_i, \Pi_i, \Delta_i$, $i \geq 0$, bilden die Stufen der **arithmetischen Hierarchie**.



Die arithmetische Hierarchie

Proposition

- $\Sigma_1 = \text{RE}$, $\Pi_1 = \text{co-RE}$.
- $\Delta_0 = \Sigma_0 = \Pi_0 = \Delta_1 = \text{REC}$.
- $\Sigma_i \cup \Pi_i \subseteq \Delta_{i+1}$ für $i \geq 0$.

Bemerkung

Mittels Diagonalisierung kann man zeigen, dass $\Sigma_i \neq \Pi_i$ für alle $i \geq 1$ gilt. Dies impliziert, dass die Inklusionen $\Sigma_i \subseteq \Delta_{i+1} \subseteq \Sigma_{i+1}$ und $\Pi_i \subseteq \Delta_{i+1} \subseteq \Pi_{i+1}$ für alle $i \geq 1$ echt sind.

Die arithmetische Hierarchie

Beweis von $\Sigma_1 \subseteq RE$

Sei $A \in \Sigma_1$ und sei $A = \exists B$ für eine Sprache $B \in REC$.
Dann wird A von einer DTM akzeptiert, die bei Eingabe x systematisch nach einem String $y \in \{0,1\}^*$ mit $x\#y \in B$ sucht. Folglich ist $A \in RE$. \square

Beweis von $RE \subseteq \Sigma_1$

Sei $A \subseteq \Sigma^*$ eine Sprache in RE und sei $A = L(M)$ für eine DTM M .
Dann ist $A = \exists B$ für die Sprache

$$B = \{x\#y \mid x \in \Sigma^* \text{ und } y \text{ kodiert eine akz. Rechnung von } M(x)\}.$$

Da B entscheidbar ist, folgt $A \in \exists REC$. \square

Beweis von $\Pi_1 = \text{co-RE}$

Folgt wegen $\Pi_1 = \text{co-}\Sigma_1$ direkt aus $\Sigma_1 = RE$. \square

Beweis von $\Delta_0 = \Sigma_0 = \Pi_0 = \Delta_1 = REC$

Klar, da $REC = \text{co-REC}$ und $REC = RE \cap \text{co-RE}$. \square

Die arithmetische Hierarchie

Beweis von $\Sigma_i \subseteq \Pi_{i+1}$

Klar, da $\Sigma_i \subseteq \exists \Sigma_i = \Pi_{i+1}$. □

Beweis von $\Sigma_i \subseteq \Sigma_{i+1}$

Dies zeigen wir induktiv.

$i = 0$: Es gilt $\Sigma_0 = \text{REC} \subseteq \text{RE} = \Sigma_1$.

$i - 1 \rightsquigarrow i$: Nach IV gilt $\Sigma_{i-1} \subseteq \Sigma_i$. Folglich ist $\Pi_{i-1} \subseteq \Pi_i$. Dies wiederum impliziert $\exists \Pi_{i-1} \subseteq \exists \Pi_i$, also $\Sigma_i \subseteq \Sigma_{i+1}$. □

Beweis von $\Sigma_i \subseteq \Delta_{i+1}$

Klar, da $\Delta_{i+1} = \Sigma_{i+1} \cap \Pi_{i+1}$. □

Beweis von $\Pi_i \subseteq \Delta_{i+1}$

Klar, da $\Pi_i = \text{co-}\Sigma_i \subseteq \text{co-}\Delta_{i+1} = \Delta_{i+1}$. □

Überblick der gezeigten (Un-)Entscheidbarkeitsresultate

Die betrachteten Entscheidungsprobleme für die verschiedenen Stufen der Chomsky-Hierarchie lassen sich nun wie folgt in die arithmetische Hierarchie einordnen.

	Wort- problem $x \in L?$	Leerheits- problem $L = \emptyset?$	Aus- schöpfung $L = \Sigma^*$?	Äquivalenz- problem $L_1 = L_2?$	Inklusions- problem $L_1 \subseteq L_2$	Schnitt- problem $L_1 \cap L_2 \neq \emptyset?$
REG	REC	REC	REC	REC	REC	REC
DCFL	REC	REC	REC	REC	co-RE	RE
CFL	REC	REC	co-RE	co-RE	co-RE	RE
DCSL	REC	co-RE	co-RE	co-RE	co-RE	RE
CSL	REC	co-RE	co-RE	co-RE	co-RE	RE
RE	RE	co-RE	Π_2	Π_2	Π_2	RE

Tatsächlich sind alle betrachteten Entscheidungsprobleme sogar vollständig für die angegebenen Sprachklassen.