Berechenbarkeit und Komplexität: Rekursive Aufzählbarkeit und die Technik der Reduktion

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

26. November 2007

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 einer nicht-entscheidbaren aber semi-entscheidbaren Sprache

Ein uns bekanntes nicht-rekursiven Problem ist das Halteproblem

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

Behauptung: Das Halteproblem ist semi-entscheidbar.

Arbeitsweise einer TM M', die H erkennt

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.

Eigenschaften des Aufzählers

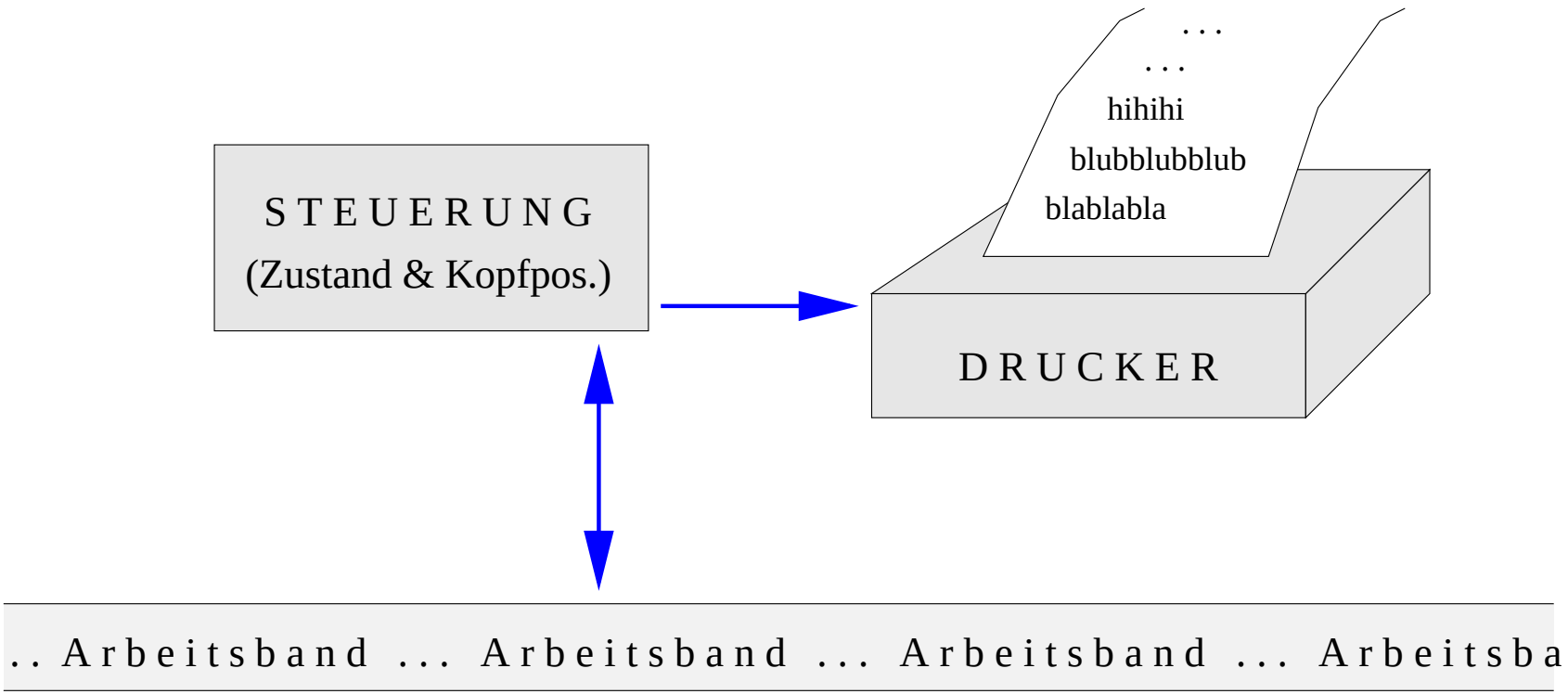
Gestartet mit leerem Arbeitsband, *enummeriert* der Aufzähler alle Wörter aus L (möglicherweise mit Wiederholungen) auf dem Drucker, d.h.

- gedruckt werden ausschließlich Wörter aus L, und
- jedes Wort aus L wird irgendwann ausgedruckt.

Die enummerierten Wörter sind dabei durch ein Zeichen getrennt, das nicht in Σ enthalten ist.

Def: 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:

Als erstes zeigen wir, wie man aus einem Aufzähler für L eine TM M konstruieren kann, die L erkennt.

- M simuliert den Aufzähler, d.h. schreibt alle Wörter die auf dem Drucker ausgegeben werden nacheinander aufs Band.
- Sobald ein neues Wort enummeriert worden ist, vergleicht M dieses Wort mit dem Eingabewort w und akzeptiert bei Übereinstimmung.

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

rekursiv aufzählbar = semi-entscheidbar – Forts. Beweis

Zum Beweis der anderen Richtung zeigen wir nun, wie man aus einer TM M, die L erkennt, einen Aufzähler konstruiert.

Seien $w_1, w_2, w_3, \ldots$ die Wörter aus Σ^* in kanonischer Reihenfolge. Der Aufzähler arbeitet wie folgt.

Für $i = 1, 2, 3, \ldots$

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

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

- Sei w_k ein Wort aus L.
- Sei t_k die Anzahl Schritte, die M benötigt, um w_k zu akzeptieren.
- In jeder Iteration $i \geq \max\{k, t_k\}$ druckt der Aufzähler dieses Wort somit aus.

□

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

Beweis:

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

Arbeitsweise einer 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 akzeptieren, so akzeptiert auch M.

M akzeptiert offensichtlich die Eingaben aus $L_1 \cap L_2$ und hält auf jeder Eingabe, da sowohl M_1 als auch M_2 auf jeder Eingabe halten.

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

Konstruktion wie in a) bis auf die Tatsache, dass die Terminierung im Falle $w \notin L_1 \cap L_2$ nicht sichergestellt ist. □

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

Beweis:

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

Arbeitsweise einer 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 akzeptieren, so akzeptiert auch M.

M akzeptiert offensichtlich die Eingaben aus $L_1 \cup L_2$ und hält auf jeder Eingabe, da sowohl M_1 als auch M_2 auf jeder Eingabe halten.

Vereinigungen von Sprachen – Forts. Beweis

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

Statt der „sequentiellen Simulation“ von M_1 und M_2 verwenden wir eine „parallele Simulation“der beiden TM:

Arbeitsweise einer TM M, die $L_1 \cup L_2$ erkennt

- M verwendet jeweils ein Band für die Simulation des Bandes von M_1 und des Bandes von M_2.
- Die Zustandsmenge von M enthält das Kreuzprodukt $Q_1 \times Q_2$ der Zustandsmengen von M_1 und M_2
- M akzeptiert, sobald eine der beiden TM akzeptiert.

□

„2 × rekursiv aufzählbar = rekursiv“

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. □

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 rekursiv.
- Also ist $\bar{H}$ nicht rekursiv aufzählbar.

Schlussfolgerung

Korollar

Für jede Sprache L gilt eine der beiden folgenden Eigenschaften.

- L ist rekursiv und sowohl L als auch $\bar{L}$ sind rekursiv aufzählbar.
- L ist nicht rekursiv und L oder $\bar{L}$ sind nicht rekursiv aufzählbar.

Der letzte dieser beiden Fälle erlaubt auch, dass sowohl L als auch $\bar{L}$ nicht rekursiv aufzählbar sind.

Wir behaupten, beispielsweise die Sprache

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

hat diese Eigenschaft.

Die Reduktion

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

Die Reduktion ist eine Spezialisierung der Unterprogrammtechnik, die gut zum Nachweis rekursiver Aufzählbarkeit geeignet ist.

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:

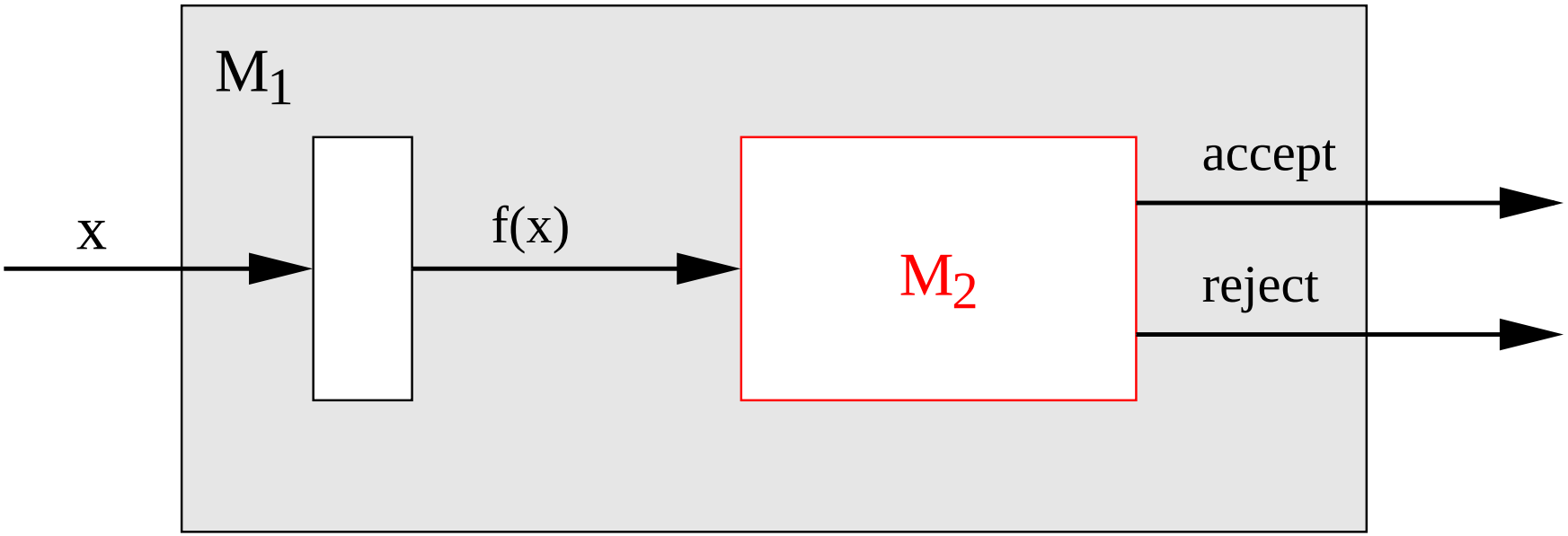
- Die TM M_1 berechnet $f(x)$ aus ihrer Eingabe x.
- Dann simuliert M_1 die TM M_2 mit der Eingabe $f(x)$ und übernimmt das Akzeptanzverhalten.

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



Die Reduktion

Es gilt übrigens auch

Lemma

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

Dieses Lemma folgt direkt daraus, dass die Reduktion eine spezialisierte Variante der Unterprogrammtechnik ist.

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

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

Wäre also $\bar{H}_{\text{all}}$ oder H_{all} rekursiv aufzählbar, so wäre auch $\bar{H}_\epsilon$ rekursiv aufzählbar. Also folgt

Satz

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

Beweis von Behauptung A: $\bar{H}_\epsilon \leq \bar{H}_{\text{all}}$

Wir konstruieren eine berechenbare Funktion f, die Ja-Instanzen von $\bar{H}_\epsilon$ auf Ja-Instanzen von $\bar{H}_{\text{all}}$ abbildet, und Nein-Instanzen von $\bar{H}_\epsilon$ auf Nein-Instanzen von $\bar{H}_{\text{all}}$ abbildet.

Die Funktion f

Sei w die Eingabe für $\bar{H}_\epsilon$.

- Wenn w keine gültige Gödelnummer ist, so sei $f(w) = w$.
- Falls $w = \langle M \rangle$ für eine TM M, so sei $f(w)$ die Gödelnummer einer TM M^*_ϵ mit der folgenden Eigenschaft: M^*_ϵ ignoriert die Eingabe und simuliert M mit der Eingabe ϵ.

Die Funktion f ist offensichtlich berechenbar.

Falls w keine Gödelnummer ist, so ist die Korrektheit klar, denn in diesem Fall gilt $w \in \bar{H}_\epsilon$ und $f(w) \in \bar{H}_{\text{all}}$.

Beweis von Behauptung A: $\bar{H}_\epsilon \leq \bar{H}_{\text{all}}$ — Fortsetzung

Sei nun $w = \langle M \rangle$ für eine TM M, so dass $f(w) = \langle M^*_\epsilon \rangle$. Es gilt

$$\begin{aligned} w \notin \bar{H}_\epsilon &\Rightarrow M \text{ hält auf der Eingabe } \epsilon \\ &\Rightarrow M^*_\epsilon \text{ hält auf jeder Eingabe} \\ &\Rightarrow \langle M^*_\epsilon \rangle \in H_{\text{all}} \\ &\Rightarrow f(w) \notin \bar{H}_{\text{all}} \ . \end{aligned}$$

$$\begin{aligned} w \in \bar{H}_\epsilon &\Rightarrow M \text{ hält nicht auf Eingabe } \epsilon \\ &\Rightarrow M^*_\epsilon \text{ hält auf keiner Eingabe} \\ &\Rightarrow \langle M^*_\epsilon \rangle \notin H_{\text{all}} \\ &\Rightarrow f(w) \in \bar{H}_{\text{all}} \ . \end{aligned}$$

Also gilt $w \in \bar{H}_\epsilon \Leftrightarrow f(w) \in \bar{H}_{\text{all}}$ und somit ist die Funktion f korrekt konstruiert. □

Beweis von Behauptung B: $\bar{H}_\epsilon \leq H_{\text{all}}$

Wir konstruieren eine berechenbare Funktion f, die Ja-Instanzen von $\bar{H}_\epsilon$ auf Ja-Instanzen von H_{all} abbildet, und Nein-Instanzen von $\bar{H}_\epsilon$ auf Nein-Instanzen von H_{all} abbildet.

Die Funktion f

Sei w die Eingabe für $\bar{H}_\epsilon$. Sei w' irgendein Wort aus H_{all}.

- Wenn w keine gültige Gödelnummer ist, so sei $f(w) = w'$.
- Falls $w = \langle M \rangle$ für eine TM M, so sei $f(w)$ die Gödelnummer einer TM M'_M, die sich auf Eingaben der Länge i wie folgt verhält: M'_M simuliert die ersten i Schritte von M auf der Eingabe ϵ. Wenn M innerhalb dieser i Schritte hält, dann geht M'_M in eine Endlosschleife, ansonsten hält M'_M.

Die Funktion f ist offensichtlich berechenbar.

Falls w keine Gödelnummer ist die Korrektheit klar, denn in diesem Fall gilt $w \in \bar{H}_\epsilon$ und $f(w) = w' \in H_{\text{all}}$.

Beweis von Behauptung B: $\bar{H}_\epsilon \leq H_{\text{all}}$ — Fortsetzung

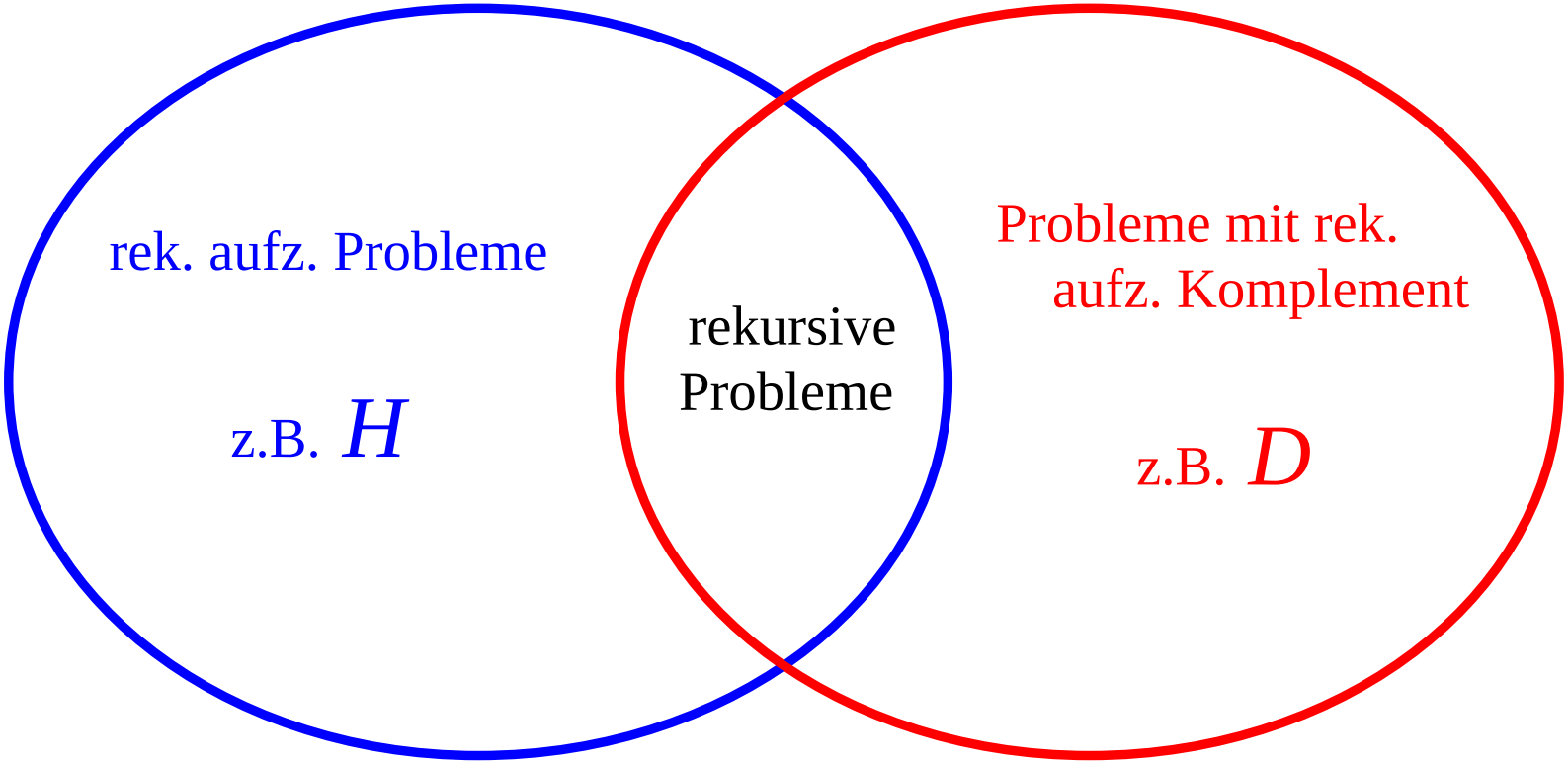
Sei nun $w = \langle M \rangle$ für eine TM M, so dass $f(w) = \langle M'_M \rangle$.

$$
\begin{aligned}
w \notin \bar{H}_\epsilon \quad &\Rightarrow \quad M \text{ hält auf der Eingabe } \epsilon \\
&\Rightarrow \quad \exists i\text{: } M \text{ hält innerhalb von } i \text{ Schritten auf } \epsilon \\
&\Rightarrow \quad \exists i\text{: } M'_M \text{ hält nicht auf Eingaben der Länge } i \\
&\Rightarrow \quad f(w) = \langle M'_M \rangle \notin H_{\text{all}} \ .
\end{aligned}
$$

$$
\begin{aligned}
w \in \bar{H}_\epsilon \quad &\Rightarrow \quad M \text{ hält nicht auf der Eingabe } \epsilon \\
&\Rightarrow \quad \neg\exists i\text{: } M \text{ hält innerhalb von } i \text{ Schritten auf } \epsilon \\
&\Rightarrow \quad \forall i\text{: } M'_M \text{ hält auf Eingaben der Länge } i \\
&\Rightarrow \quad f(w) = \langle M'_M \rangle \in H_{\text{all}} \ .
\end{aligned}
$$

Also gilt $w \in \bar{H}_\epsilon \ \Leftrightarrow \ f(w) \in H_{\text{all}}$ und somit ist die Funktion f korrekt konstruiert. □

Berechenbarkeitslandschaft



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

z.B. H_{all}