Rekursive Aufzählbarkeit
Die Reduktion

Prof. Dr. Berthold Vöcking
Lehrstuhl Informatik 1
Algorithmen und Komplexität
RWTH Aachen

November 2011

Semi-Entscheidbarkeit

Eine Sprache L wird von einer TM M *entschieden*, wenn

- M auf jeder Eingabe hält, und
- M genau die Wörter aus L akzeptiert.

Eine Sprache L, für die eine TM existiert, die L entscheidet, wird als *rekursiv* oder auch als *entscheidbar* bezeichnet.

Eine Sprache L wird von einer TM M *erkannt*, wenn

- M jedes Wort aus L akzeptiert, und
- M kein Wort akzeptiert, das nicht in L enthalten ist.

Def: Eine Sprache L, für die eine TM existiert, die L erkennt, wird als *semi-entscheidbar* bezeichnet.

Beispiel

Das Halteproblem

$$H \;=\; \{\langle M\rangle w \mid M \text{ hält auf } w\}\;\;.$$

ist nicht entscheidbar.

Behauptung: Das Halteproblem ist jedoch semi-entscheidbar.

Folgendes TM-Programm erkennt H:

Erhält M' eine Eingabe der Form $\langle M\rangle w$ so

- simuliert M' die TM M mit Eingabe w, und
- akzeptiert, falls M auf w hält.

Syntaktisch inkorrekte Eingaben werden von M' verworfen.

Aufzähler, Rekursive Aufzählbarkeit – Definition

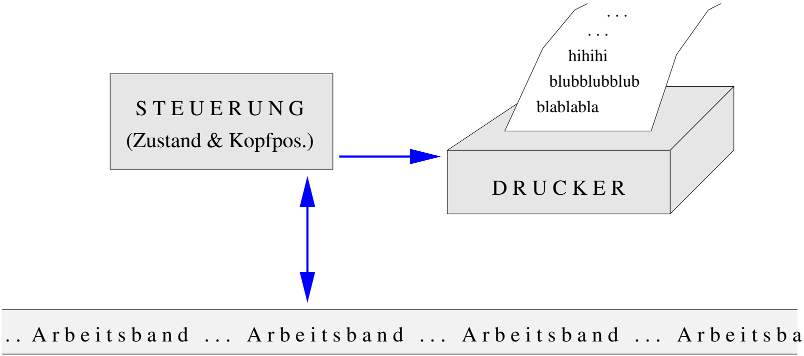
Ein *Aufzähler für eine Sprache* $L \subseteq \Sigma^*$ ist eine Variante einer TM mit einem angeschlossenen *Drucker* im Sinne eines zusätzlichen Ausgabebandes, auf dem sich der Kopf nur nach rechts bewegt.

Gestartet mit leerem Arbeitsband, gibt der Aufzähler alle Wörter aus L (möglicherweise mit Wiederholungen) auf dem Drucker aus. Die ausgegebenen Wörter sind dabei durch ein Zeichen getrennt, das nicht in Σ enthalten ist.

Definition

Eine Sprache, für die es einen Aufzähler gibt, heißt *rekursiv aufzählbar*.

Aufzähler – Illustration



rekursiv aufzählbar = semi-entscheidbar

Satz

Eine Sprache L ist genau dann semi-entscheidbar, wenn sie rekursiv aufzählbar ist.

Beweis: L ist rekursiv aufzählbar $\Rightarrow$ L ist semi-entscheidbar

Sei A ein Aufzähler für L. Wir konstruieren eine TM M, die L erkennt.

Bei Eingabe w arbeitet M wie folgt:

- M simuliert A mit Hilfe einer Spur, die die Rolle des Druckers übernimmt.
- Wenn immer ein neues Wort gedruckt worden ist, vergleicht M dieses Wort mit w und akzeptiert bei Übereinstimmung.

Korrektheit:

- Falls $w \in L$, so wird w irgendwann gedruckt und somit von M akzeptiert.
- Falls $w \notin L$, so wird w nicht gedruckt und somit nicht akzeptiert.

Beweis: L ist semi-entscheidbar $\Rightarrow$ L ist rekursiv aufzählbar

Sei nun M eine TM M, die L erkennt. Wir konstruieren einen Aufzähler A für L.

Programm von A:

For $i = 1, 2, 3, \ldots$

- Simuliere i Schritte von M auf jedem Wort aus $w_1, \ldots, w_i$.
- Wird dabei eines der Worte akzeptiert, so drucke es aus.

Korrektheit:

A druckt offensichtlich nur Wörter aus L aus. Aber druckt er auch alle Wörter aus L aus?

- Sei w_k ein Wort aus L.
- Dann wird w_k von M nach einer endlichen Anzahl von Schritten, sagen wir nach t_k vielen Schritten, akzeptiert.
- D.h. w_k wird von A ausgedruckt, und zwar in jeder Iteration $i \geq \max\{k, t_k\}$.

$\square$

Schnitte von Sprachen

Satz

a) Wenn die Sprachen L_1 und L_2 rekursiv sind, so ist auch die Sprache $L_1 \cap L_2$ rekursiv.

b) Wenn die Sprachen L_1 und L_2 rekursiv aufzählbar sind, so ist auch die Sprache $L_1 \cap L_2$ rekursiv aufzählbar.

Schnitte von Sprachen – Beweis a)

Seien M_1 und M_2 zwei TM, die L_1 bzw. L_2 entscheiden.

TM M, die $L_1 \cap L_2$ entscheidet:

- Auf Eingabe w, simuliert M zunächst das Verhalten von M_1 auf w und dann das Verhalten von M_2 auf w.
- Falls M_1 und M_2 das Wort w akzeptieren, so akzeptiert auch M; sonst wird w von M verworfen.

Korrektheit:

- Falls $w \in L_1 \cap L_2$, so wird M akzeptiert.
- Sonst wird w verworfen.

$\square$

Schnitte von Sprachen – Beweis b)

Seien nun M_1 und M_2 zwei TM, die L_1 bzw. L_2 erkennen.

TM M, die $L_1 \cap L_2$ erkennt:

- Auf Eingabe w, simuliert M zunächst das Verhalten von M_1 auf w und dann das Verhalten von M_2 auf w.
- Falls M_1 und M_2 akzeptieren, so akzeptiert auch M.

Wir verwenden dieselbe TM M wie in a)

Korrektheit:

- Falls $w \in L_1 \cap L_2$, so wird w von M akzeptiert.
- Sonst wird w nicht akzeptiert.

$\square$

Vereinigungen von Sprachen

Satz

a) Wenn die Sprachen L_1 und L_2 rekursiv sind, so ist auch die Sprache $L_1 \cup L_2$ rekursiv.
b) Wenn die Sprachen L_1 und L_2 rekursiv aufzählbar sind, so ist auch die Sprache $L_1 \cup L_2$ rekursiv aufzählbar.

Vereinigungen von Sprachen – Beweis a)

Seien M_1 und M_2 zwei TM, die L_1 bzw. L_2 entscheiden.

TM M, die $L_1 \cup L_2$ entscheidet

- Auf Eingabe w, simuliert M zunächst das Verhalten von M_1 auf w und dann das Verhalten von M_2 auf w.
- Falls M_1 oder M_2 akzeptiert, so akzeptiert auch M. Sonst verwirft M die Eingabe.

Korrektheit:

- Falls $w \in L_1 \cup L_2$, so wird w von M_1 oder M_2 und somit auch von M akzeptiert.
- Sonst verwerfen M_1 und M_2, so dass auch M verwirft.

Vereinigungen von Sprachen – Beweis b)

Seien nun M_1 und M_2 zwei TM, die L_1 bzw. L_2 erkennen.

Welches Problem tritt auf, wenn wir die Simulation aus a) einfach übernehmen?

Idee: Simuliere M_1 und M_2 *parallel* statt sequentiell ...

Vereinigungen von Sprachen – Beweis b)

TM M, die $L_1 \cup L_2$ erkennt

- Wir nehmen o.B.d.A. an, dass M über zwei Bänder verfügt.
- Auf Band 1 wird M_1 auf w simuliert.
- Auf Band 2 wird M_2 auf w simuliert.
- Sobald ein Schritt erreicht wird, in dem M_1 oder M_2 akzeptieren, so akzeptiert auch M.

Korrektheit:

- Falls $w \in L_1 \cup L_2$, so wird w von M_1 oder M_2 und somit auch von M akzeptiert.
- Sonst wird w nicht akzeptiert.

Lemma $*$

Seien $L \subseteq \Sigma^*$ und $\bar{L} = \Sigma^* \setminus L$ rekursiv aufzählbar. Dann ist L rekursiv.

Beweis: Seien M und $\bar{M}$ Maschinen, die L bzw. $\bar{L}$ erkennen.

Die TM M' entscheidet L durch eine parallele Simulation von M und $\bar{M}$ auf der Eingabe w:

- M' akzeptiert w, sobald M akzeptiert.
- M' verwirft w, sobald $\bar{M}$ akzeptiert.

Da entweder $w \in L$ oder $w \notin L$, tritt eines dieser Ereignisse nach endlicher Zeit ein, so dass die Terminierung von M' sichergestellt ist. $\square$

Komplemente von Sprachen

Beobachtung 1:

Wenn die Sprache L rekursiv ist, so ist auch $\bar{L}$ rekursiv, da wir das Akzeptanzverhalten einer TM M, die M entscheidet invertieren können.

Beobachtung 2:

Die Menge der rekursiv aufzählbaren Sprachen ist hingegen nicht gegen Komplementbildung abgeschlossen.

Beispiel:

- H ist rekursiv aufzählbar.
- Wäre $\bar{H}$ ebenfalls rekursiv aufzählbar, so wäre H nach Lemma $*$ rekursiv.
- Also ist $\bar{H}$ nicht rekursiv aufzählbar.

Schlussfolgerung

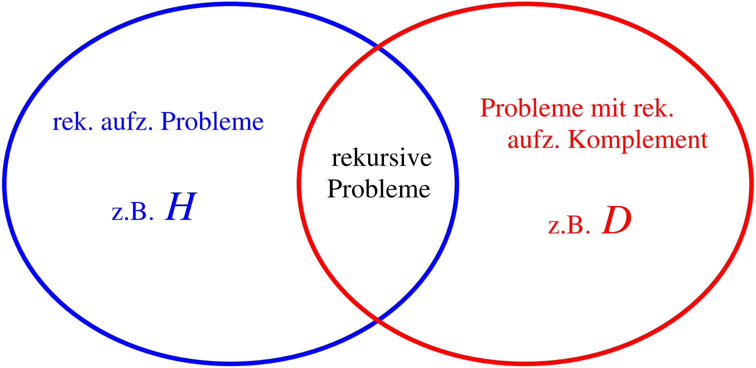
Korollar

Für jede Sprache L gilt eine der vier folgenden Eigenschaften.

1. L ist rekursiv und sowohl L als auch $\bar{L}$ sind rekursiv aufzählbar.
2. L ist rekursiv aufzählbar, aber $\bar{L}$ ist nicht rekursiv aufzählbar
3. $\bar{L}$ ist rekursiv aufzählbar, aber L ist nicht rekursiv aufzählbar
4. sowohl L als auch $\bar{L}$ sind nicht rekursiv aufzählbar

Beispiele:

- Kategorie 1: Graphzusammenhang, Hamiltonkreis
- Kategorie 2: H, H_ϵ, $\bar{D}$
- Kategorie 3: D, $\bar{H}$, $\bar{H}_\epsilon$
- Kategorie 4: $H_{all} = \{\langle M \rangle \mid M \text{ hält auf jeder Eingabe}\}$



nicht rek. aufz. Probleme, deren Komplement ebenfalls nicht rek. aufz. ist

z.B. H_{all}

Allgemeines Halteproblem

Das *allgemeine Halteproblem* ist definiert als

$$H_{\text{all}} = \{\langle M\rangle \mid M \text{ hält auf jeder Eingabe}\}$$

Wie kann man nachweisen, dass sowohl H_{all} als auch $\bar{H}_{\text{all}}$ nicht rekursiv aufzählbar sind?

Wir verwenden eine spezielle Variante der Unterprogrammtechnik, die *Reduktion*.

Die Reduktion

Definition

Es seien L_1 und L_2 Sprachen über einem Alphabet Σ. Dann heißt L_1 auf L_2 *reduzierbar*, Notation $L_1 \leq L_2$, wenn es eine berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ gibt, so dass für alle $x \in \Sigma^*$ gilt

$$x \in L_1 \;\Leftrightarrow\; f(x) \in L_2 \;\; .$$

Die Reduktion

Lemma

Falls $L_1 \leq L_2$ und L_2 rekursiv aufzählbar ist, so ist L_1 rekursiv aufzählbar.

Beweis: Wir konstruieren eine TM M_1, die L_1 erkennt, durch Unterprogrammaufruf einer TM M_2, die L_2 erkennt:

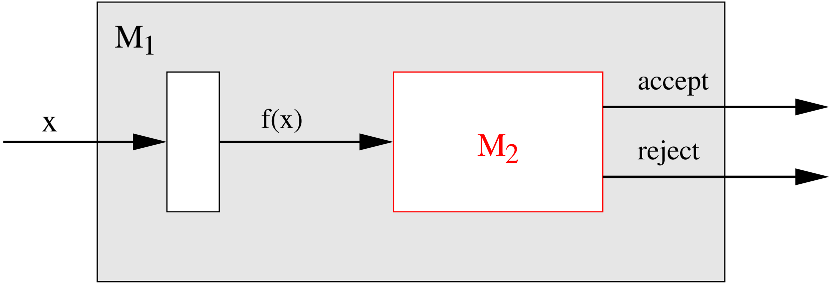
- Die TM M_1 berechnet $f(x)$ aus ihrer Eingabe x.
- Dann simuliert M_1 die TM M_2 mit der Eingabe $f(x)$ und übernimmt das Akzeptanzverhalten.

Korrektheit:

$$M_1 \text{ akz } x \;\Leftrightarrow\; M_2 \text{ akz } f(x) \;\Leftrightarrow\; f(x) \in L_2 \;\Leftrightarrow\; x \in L_1 \;.$$

□

Die Reduktion



Wdh: Die Reduktion

Im Umkehrschluss gilt:

Lemma

Falls $L_1 \leq L_2$ und L_1 nicht rekursiv aufzählbar ist, so ist L_2 nicht rekursiv aufzählbar.

Anwendung der Reduktion

H_ϵ ist nicht rekursiv, aber rekursiv aufzählbar. Folglich ist $\bar{H}_\epsilon$ nicht rekursiv aufzählbar.

Wir zeigen nun

Behauptung A

$$\bar{H}_\epsilon \leq \bar{H}_{\text{all}}$$

Behauptung B

$$\bar{H}_\epsilon \leq H_{\text{all}}$$

Aus diesen Reduktionen folgt:

Satz

Sowohl $\bar{H}_{\text{all}}$ als auch H_{all} sind nicht rekursiv aufzählbar.