Grundlagen der Theoretischen Informatik

Turingmaschinen und rekursiv aufzählbare Sprachen (IV)

23.07.2021

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Übersicht

- Determinierte Turing-Maschinen (DTMs)
- Varianten von Turing-Maschinen
- Indeterminierte Turing-Maschinen (NTMs)
- Universelle determinierte Turing-Maschinen
- Entscheidbar/Aufzählbar
- Determinierte Turing-Maschinen entsprechen Typ 0
- Unentscheidbarkeit

Zentrale Fragestellung

Welche Funktionen sind durch einen Algorithmus berechenbar?

bzw.

Welche Probleme sind durch einen Algorithmus entscheidbar?

Die Motivation, die Entscheidbarkeit und Unentscheidbarkeit zu studieren, stammt ursprünglich von dem Mathematiker David Hilbert:

Anfang des 20. Jahrhunderts formulierte er einen Forschungsplan, dessen Ziel die Entwicklung eines Formalismus war, mit dem man alle mathematischen Probleme lösen konnte.

Zentrale Fragestellung

Welche Funktionen sind durch einen Computer berechenbar?

bzw.

Welche Probleme kann ein Computer entscheiden?

Um diese Fragen in einem mathematisch exakten Sinne klären zu können, müssen wir klären was eigentlich ein Computer ist.

Rechnermodell: Turingmaschinen

Alan Turing

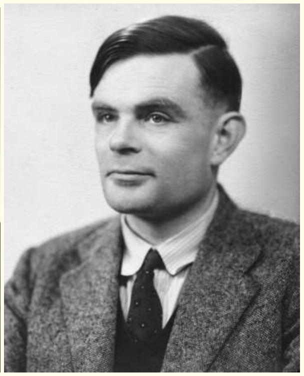
Church-Turing-These

Es wurde festgestellt, dass alle vernünftigen Modelle zur Spezifikation der algorithmischen Lösbarkeit äquivalent zu Turingmaschinen sind.

mehr darüber: Vertiefung Theoretische Informatik

Dies führte zu der Formulierung der sogenannten Church-Turing-These:

Church-Turing-These
Die Klasse der Turing-berechenbaren Funktionen stimmt mit der Klasse der "intuitiv berechenbaren" Funktionen überein.



Beispiel einer nicht berechenbaren Funktion

Definition (Busy-Beaver-Funktion)

Die Funktion $BB : \mathbb{N} \to \mathbb{N}$ sei wie folgt definiert

$n \mapsto BB(n) :=$ die maximale Anzahl an Einsen, die eine **haltende** DTM mit maximal n Zuständen auf einem leeren Band erzeugen kann

Beispiel einer nicht berechenbaren Funktion

Definition (Busy-Beaver-Funktion)

Die Funktion $BB : \mathbb{N} \to \mathbb{N}$ sei wie folgt definiert

$n \mapsto BB(n) :=$ die maximale Anzahl an Einsen, die eine **haltende** DTM mit maximal n Zuständen auf einem leeren Band erzeugen kann

BB wächst extrem schnell

Exakte Werte von $BB(n)$ für $n \geq 5$ nicht bekannt.

- $BB(4) \geq 4098$
- $BB(5) \geq 1,29 * 10^{865}$

Beispiel einer nicht berechenbaren Funktion

Theorem

BB wächst zu stark um berechenbar zu sein:
Es gibt keine DTM, die BB berechnet.

Beweis (erster Teil)

Man kann immer mindestens ein $|$ mehr erzeugen, wenn man einen weiteren Zustand zur Verfügung hat:

- man benennt den Haltezustand um in q_{neu} und geht in den richtigen Haltezustand h nur, wenn man in q_{neu} ein Blank $\#$ gelesen hat. Zusätzlich ersetzt man das Blank durch $|$.
- Wenn man ein $|$ liest, geht man nach rechts und bleibt in q_{neu}.

Damit haben wir bewiesen:

BB wächst streng monoton.

Beispiel einer nicht berechenbaren Funktion

Beweis (zweiter Teil)

Angenommen, es gäbe eine DTM $\mathcal{M}_{BB}$, die BB berechnet.
Sie habe n_0 Zustände.

Wir betrachten folgende zusammengesetzte Maschine:

- zuerst schreibt sie m Einsen auf das leere Band
- dann führt sie $\mathcal{M}_{BB}$ aus

Diese Maschine kommt mit $n_0 + \lfloor \frac{m}{2} \rfloor + 10$ Zuständen aus.

Sei nun m so groß, dass

$$m > n_0 + \left\lfloor \frac{m}{2} \right\rfloor + 10$$

Dann schreibt die neue Maschine $BB(m)$ Einsen auf das Band und arbeitet mit streng weniger als m Zuständen: Widerspruch. $\square$

Zur Erinnerung

Beispiel 4 (Print n)

Für jedes $n \in \mathbb{N}$ konstruieren wir eine Maschine, die genau n Einsen auf das leere Band schreibt (**mit möglichst wenig Zuständen**):

1. schreibe $\lfloor \frac{n}{2} \rfloor$ viele Einsen auf das Band
   (höchstens $\lfloor \frac{n}{2} \rfloor$ Zustände)
2. kopiere diesen String
   (8 Zustände)
3. ersetze das trennende $\#$ durch eine 1
4. falls n gerade ist, ersetzen die letzte 1 durch $\#$ (2 neue Zustände)

Insgesamt können wir n Einsen mit höchstens $\lfloor \frac{n}{2} \rfloor + 10$ Zuständen konstruieren

Beispiel einer nicht berechenbaren Funktion

Beweis (zweiter Teil)

Angenommen, es gäbe eine DTM $\mathcal{M}_{BB}$, die BB berechnet.
Sie habe n_0 Zustände.

Wir betrachten folgende zusammengesetzten Maschine:

- zuerst schreibt sie m Einsen auf das leere Band
- dann führt sie $\mathcal{M}_{BB}$ aus

Diese Maschine kommt mit $n_0 + \lfloor \frac{m}{2} \rfloor + 10$ Zuständen aus.

Sei nun m so groß, dass

$$m > n_0 + \left\lfloor \frac{m}{2} \right\rfloor + 10$$

Dann schreibt die neue Maschine $BB(m)$ Einsen auf das Band und arbeitet mit streng weniger als m Zuständen: Widerspruch. $\square$

Gödelisierung von DTMs

Definition (Gödelnummern von DTMs)

DTMs werden als Gödelzahlen kodiert:

- DTMs können als Gödelwörter dargestellt werden.
  $\mathcal{M} \mapsto a_{i_1} \dots a_{i_n}$
- Die Buchstaben der Gödelwörter können in Ziffern kodiert werden, um Gödelnummern zu bekommen.
  $$a_{i_1} \dots a_{i_n} \mapsto p_1^{i_1} \dots p_n^{i_n} \in \mathbb{N} \qquad p_i\text{: } i\text{-te Primzahl.}$$
- Notation: $\hat{g}(\mathcal{M})$ für die Gödelnummer der DTM $\mathcal{M}$.

Gödelisierung von DTMs

Definition (Jede Zahl ist Gödelnummer)

Jede natürliche Zahl n soll Gödelnummer einer DTM $\mathcal{M}_n$ sein.

Wir definieren

$$\mathcal{M}_n := \begin{cases} \mathcal{M}, & \text{falls } \hat{g}(\mathcal{M}) = n \\ \mathcal{M}_{halt} & \text{falls } \nexists \mathcal{M} \;\; \hat{g}(\mathcal{M}) = n \end{cases}$$

$\mathcal{M}_{halt}$ ist eine TM, die sofort anhält und weiter nichts tut.

Halteproblem

Definition [Allgemeines Halteproblem]

Das **allgemeine Halteproblem** ist die Frage, ob die DTM mit Gödelnummer n bei Eingabe i hält.

Es entspricht der Sprache

$$\mathcal{H}_{allg} := \{\langle n, i\rangle \mid \mathcal{M}_n \text{ hält bei Eingabe } i\}.$$

Halteproblem

Definition [Spezielles Halteproblem]

Das **spezielle Halteproblem** ist die Frage, ob die DTM mit Gödelnummer n bei Eingabe n hält.

Es entspricht der Sprache

$$\mathcal{H} := \{n \mid \mathcal{M}_n \text{ hält bei Eingabe } n\}.$$

Halteproblem

Definition [Null-Halteproblem]
Das **Null-Halteproblem** ist die Frage, ob die DTM mit Gödelnummer n bei leerer Eingabe hält.

Es entspricht der Sprache

$$\mathcal{H}_0 := \{n \mid \mathcal{M}_n \text{ hält bei leerer Eingabe}\}$$

Manchmal auch:
"bei Eingabe 0" anstatt "bei leerer Eingabe".

Leerheitsproblem

Definition [Leerheitsproblem]
Das **Leerheitsproblem** ist die Frage, ob die DTM mit Gödelnummer n bei keiner Eingabe aus Σ^* hält.

Es entspricht der Sprache

$$\mathcal{E} := \{n \mid \mathcal{M}_n \text{ hält bei keiner Eingabe aus } \Sigma^*\}.$$

Totalitätsproblem

Definition [Totalitätsproblem]

Das **Totalitätsproblem** ist die Frage, ob die DTM mit Gödelnummer n bei jeder Eingabe aus Σ^* hält.

Es entspricht der Sprache

$$\mathcal{T} := \{n \mid \mathcal{M}_n \text{ hält bei jeder Eingabe aus } \Sigma^*\}.$$

Gleichheitsproblem

Definition [Gleichheitsproblem]

Das **Gleichheitsproblem** ist die Frage, ob die DTM mit Gödelnummer n die gleiche Sprache über Σ akzeptiert wie die DTM mit Gödelnummer m.

Es entspricht der Sprache

$$\mathcal{E}q := \{\langle n, m\rangle \mid \quad \mathcal{M}_n \text{ akzeptiert die gleiche Sprache über } \Sigma \text{ wie } \mathcal{M}_m \}.$$

Entscheidbarkeitsproblem

Definition [Entscheidbarkeitsproblem]
Das **Entscheidbarkeitsproblem** ist die Frage, ob die DTM mit Gödelnummer n eine entscheidbare Sprache über Σ akzeptiert.

Es entspricht der Sprache

$$\mathcal{E}nt := \{n \mid \quad \mathcal{M}_n \text{ akzeptiert eine entscheidbare Sprache über } \Sigma\}.$$

Unentscheidbarkeit des Halteproblems

Theorem (Halteproblem ist unentscheidbar)
Das spezielle Halteproblem $\mathcal{H} := \{n \mid \mathcal{M}_n \text{ hält bei Eingabe } n\}$ ist unentscheidbar.

Unentscheidbarkeit des Halteproblems

Theorem (Halteproblem ist unentscheidbar)
Das spezielle Halteproblem $\mathcal{H} := \{n \mid \mathcal{M}_n \text{ hält bei Eingabe } n\}$ ist unentscheidbar.

Beweis (A. Turing)

Beweis durch Widerspruch mit einem **Diagonalisierungsargument**.

Angenommen, es gäbe eine DTM $\mathcal{M}_{\mathcal{H}}$, die das spezielle Halteproblem $\mathcal{H}$ entscheidet.

Konstruiere eine neue Maschine $\mathcal{M}'$ aus $\mathcal{M}_{\mathcal{H}}$:

- Wenn $\mathcal{M}_{\mathcal{H}}$ „Y“ antwortet, geht $\mathcal{M}'$ in eine Endlosschleife (terminiert nicht).
- Wenn $\mathcal{M}_{\mathcal{H}}$ „N“ antwortet, terminiert $\mathcal{M}'$.

Die neue Maschine habe Gödelnummer n (also: $\mathcal{M}_n = \mathcal{M}'$)

Unentscheidbarkeit des Halteproblems

Beweis (A. Turing), Forts.

Was macht $\mathcal{M}_n$ bei Eingabe n?

- Falls $\mathcal{M}_n$ bei Eingabe n terminiert,
  dann antwortet $\mathcal{M}_{\mathcal{H}}$ auf Eingabe n mit „Y“,
  dann terminiert $\mathcal{M}_n$ auf Eingabe von n nicht
  Widerspruch!

- Falls $\mathcal{M}_n$ bei Eingabe n nicht terminiert,
  dann antwortet $\mathcal{M}_{\mathcal{H}}$ auf Eingabe n mit „N“,
  dann terminiert $\mathcal{M}_n$ auf Eingabe von n
  Widerspruch!

Akzeptierbarkeit des Halteproblems

Theorem (Akzeptierbarkeit von $\mathcal{H}$)

Das spezielle Halteproblem $\mathcal{H}$ ist aufzählbar. D. h. die Sprache

$$\mathcal{H} = \{n \mid \mathcal{M}_n \text{ hält bei Eingabe } n\}$$

ist akzeptierbar.

Akzeptierbarkeit des Halteproblems

Theorem (Akzeptierbarkeit von $\mathcal{H}$)
Das spezielle Halteproblem $\mathcal{H}$ ist aufzählbar. D. h. die Sprache

$$\mathcal{H} = \{ n \mid \ \mathcal{M}_n \text{ hält bei Eingabe } n \ \}$$

ist akzeptierbar.

Beweis:

Idee:

Akzeptieren durch Simulation von $\mathcal{M}_n$ mit Hilfe der universellen DTM.

Akzeptierbarkeit des Halteproblems

Theorem (Akzeptierbarkeit von $\mathcal{H}$)
Das spezielle Halteproblem $\mathcal{H}$ ist aufzählbar. D. h. die Sprache

$$\mathcal{H} = \{n \mid \ \mathcal{M}_n \text{ hält bei Eingabe } n \ \}$$

ist akzeptierbar.

Beweis:

Sei $\mathcal{M}_{\text{prep}}$ die TM, die die Eingabe auf dem Arbeitsband in die Form bringt, die $\mathcal{U}$ fordert.

Sei $\mathcal{M}_{\mathcal{H}} := \mathcal{M}_{\text{prep}}\mathcal{U}$

$s, \#n\underline{\#} \vdash^*_{\mathcal{M}_{\text{prep}}} q, \#n\#w_{\delta,\mathcal{M}_n}\underline{\#}$ $\quad$ $\mathcal{U}$ simuliert dann die Arbeit von $\mathcal{M}_n$ bei Input n.

$$\begin{array}{lll} \mathcal{M}_{\mathcal{H}} := \mathcal{M}_{\text{prep}}\mathcal{U} \text{ hält bei Input } n & \text{genau dann, wenn} & \mathcal{M}_n \text{ bei Input } n \text{ hält} \\ & \text{genau dann, wenn} & n \in \mathcal{H}. \end{array}$$

Halteproblem

Theorem (Halteproblem ist unentscheidbar)
Das spezielle Halteproblem $\mathcal{H} := \{n \mid \mathcal{M}_n \text{ hält bei Eingabe } n\}$ ist unentscheidbar.

Theorem (Akzeptierbarkeit von $\mathcal{H}$)
Das spezielle Halteproblem $\mathcal{H}$ ist aufzählbar. D. h. die Sprache

$$\mathcal{H} = \{n \mid \ \mathcal{M}_n \text{ hält bei Eingabe } n \ \}$$

ist akzeptierbar.

Korollar Das Komplement von $\mathcal{H}$ ist nicht aufzählbar.

Reduktion von Problemen

Wie zeigt man, dass ein Problem unentscheidbar ist?

Reduktion (informell)

Wir geben eine **totale, berechenbare Funktion** f an, die

- eine Instanz p_1 von P_1
- in eine Instanz p_2 von P_2 umwandelt,
- und zwar so, dass die Antwort zu p_1 „ja" ist gdw die Antwort zu p_2 „ja" ist.

Wenn P_1 unentscheidbar ist, dann ist auch P_2 unentscheidbar.

Reduktion von Problemen

Definition

Seien L_1, L_2 Sprachen über $\mathbb{N}$.

L_1 wird auf L_2 reduziert,

$$L_1 \preceq L_2$$

<u>gdw</u>

es gibt eine TM-berechenbare Funktion $f : \mathbb{N} \to \mathbb{N}$, so dass gilt:

$$\forall n \in \mathbb{N} \quad \big(n \in L_1 \ \underline{\text{gdw}}\ f(n) \in L_2\big).$$

Reduktion von Problemen

Lemma

Ist $L_1 \preceq L_2$, und ist L_1 **unentscheidbar**, so ist auch L_2 **unentscheidbar**.

Reduktion von Problemen

Lemma

Ist $L_1 \preceq L_2$, und ist L_1 **unentscheidbar**, so ist auch L_2 **unentscheidbar**.

Beweis

- Angenommen, L_2 ist entscheidbar.
- Sei $\mathcal{M}_2$ eine Turing-Maschine, die L_2 entscheidet.
- Wegen $L_1 \preceq L_2$ gibt es eine Funktion $f{:}\mathbb{N}{\to}\mathbb{N} \in TM$ mit $n \in L_1$ <u>gdw</u> $f(n) \in L_2$.
- Sei $\mathcal{M}_f$ eine DTM, die f berechnet.
- Dann kann man daraus die Maschine $\mathcal{M}_1 := \mathcal{M}_f\mathcal{M}_2$ konstruieren, für die gilt:
  - $\mathcal{M}_1$, gestartet mit Input n, hält mit $h, \#Y\underline{\#}$, falls $f(n) \in L_2$, d.h. wenn $n \in L_1$ ist.
  - $\mathcal{M}_1$, gestartet mit Input n, hält mit $h, \#N\underline{\#}$, falls $f(n) \notin L_2$, d.h. wenn $n \notin L_1$ ist.
- Die Maschine $\mathcal{M}_1$ entscheidet also L_1, ein Widerspruch.

Unentscheidbarkeit

Theorem [Unentscheidbarkeit von $\mathcal{H}_0$].
Das Null-Halteproblem

$$\mathcal{H}_0 = \{n \mid \; \mathcal{M}_n \text{ hält bei Eingabe } 0\}$$

ist unentscheidbar.

Beweis: (Idee) Reduktion des speziellen Halteproblems auf das Null-Halteproblem.

Finde f TM-berechenbar s.d. $n \in \mathcal{H}$ gdw. $f(n) \in \mathcal{H}_0$.

i.e. s.d. (M_n hält bei Eingabe n) gdw. ($M_{f(n)}$ hält bei Eingabe 0)

Unentscheidbarkeit

Beweis: Gegeben eine TM $\mathcal{M}_n$.

Kombiniere diese mit einer DTM, die n aufs Band schreibt.

$f(n)$ sei definiert als die Gödelnummer dieser neuen Maschine.

$\mathcal{M}_{f(n)}$ terminiert auf Eingabe von 0
<u>gdw</u>
$\mathcal{M}_n$ terminiert auf Eingabe von n

Damit:
Reduktion des speziellen Halteproblems auf das Null-Halteproblem.
(f ist total und berechenbar!)

Also:
Unentscheidbarkeit des Null-Halteproblems folgt aus
Unentscheidbarkeit des speziellen Halteproblems

Unentscheidbarkeit

Theorem [Unentscheidbarkeit von $\mathcal{K}_0$].
Das allgemeine Halteproblem

$$\mathcal{H}_{\text{allg}} = \{\langle n, i \rangle \mid \ \mathcal{M}_n \text{ hält bei Eingabe } i\}$$

ist unentscheidbar.

Unentscheidbarkeit

Theorem [Unentscheidbarkeit von $\mathcal{K}_0$].
Das allgemeine Halteproblem

$$\mathcal{H}_{\mathrm{allg}} = \{\langle n, i\rangle \mid \mathcal{M}_n \text{ hält bei Eingabe } i\}$$

ist unentscheidbar.

Beweis 1:

Wir zeigen, dass $\mathcal{H}_0 \preceq \mathcal{H}_{\mathrm{allg}}$:

Sei $f : \mathbb{N} \to \mathbb{N}$ mit $f(n) = \langle n, 0\rangle$ (TM berechenbar).

Dann $n \in \mathcal{H}_0$ gdw. $\mathcal{M}_n$ hält bei Eingabe 0 gdw. $f(n) \in \mathcal{H}_{\mathrm{allg}}$.

Unentscheidbarkeit

Theorem [Unentscheidbarkeit von $\mathcal{K}_0$].
Das allgemeine Halteproblem

$$\mathcal{H}_{\text{allg}} = \{\langle n, i\rangle \mid \mathcal{M}_n \text{ hält bei Eingabe } i\}$$

ist unentscheidbar.

Beweis 2:

Wir zeigen, dass $\mathcal{H} \preceq \mathcal{H}_{\text{allg}}$:

Sei $g : \mathbb{N} \to \mathbb{N}$ mit $g(n) = \langle n, n\rangle$ (TM berechenbar).

Dann $n \in \mathcal{H}$ gdw. $\mathcal{M}_n$ hält bei Eingabe n gdw. $g(n) \in \mathcal{H}_{\text{allg}}$.

Unentscheidbarkeit

Ähnlich kann man per Reduktion die Unentscheidbarkeit der folgenden Probleme zeigen:

- $\mathcal{E}$, das Leerheitsproblem.

  $$\mathcal{E} := \{n \mid \mathcal{M}_n \text{ hält bei keiner Eingabe aus } \Sigma^*\}$$

- $\mathcal{T}$, das Totalitätsproblem.

  $$\mathcal{T} := \{n \mid \mathcal{M}_n \text{ hält bei jeder Eingabe aus } \Sigma^*\}.$$

- $\mathcal{E}q$, das Gleichheitsproblem.

  $$\mathcal{E}q := \{\langle n, m\rangle \mid \mathcal{M}_n \text{ akzeptiert die gleiche Sprache über } \Sigma \text{ wie } \mathcal{M}_m \}.$$

- $\mathcal{E}nt$, das Entscheidbarkeitsproblem.

  $$\mathcal{E}nt := \{n \mid \mathcal{M}_n \text{ akzeptiert eine entscheidbare Sprache über } \Sigma\}.$$

Übersicht

- Determinierte Turing-Maschinen (DTMs)
- Varianten von Turing-Maschinen
- Indeterminierte Turing-Maschinen (NTMs)
- Universelle determinierte Turing-Maschinen
- Entscheidbar/Aufzählbar
- Determinierte Turing-Maschinen entsprechen Typ 0
- Unentscheidbarkeit

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP