universität innsbruck

Einführung in die Theoretische Informatik

Martin Avanzini Christian Dalvit Jamie Hochrainer
Georg Moser Johannes Niederhauser Jonas Schöpf

`https://tcs-informatik.uibk.ac.at`

universität
innsbruck

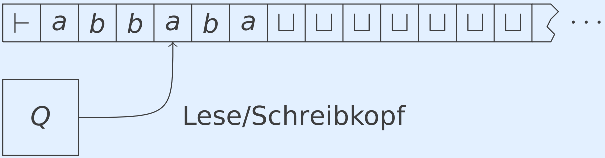
Frohes Neues!

universität
innsbruck

Zusammenfassung

Definition (informell)

deterministische, einbändige Turingmaschine (TM):



- Eine TM verwendet ein einseitig unendliches Band als Speicher
- Zu Beginn der Berechnung steht die Eingabe auf dem Band
- Der Speicher wird mit einem Lese/Schreibkopf gelesen oder beschrieben
- Das Verhalten der TM wird durch die endliche Kontrolle Q kontrolliert

Unentscheidbarkeit

Satz

Es kann niemals ein Testprogramm für „hello, world“-Programme geben

Definition (informell)

ein Problem, das nicht algorithmisch lösbar ist, heißt unentscheidbar

Satz

die folgenden Probleme sind unentscheidbar:

1. *das Halteproblem*
2. *das Postsche Korrespondenzproblem*
3. *ist eine beliebige kontextfreie Grammatik eindeutig*
4. *...*

Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Formales Beweisen, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

Algebraische Strukturen, Boolesche Algebra, Universelle Algebra

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Chomsky-Hierarchie, Reguläre Sprachen, Kontextfreie Sprachen, Anwendungen von formalen Sprachen

Einführung in die Berechenbarkeitstheorie und Komplexitätstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen, Komplexitätstheorie

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare

Turing Maschinen Berechenbarkeit

Beispiel

Wir beschreiben informell eine TM M, die zwei Zahlen x_1 und x_2 als Eingabe bekommt; M soll $x_1 - x_2$ berechnen und diese Differenz verdoppeln

Algorithmus

- wir kodieren die Eingabe x_1, x_2 durch Wörter über dem Alphabet $\{\sqcap\}$ der Länge x_1 bzw. x_2
- die beiden Eingaben werden durch Trennsymbol $\Box$ getrennt
- Subtraktion funktioniert, indem wir sukzessive für alle $\sqcap$ rechts von $\Box$ ein $\sqcap$ links mit $\sqcup$ überschreiben
- Verdopplung funktioniert indem wir für jedes übrige $\sqcap$, $\sqcap\sqcap$ schreiben

Registermaschinen

Definition

Eine Registermaschine (RM) R ist ein Paar $R = ((x_i)_{1 \leqslant i \leqslant n}, \mathrm{P})$ sodass

1. $(x_i)_{1 \leqslant i \leqslant n}$ eine Sequenz von n Registern x_i, die natürliche Zahlen beinhalten
2. P ein Programm

Programme sind endliche Folgen von Befehlen und sind induktiv definiert:

1. Für jedes Register x_i sind die folgenden Instruktionen sowohl Befehle wie Programme: $x_i := x_i + 1$ und $x_i := x_i - 1$
2. wenn P_1, P_2 Programme sind, dann ist $P_1; P_2$ ein Programm
3. wenn P_1 ein Programm und x_i ein Register, dann ist

   $$\textsf{while } x_i \neq 0 \textsf{ do } P_1 \textsf{ end}$$

   sowohl ein Befehl als auch ein Programm

Definition (Semantik von Registermaschinen)

1. Zu Beginn der Berechnung steht die Eingabe (als natürliche Zahlen) in den Registern
2. Die Befehle
   - $x_i := x_i + 1$
   - $x_i := x_i - 1$

   bedeuten, dass der Inhalt des Register x_i entweder um 1 erhöht oder vermindert wird
3. $P_1; P_2$ bedeutet, dass zunächst das Programm P_1 und dann das Programm P_2 ausgeführt wird
4. Der Befehl (und das Programm)

   $$\text{while } x_i \neq 0 \text{ do } P_1 \text{ end}$$

   bedeutet, der Schleifenrumpf P_1 wird ausgeführt, bis die Bedingung $x_i \neq 0$ falsch ist

Beispiel

Sei $R = ((x_i)_{1 \leqslant i \leqslant 5}, \mathrm{P})$ eine RM mit dem folgenden Programm:

Zuweisung $x_i := x_j$

$$
\begin{array}{l}
\text{while } x_i \neq 0 \text{ do} \\
\quad x_i := x_i - 1 \\
\text{end;} \\
\text{while } x_k \neq 0 \text{ do} \\
\quad x_k := x_k - 1 \\
\text{end}
\end{array}
\qquad
\begin{array}{l}
\text{while } x_j \neq 0 \text{ do} \\
\quad x_i := x_i + 1; \\
\quad x_j := x_j - 1; \\
\quad x_k := x_k + 1 \\
\text{end;} \\
\text{while } x_k \neq 0 \text{ do} \\
\quad x_j := x_j + 1; \\
\quad x_k := x_k - 1 \\
\text{end}
\end{array}
$$

Multiplikation

$$
\begin{array}{l}
x_3 := 0; \\
\text{while } x_1 \neq 0 \text{ do} \\
\quad x_1 := x_1 - 1; \\
\quad x_4 := x_2; \\
\quad \text{while } x_2 \neq 0 \text{ do} \\
\quad\quad x_2 := x_2 - 1; \\
\quad\quad x_3 := x_3 + 1 \\
\quad \text{end;} \\
\quad x_2 := x_4 \\
\text{end}
\end{array}
$$

Bei Eingabe $(m, n, 0, 0, 0)$ berechnet R $(0, n, m \times n, n, 0)$

Berechenbarkeit mit einer RM

Definition

- sei $R = ((x_i)_{1 \leqslant i \leqslant n}, \mathrm{P})$ eine RM
- eine partielle Funktion $f \colon \mathbb{N}^k \to \mathbb{N}$, heißt R-berechenbar, wenn

$$f(n_1, \dots, n_k) = m \quad \text{gdw.} \quad \begin{array}{l} R \text{ startet mit } n_i \text{ in den Registern } x_i \text{ und en-} \\ \text{det mit } m \text{ im Register } x_{k+1} \text{ (und Eingaben} \\ n_i \text{ in den Registern } x_i\text{)} \end{array}$$

- Eine partielle Funktion $f \colon \mathbb{N}^k \to \mathbb{N}$ heißt berechenbar auf einer RM, wenn eine RM R existiert, sodass f R-berechenbar.

Satz

Jede partielle Funktion $f \colon \mathbb{N}^k \to \mathbb{N}$, die berechenbar auf einer RM ist, ist auf einer TM berechenbar und umgekehrt

Church-Turing-These („Naturgesetz“ der Informatik)

Jedes algorithmisch lösbare Problem ist auch mit Hilfe einer Turingmaschine lösbar.

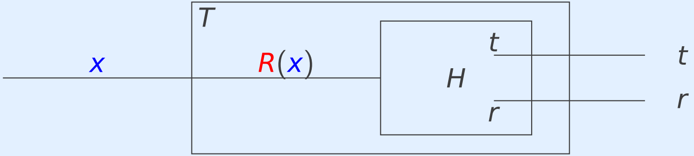
Definition (Turingreduktion)

angenommen

- L, M Sprachen über Σ
- $L \leqslant_T M$ mit $R\colon \Sigma^* \to \Sigma^*$
- die Reduktion R wird von TM T berechnet, sodass gilt

$$x \in L \Leftrightarrow R(x) \in M$$

Entscheidbarkeit von L, durch Entscheider H von M



Beispiel

Seien

$$L = \{x \in \{a,b\}^* \mid |x| \text{ ist gerade}\}$$
$$M = \{x \in \{a,b\}^* \mid x \text{ ist ein Palindrom gerader Länge}\}$$

dann gilt $L \leqslant_T M$

Reduktion

Wir geben eine (TM) berechenbare Abbildung $R\colon \{a,b\}^* \to \{a,b\}^*$ an, sodass $x \in L \Leftrightarrow R(x) \in M$:

- definiere R', sodass $R'(a) := a$ und $R'(b) := a$
- definiere R als Erweiterung von R' auf Wörter
- R ist eine Stringfunktion, die ein Wort aus $\{a,b\}^n$ in das Wort a^n umwandelt
- Genau dann wenn n gerade ist, ist a^n ein Palindrom gerader Länge

Tabelle für $x \in L$ gdw $R(x) \in M$

$x \in L$	x	$R(x)$	$R(x) \in M$
$\checkmark$	ϵ	ϵ	$\checkmark$
$\times$	a	a	$\times$
$\times$	b	a	$\times$
$\checkmark$	aa	aa	$\checkmark$
$\checkmark$	ab	aa	$\checkmark$
$\checkmark$	ba	aa	$\checkmark$
$\checkmark$	bb	aa	$\checkmark$
$\times$	aaa	aaa	$\times$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

wobei

$L = \{x \in \{a,b\}^* \mid |x| \text{ ist gerade}\}$ $\quad M = \{x \in \{a,b\}^* \mid x \text{ ist ein Palindrom gerader Länge}\}$

Anwendungen von Reduktionen

Lemma

wenn $L \leqslant_T M$ und M rekursiv, dann ist L rekursiv

Unentscheidbarkeit

Unentscheidbarkeit eines Problems zeigt man mittels Reduktion von einem unentscheidbares Problem (zum Beispiel das Halteproblem) auf das betrachtete Problem

Satz

es kann kein Testprogramm für “hello, world” Programme geben

Beweis.

$$\text{HP} \leqslant_T \text{“hello, world” Programme}$$

∎

Satz

Sei Σ ein Alphabet und $L \subseteq \Sigma^*$ rekursiv; dann ist $\sim L$ rekursiv

Beweis.

Da L rekursiv ist, gibt es eine totale TM M mit $L = \mathrm{L}(M)$. Wir definieren eine TM M', wobei der akzeptierende und der verwerfende Zustand von M vertauscht werden. Weil M total ist, ist auch M' total. Somit akzeptiert M' ein Wort genau dann, wenn M es verwirft und es folgt $\sim L = \mathrm{L}(M')$, d.h. $\sim L$ ist rekursiv. ∎

Satz

Jede rekursive Menge ist rekursiv aufzählbar. Andererseits ist nicht jede rekursiv aufzählbare Menge rekursiv.

Beweis.

Der erste Teil des Satzes ist eine Konsequenz der Definitionen; der zweite Teil wird in „Diskrete Strukturen“ bewiesen werden. ∎

Satz

Wenn L und $\sim L$ rekursiv aufzählbar sind, dann ist L rekursiv.

Beweis.

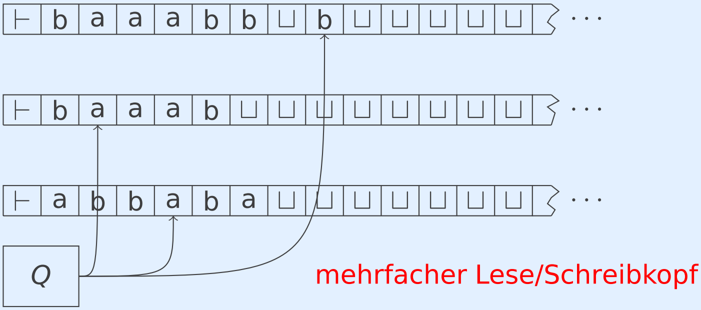
- $\exists$ TM M_1, M_2 mit $L = \mathrm{L}(M_1)$ und $\sim(L) = \mathrm{L}(M_2)$
- definiere TM M', sodass das Band zwei Hälften hat

b	$\hat{b}$	a	b	a	a	a	a	b	a	a	a	$\dots$
c	c	c	d	d	d	c	$\hat{c}$	d	c	d	c	

- M_1 wird auf der oberen und M_2 auf der unteren Hälfte simuliert
- wenn M_1 x akzeptiert, M' akzeptiert x
- wenn M_2 x akzeptiert, M' verwirft x ■

Definition (informell)

Erweiterung um mehrere Bänder und Lese/Schreibköpfe:



Definition

$$\delta \colon Q \times \Gamma^3 \to Q \times \Gamma^3 \times \{\mathsf{L}, \mathsf{R}\}^3$$

Satz

Sei M eine k-bändige TM. Dann existiert eine (einbändige) TM M', sodass $\mathrm{L}(M) = \mathrm{L}(M')$

Beweisskizze.

- wir simulieren die Bänder übereinander, oBdA sei $k = 2$
- wir erweitern das Alphabet von M'

$$\begin{array}{|c|}\hline \mathrm{a} \\ \hline \mathrm{c} \\ \hline\end{array} \qquad \begin{array}{|c|}\hline \hat{\mathrm{a}} \\ \hline \mathrm{c} \\ \hline\end{array} \qquad \begin{array}{|c|}\hline \mathrm{a} \\ \hline \hat{\mathrm{c}} \\ \hline\end{array} \qquad \begin{array}{|c|}\hline \hat{\mathrm{a}} \\ \hline \hat{\mathrm{c}} \\ \hline\end{array}$$

- Band von M' kann folgende Gestalt haben:

b	$\hat{b}$	a	b	a	a	a	a	b	a	a	a	...
c	c	c	d	d	d	c	$\hat{c}$	d	c	d	c	

- alle Bänder von M sind nun repräsentiert und die Leseköpfe werden durch die Zusatzmarkierung $\hat{}$ ausgedrückt

$\blacksquare$

Berechenbarkeitstheorie, graphisch

"I can't find an efficient algorithm, because no such algorithm is possible!"