CoNe Freiburg

Informatik III

3.2 Nichtberechenbarkeit

Christian Schindelhauer

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Wintersemester 2007/08

Berechenbarkeitstheorie

Aufzählen und Abzählen

2

Warum rekursiv aufzählbar rekursiv aufzählbar heißt

- **Definition**
  - Eine aufzählende Turing-Maschine ist eine Turingmaschine, die mit einem zusätzlichen speziellen Ausgabe-Band ausgestattet ist.
    - Die Turing-Maschine muss nicht unbedingt halten
    - Auf dem Ausgabeband kann die Turingmaschine nur nach rechts gehen.
    - Wörter sind durch das Sondersymbol “_” von einander getrennt und können damit weder gelöscht noch überschrieben werden
    - Die Vereinigung aller jemals erzeugten Wörter, beschreibt die Sprache der aufzählenden Turing-Maschine
- **Theorem**
  - Eine Sprache ist rekursiv aufzählbar genau dann wenn eine aufzählende Turing-Maschine sie beschreibt.

Was heißt abzählbar im Gegensatz zu rekursiv aufzählbar?

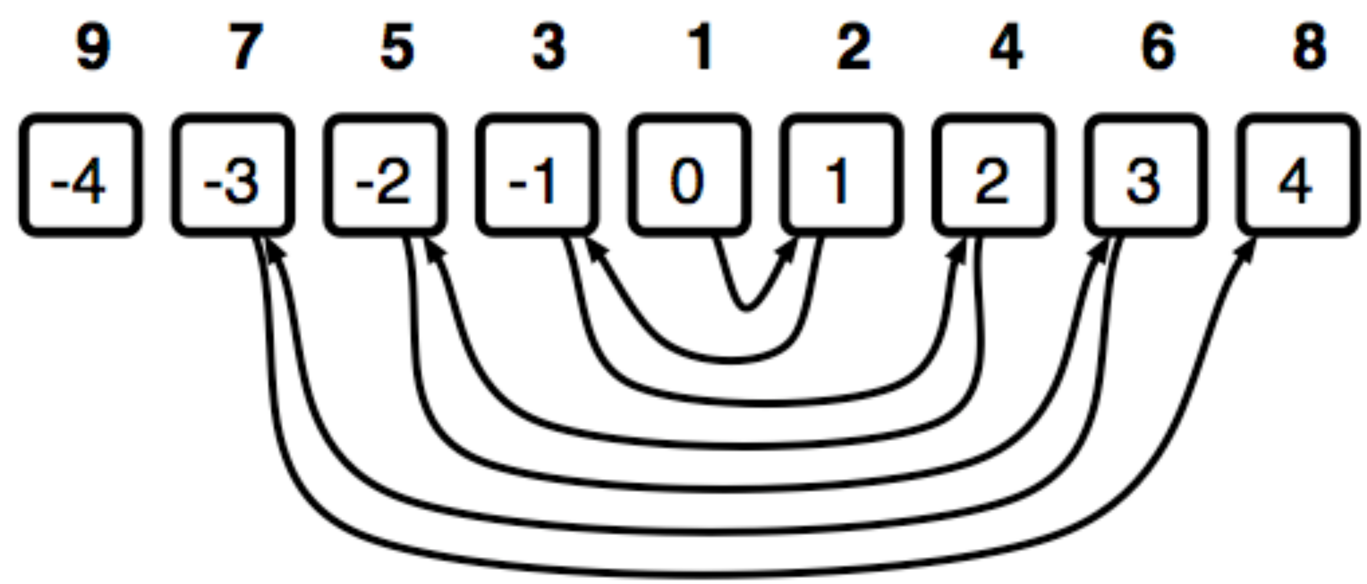
- **Definition**
  - Eine Menge M heißt abzählbar, wenn es eine (nicht unbedingt berechenbare) Funktion $f: \mathbb{N} \rightarrow M$ gibt,
    - so dass es für jedes $m \in M$ eine natürliche Zahl $i \in \mathbb{N}$ gibt mit $f(i) = m$.
- **Lemma**
  - Jede rekursiv aufzählbare Menge ist abzählbar
  - Jede Teilmenge einer abzählbaren Menge ist abzählbar

Hilberts Hotel

- **Hilberts Hotel hat unendlich viele Zimmer**
- **Alle Zimmer sind ausgebucht**
  - Es sind also schon unendlich viele Gäste da!
- **Kann der Hotelier dennoch weitere Gäste aufnehmen?**
  - Ein neuer Gast
  - Ein Bus mit unendlich vielen neuen Gästen
  - Unendlich viele Busse mit unendlich vielen neuen Gästen

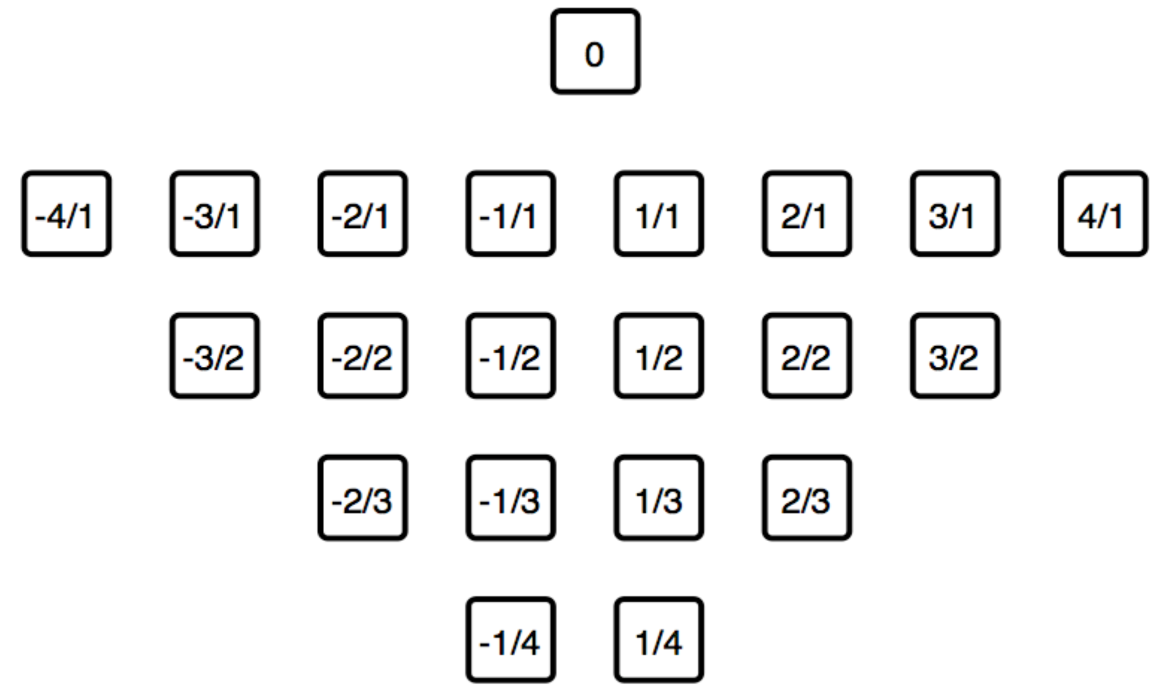
Die ganzen Zahlen sind abzählbar

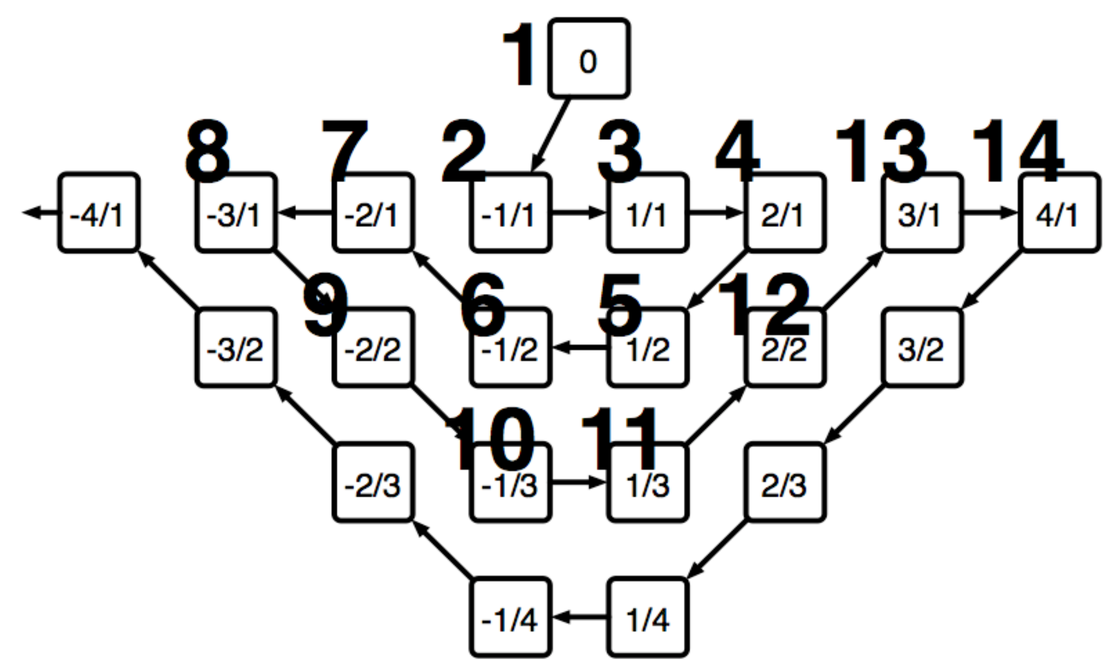
- **Theorem**
  - Die Menge der ganzen Zahlen ist abzählbar
- **Beweis**
  - Konstruiere eine Abzählung aller ganzen Zahlen:
  - 0 erhält die Nummer 1
  - alle positiven Zahlen x erhalten die Nummer 2x
  - alle negativen Zahlen x erhalten die Nummer 2(-x)+1



Die rationalen Zahlen sind abzählbar

- **Theorem**
  - Die Menge der rationalen Zahlen ist abzählbar
- **Beweis**
  - Die rationalen Zahlen sind definiert als Tupel aus einer ganzen Zahl und einer natürlichen Zahl
  - Zähle alle diese Paare geeignet auf
    - Mehrfachaufzählungen sind irrevelant (=egal)





Die reellen Zahlen sind nicht abzählbar

- **Theorem**
  - Die Menge der reellen Zahlen sind nicht abzählbar
- **Beweis**
  - Betrachte alle reellen Zahlen aus [0,1[in der Dezimaldarstellung
  - Angenommen alle reellen Zahlen sind abzählbar mit x_1, x_2, x_3,...

	1	2	3	4	5	...
$x_1 = 0,$	9	9	9	9	9	...
$x_2 = 0,$	6	4	2	5	2	...
$x_3 = 0,$	3	2	7	3	2	...
$x_4 = 0,$	6	7	2	1	6	...
$x_5 = 0,$	2	1	7	2	8	...
...	...	...	...	...	...	...

Die reellen Zahlen sind nicht abzählbar

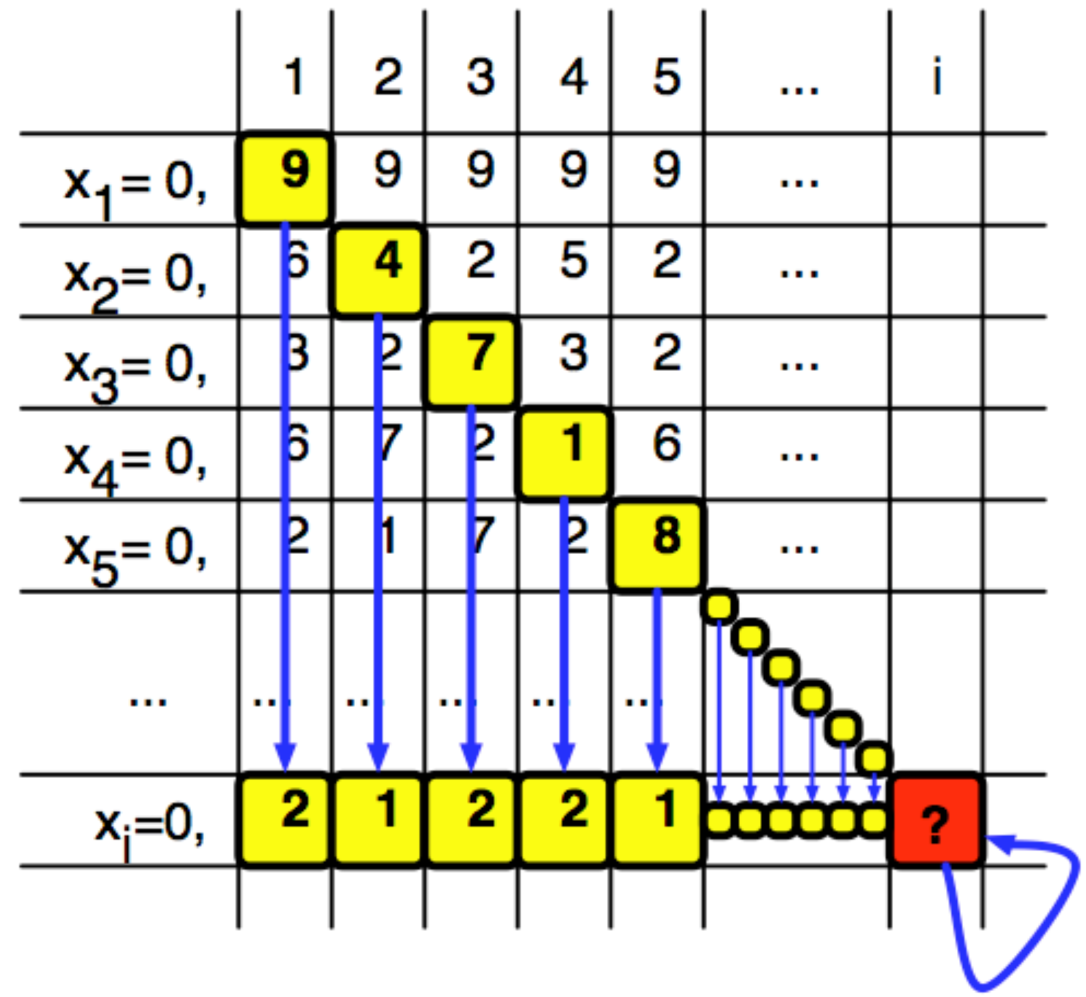
- **Theorem**
  - Die Menge der reellen Zahlen sind nicht abzählbar
- **Beweis**
  - Betrachte alle reellen Zahlen aus [0,1[in der Dezimaldarstellung
  - Angenommen alle reellen Zahlen sind abzählbar mit x_1, x_2, x_3, ...
  - Betrachte reelle Zahl z, wobei
    - j-te Ziffer ist 1, falls j-te Ziffer von x_j gerade ist
    - j-te Ziffer ist 2, falls j-te Ziffer von x_j ungerade ist
  - Diese Zahl hat auch einen Index i,
    - d.h. es gibt ein i mit $x_i = z$
    - wenn die Annahme stimmt

	1	2	3	4	5	...	i
$x_1 = 0,$	**9**	9	9	9	9	...	
$x_2 = 0,$	6	**4**	2	5	2	...	
$x_3 = 0,$	3	2	**7**	3	2	...	
$x_4 = 0,$	6	7	2	**1**	6	...	
$x_5 = 0,$	2	1	7	2	**8**	...	
...	...	...	...	...	...	...	
$x_i = 0,$	**2**	**1**	**2**	**2**	**1**	...	

j-te Ziffer von x_i ist 2, falls j-te Ziffer von x_j ungerade
ist 1, falls j-te Ziffer von x_j gerade

Die reellen Zahlen sind nicht abzählbar

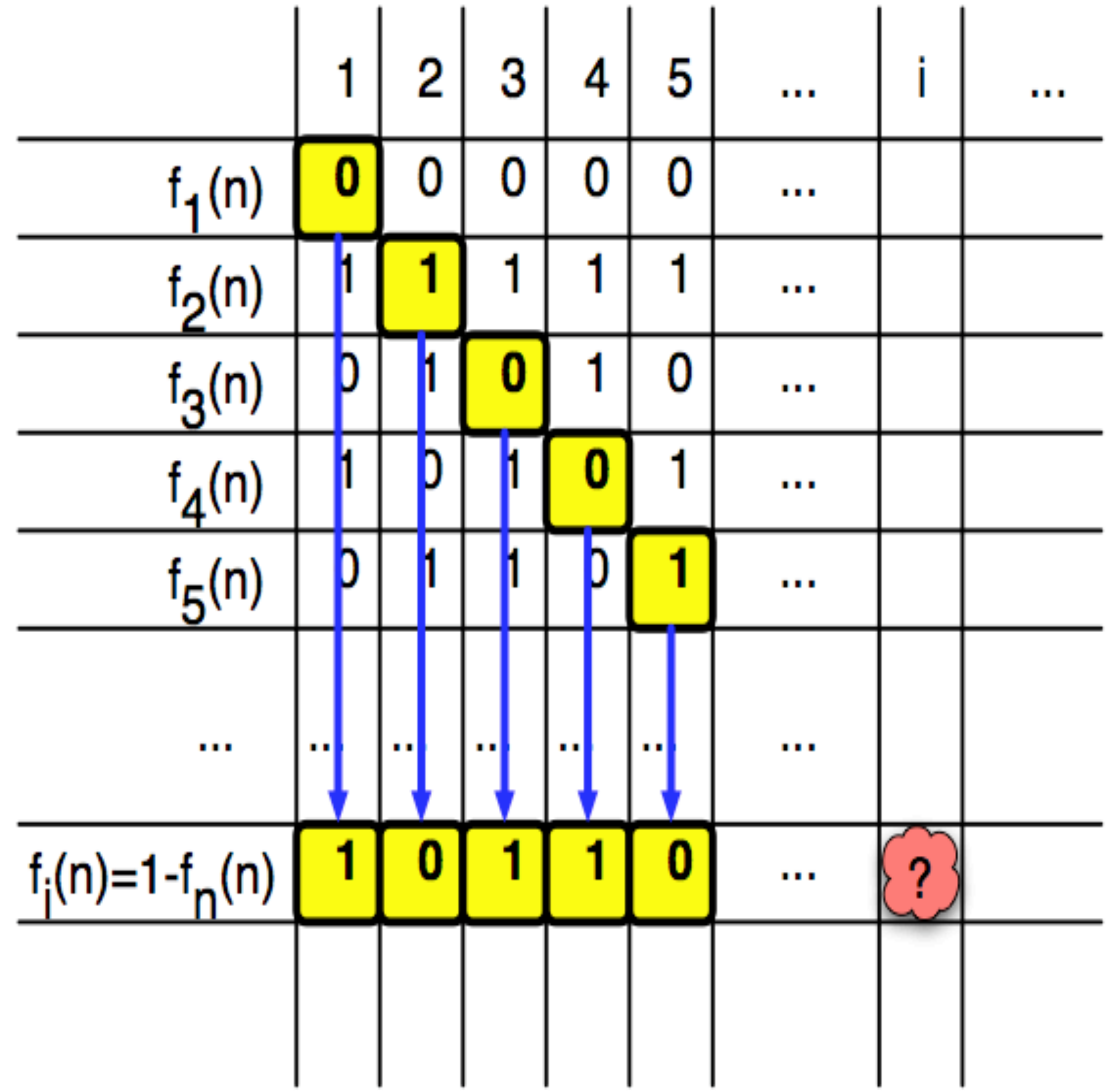
- **Beweis**
  - Betrachte alle reellen Zahlen aus [0,1[in der Dezimaldarstellung
  - Angenommen alle reellen Zahlen sind abzählbar mit x_1, x_2, x_3, ...
  - Betrachte reelle Zahl z, wobei
    - j-te Ziffer ist 1, falls j-te Ziffer von x_j gerade ist
    - j-te Ziffer ist 2, falls j-te Ziffer von x_j ungerade ist
  - Diese Zahl hat auch einen Index i,
    - d.h. es gibt ein i mit $x_i = z$
    - wenn die Annahme stimmt
  - Ist die i-te Ziffer von x_i jetzt gerade oder ungerade
    - weder noch: Widerspruch
  - Also ist die Annahme falsch



i-te Ziffer von x_i ist 2, falls i-te Ziffer von x_i ungerade
ist 1, falls i-te Ziffer von x_i gerade

Noch ein Beispiel

- **Theorem**
  - Die Menge aller Funktionen die von $\mathbb{N}$ auf {0,1} abbilden, ist nicht abzählbar.
- **Beweis:**
  - Angenommen es gibt eine Funktion die alle Funktionen f_1, f_2, ... abzählt.
  - Betrachte die Funktion 1 - $f_i(i)$.
  - Diese Funktion ist nicht in f_1, f_2, ..., da für jedes i gilt
    $f_i(i) \neq 1 - f_i(i)$.
  - Diese Funktion ist also in der Abzählung f_1, f_2, ... nicht enthalten (sollte aber).



Berechenbarkeitstheorie

Diagonalisierung und das Wortproblem

12

Die Simulator-TM

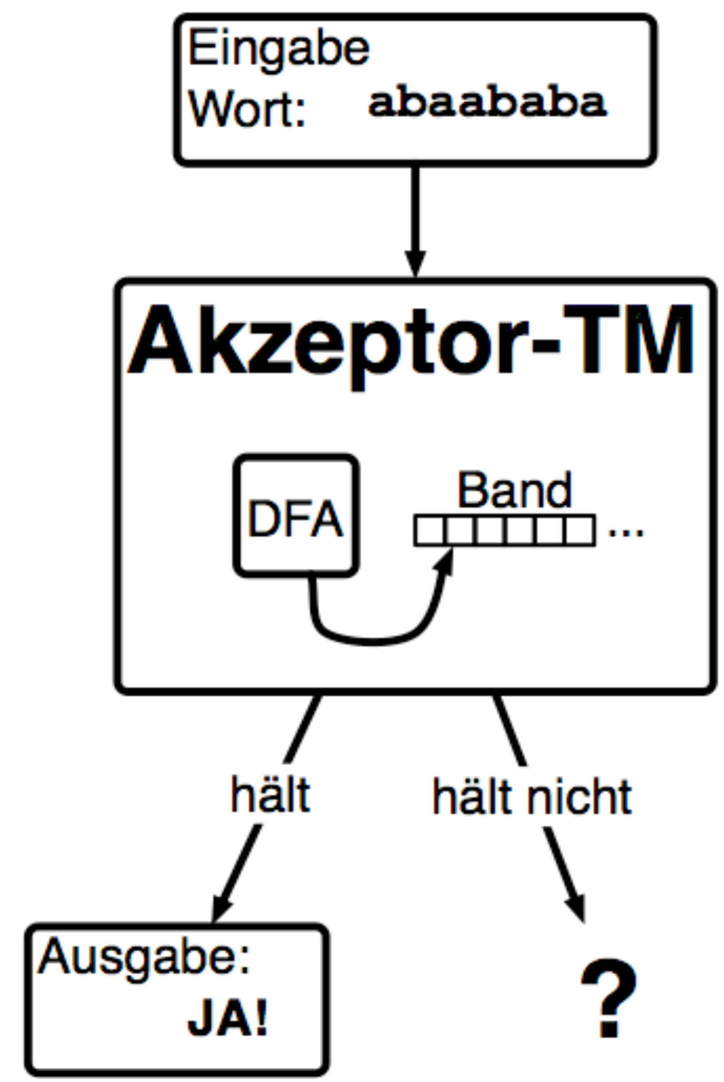
- **Lemma**
  - Es gibt eine TM S, die einen Schritt einer gegebenen TM M und einer Konfiguration berechnen kann,
    - d.h. die Nachfolgekonfiguration ausgeben
- **Beweis**
  - S verfügt über
    - ein Band für Kodierung der 1-Band-TM M
    - ein Band für den aktuellen Zustand
    - ein Band für den Bandinhalt von M mit markierter Kopfposition
    - ein Band für eigene Berechnungen (Zählen)
  - S sucht nach der aktuellen Kopfposition
  - S sucht in der Kodierung von M nach Übergang
  - S schreibt Buchstaben auf das “Band”-Band
  - S bewegt die Kopfposition und markiert entsprechendes Zeichen auf dem Band
  - S schreibt Folgezustand

Das TM-Wortproblem

- **Definition**
  - Das Wortproblem der Turing-Maschinen ist definiert als
  - gegeben:
    - eine Turingmaschine M
    - ein Wort w
  - gesucht:
    - akzeptiert M das Wort w?
- **Die alternative Darstellung als Sprache ist:**

$$\mathsf{A}_{\mathsf{TM}} = \{\langle \mathsf{M}, \mathsf{w}\rangle \mid M \text{ ist eine TM und } M \text{ akzeptiert } w\}$$

  - hierbei ist <M,w> eine geeignete Kodierung der TM M und des Wortes w



Das TM-Wortproblem ist nicht entscheidbar

- **Theorem**
  - Die Sprache A_{TM} ist nicht rekursiv, d.h. das TM-Wortproblem ist nicht entscheidbar
- **Beweis**
  - Annahme A_{TM} ist entscheidbar:
    - Dann gibt es eine TM M_H, die auf Eingabe M und w
      - immer hält und
      - folgendes berechnet:
  - $M_H(\langle M,w\rangle) = \chi_{A_{TM}}(\langle M,w\rangle)$

$$\mathsf{H}(\langle \mathsf{M}, \mathsf{w}\rangle) = \begin{cases} 1\,, & \text{falls } M \text{ das Wort } w \text{ akzeptiert} \\ 0\,, & \text{falls } M \text{ nicht hält oder } w \text{ verwirft} \end{cases}$$

- **Konstruiere neue TM D**
  - D = “Auf Eingabe <M>, wobei M TM ist
  - Führe Berechnung von M_H auf Eingabe <M, <M> > durch
  - Falls M_H akzeptiert, dann verwerfe
Falls M_H verwirft, dann akzeptiere
- **Beachte:**
  - <M, <M> > ist eine Kodierung aus
    - einer Turingmaschine M und
    - einer kodierten Turingmaschine
- **Fakt**
  - D hält immer und berechnet die folgende Funktion

$$\mathsf{D}(\langle \mathsf{M}\rangle) = \begin{cases} 0\,, & \text{falls } M \text{ das Wort } \langle M\rangle \text{ akzeptiert} \\ 1\,, & \text{falls } M \text{ nicht hält oder } \langle M\rangle \text{ verwirft} \end{cases}$$

Beweis: Was berechnet D auf <D>?

- **Unter der Annahme, dass das TM-Wortproblem berechenbar ist:**
- **Fakt: TM D hält immer**
- **Fakt: TM D berechnet die folgende Funktion:**

$$\mathrm{D}(\langle \mathrm{M} \rangle) = \begin{cases} 0\,, & \text{falls } M \text{ das Wort } \langle M \rangle \text{ akzeptiert} \\ 1\,, & \text{falls } M \text{ nicht hält oder } \langle M \rangle \text{ verwirft} \end{cases}$$

- **Was liefert nun D(<D>)?**
  - Entweder akzeptieren oder verwerfen
- **Akzeptieren?**
  - Falls D(<D>) akzeptiert, dann ist D(<D>) = 0
  - Also D(<D>) verwirft ⇒ Widerspruch
- **Verwerfen**
  - Falls D(<D>) verwirft, dann ist D(<D>) = 1
  - Also D(<D>) akzeptiert ⇒ Widerspruch

Grafische Darstellung des Beweises

- **Angenommen: Das TM-Wortproblem ist rekursiv!**

	$<M_1>$	$<M_2>$	$<M_3>$	$<M_4>$	...
M_1	1	0	1	1	...
M_2	1	1	1	1	...
M_3	1	0	0	1	...
M_4	0	0	0	0	...
...	...	...	...	...	

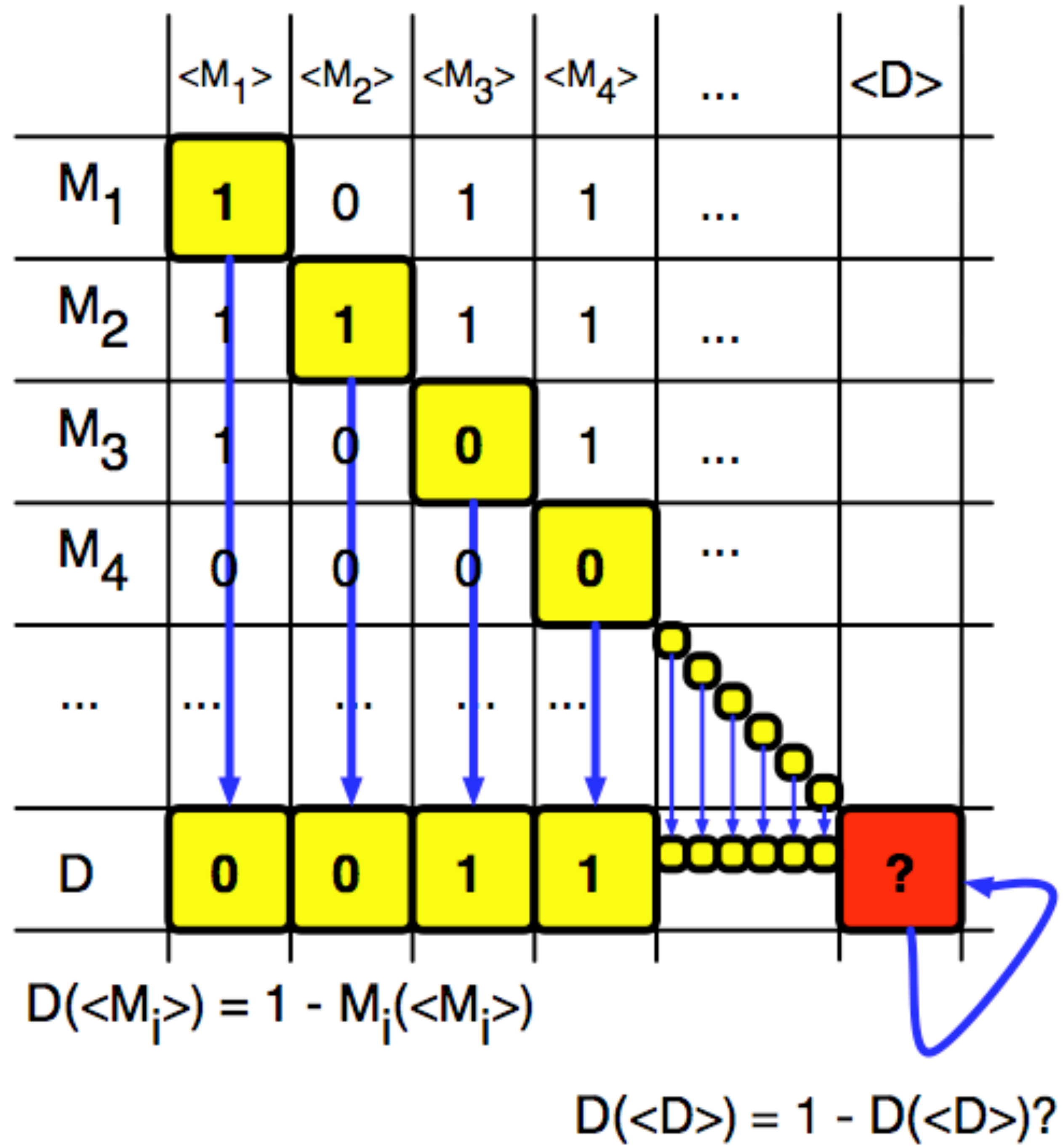
0: Turingmaschine hält und akzeptiert
1: Turingmaschine hält nicht oder verwirft

	$<M_1>$	$<M_2>$	$<M_3>$	$<M_4>$	...
M_1	**1**	0	1	1	...
M_2	1	**1**	1	1	...
M_3	1	0	**0**	1	...
M_4	0	0	0	**0**	...
...	...	...	...	...	
D	**0**	**0**	**1**	**1**	...

$D(<M_i>) = 1 - M_i(<M_i>)$

Dial D for Diagonalization!

- **Angenommen: Das TM-Wortproblem ist berechenbar**
- **Dann kann D existieren!**
- **Das führt zu einem Widerspruch!**



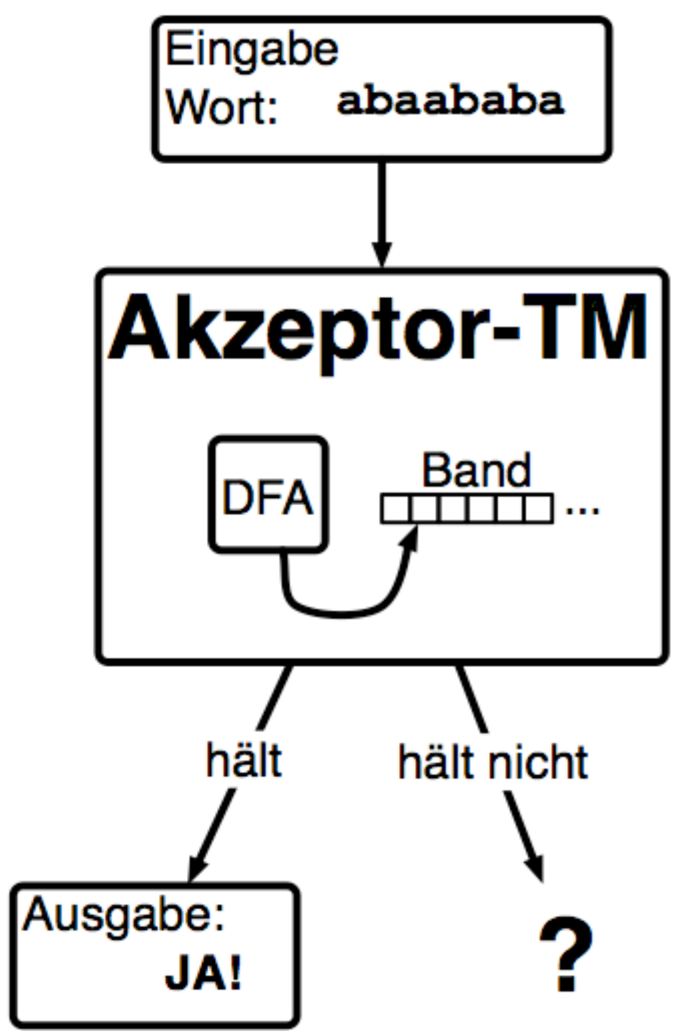
Berechenbarkeitstheorie

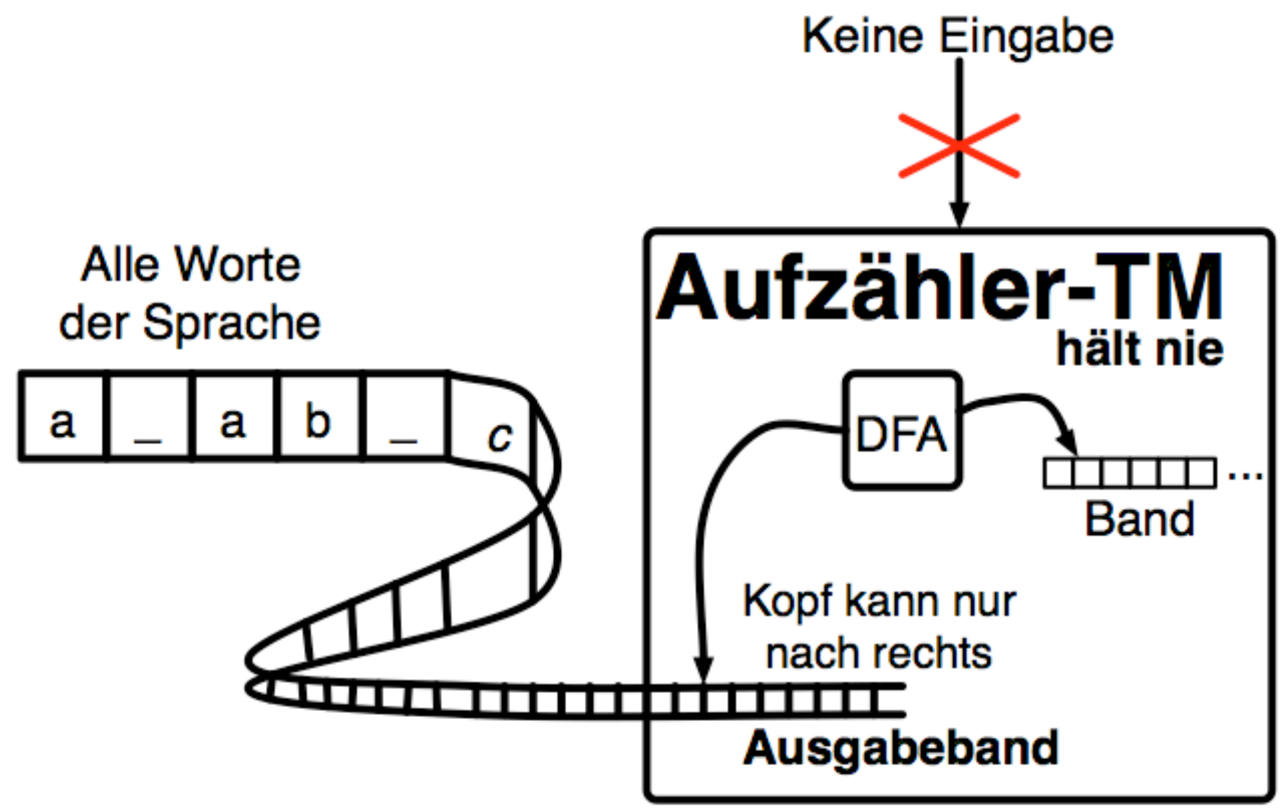
Ein Nichtaufzählbares Problem

19

Wiederholung: Was ist rekursiv aufzählbar?

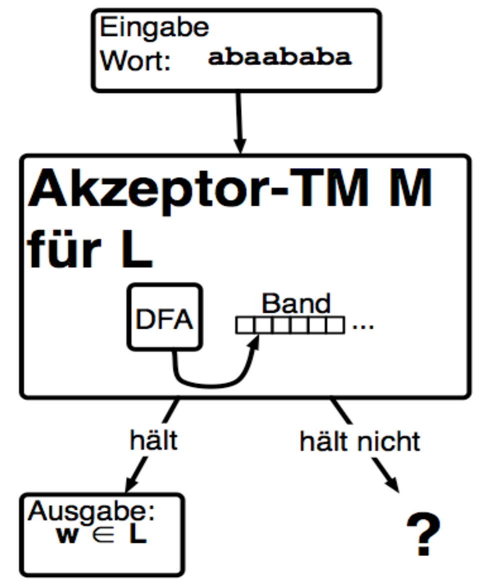
- **oder: Was ist das Komplement einer rekursiv aufzählbaren Sprache?**

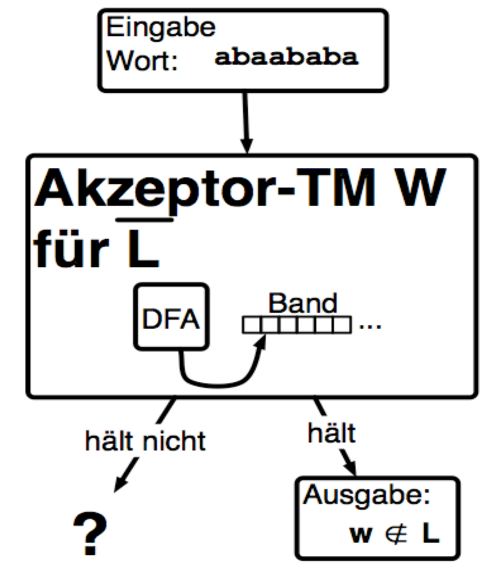


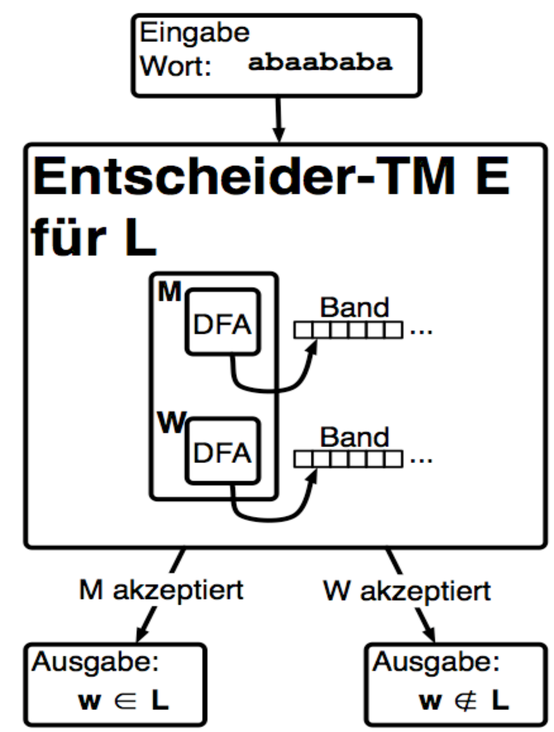


Rekursiv = Aufzählbar + Ko-Aufzählbar

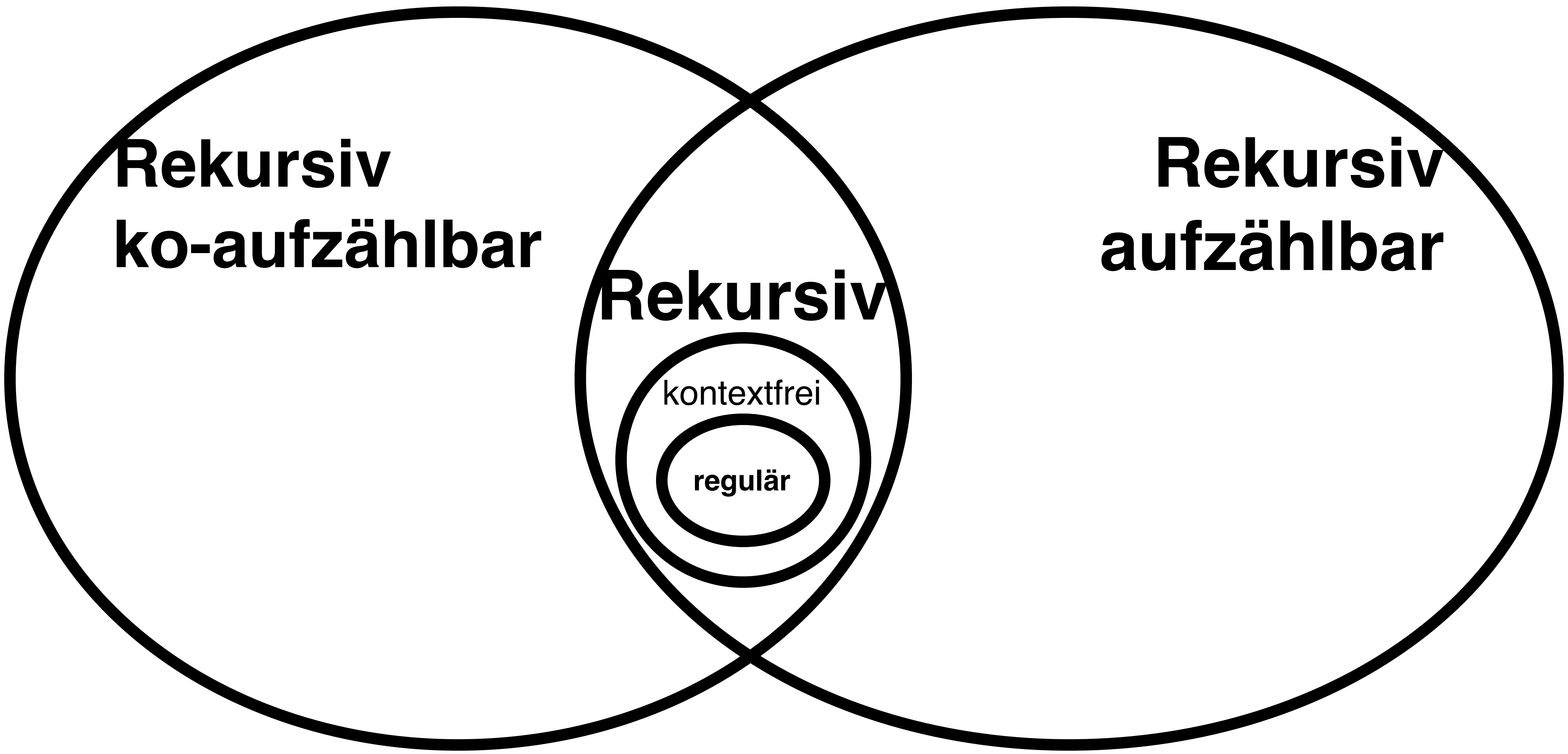
- **Definition**
  - Eine Sprache ist rekursiv ko-aufzählbar, wenn das Komplement der Menge rekursiv aufzählbar ist.
- **Theorem**
  - Eine Sprache L ist rekursiv, genau dann
    - wenn sie rekursiv aufzählbar
    - und rekursiv ko-aufzählbar ist.
- **Beweis (Rückrichtung)**
  - Betrachte Akzeptor-TM M für L und Akzeptor-TM W für Σ* \ L
  - Konstruiere TM E für Eingabe x
    - Berechne parallel M(x) und W(x)
  - Falls M(x) zuerst akzeptiert, akzeptiere
  - Falls W(x) zuerst akzeptiert, halte und verwerfe
- **Beweis (Hinrichtung): einfach**







Überblick


Rekursiv
ko-aufzählbar
Rekursiv
aufzählbar
Rekursiv
kontextfrei
regulär

Das TM-Wortproblem ist rekursiv aufzählbar

- **Theorem**
  - Die Sprache A_{TM} ist rekursiv aufzählbar.
- **Beweis**
  - Betrachte folgende TM A:
  - A = “Für gegebene TM M und Eingabe x
    - Kodiere Anfangskonfiguration für M und x für Simulator S
    - Solange keine akzeptierende Endkonfiguration
      - S simuliert einen Schritt von M
    - Falls akzeptierende Endkonfiguration erreicht wird, halte und akzeptiere”
- **Beobachtung:**
  - A akzeptiert genau dann, wenn M die Eingabe x akzeptiert
  - A ist Akzeptor-TM für A_{TM}
    - Mit A kann eine Aufzähler-TM konstruiert werden

Eine nicht rekursiv aufzählbare Sprache

- **Theorem:**
  - Das Komplement der Sprache A_{TM} des TM-Wortproblems ist nicht rekursiv aufzählbar.
- **Beweis**
  - Angenommen doch.
  - Dann ist A_{TM} rekursiv aufzählbar und rekursiv ko-aufzählbar
  - dann ist A_{TM} rekursiv (also entscheidbar).
  - **Widerspruch**!

Berechenbarkeitstheorie

Das Halteproblem

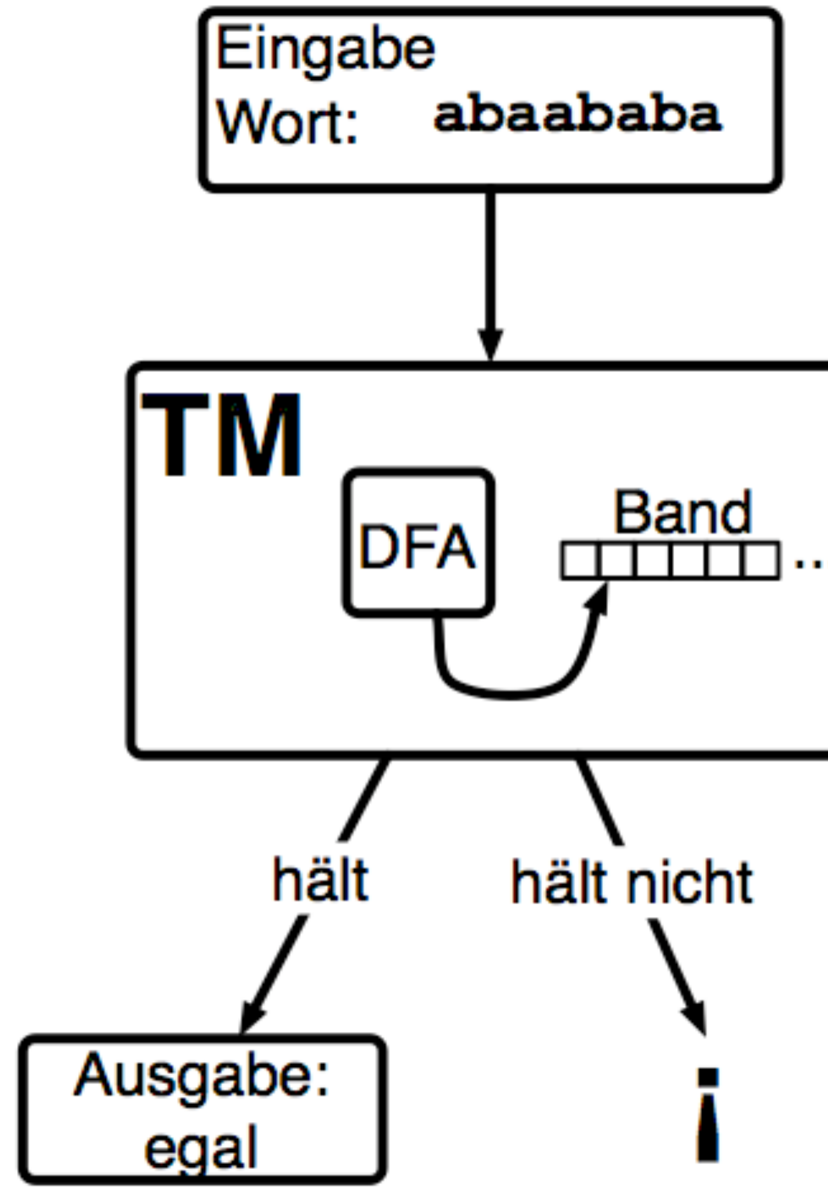
25

Das Halteproblem

- **Definition**
  - Das Halteproblem der Turing-Maschinen ist definiert als
  - Gegeben:
    - eine Turingmaschine M
    - ein Wort w
  - Gesucht:
    - hält die Turingmaschine auf Eingabe w?
- **Alternative Darstellung als Sprache**

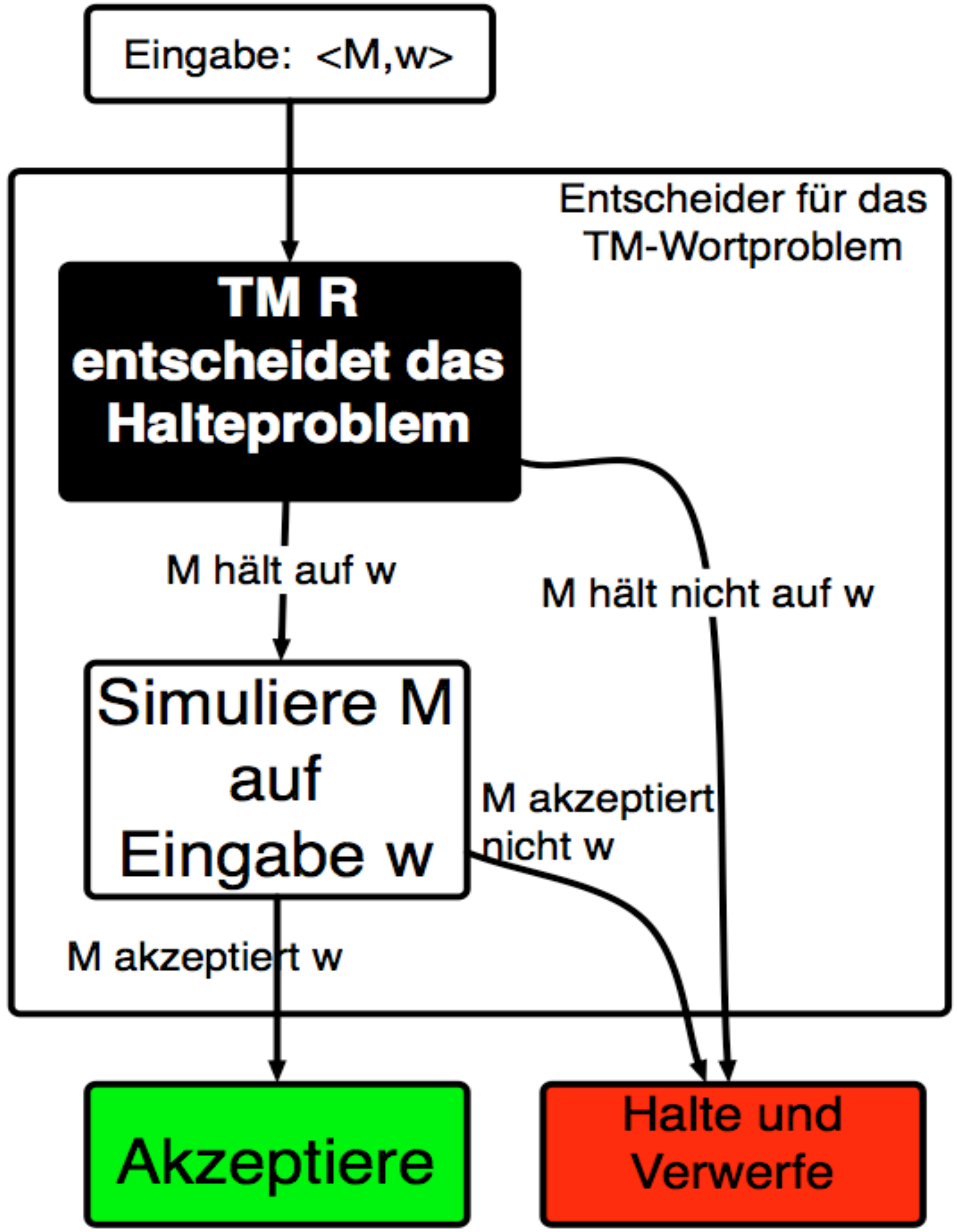
$\mathsf{HALT_{TM}} = \{\langle \mathsf{M}, \mathsf{w} \rangle \mid M \text{ ist eine TM und hält auf Eingabe } w\}$

  - hierbei ist <M,w> eine geeignete Kodierung der TM M und des Wortes w

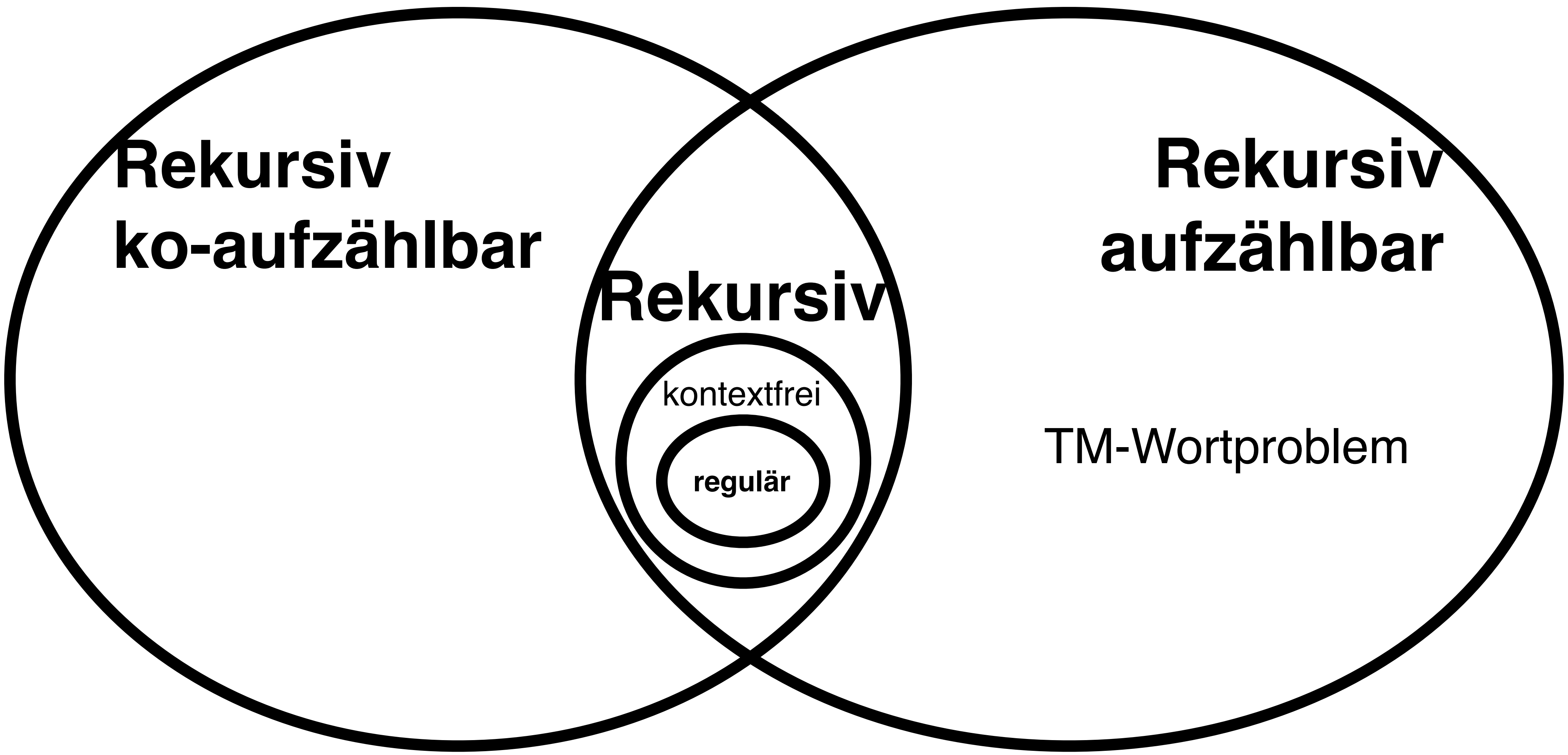


Das Halteproblem ist nicht entscheidbar

- **Theorem**
  - $HALT_{TM}$ ist nicht entscheidbar.
- **Beweis**
  - Angenommen, es gibt eine TM R die $HALT_{TM}$ entscheidet.
  - Konstruiere nun TM S wie folgt:
  - S = “Auf Eingabe <M,w>, Kodierung einer TM und Zeichenkette w
    - Lasse R auf <M,w> laufen
    - Falls R verwirft, verwirft S
    - Falls R akzeptiert,
      - simuliere M auf Eingabe w bis M hält
      - Falls M akzeptiert, dann akzeptiert S
      - Falls M verwirft, dann verwirft S”
  - S entscheidet das (unentscheidbare) TM-Wortproblem
  - Widerspruch ⇒ Annahme ist falsch.



Überblick


Rekursiv
ko-aufzählbar
Rekursiv
aufzählbar
Rekursiv
kontextfrei
regulär
TM-Wortproblem

Berechenbarkeitstheorie

Entscheidbare und Unentscheibare Probleme

29

Kann man entscheiden, ob eine TM überhaupt etwas akzeptiert?

- **Das Leerheitsproblem:**
  - Gegeben
    - eine TM M
  - Gesucht:
    - Akzeptiert M kein einziges Wort?
- **Darstellung als Sprache:**

$$\mathsf{E}_{\mathsf{TM}} = \{\langle \mathsf{M} \rangle \mid M \text{ ist eine TM und } L(M) = \emptyset\}$$

- **Theorem**
  - E_{TM} ist nicht entscheidbar

Beweis für die Unentscheidbarkeit des TM-Leerheitsproblems

- **Theorem**
  - E_{TM} ist nicht entscheidbar

    $$E_{TM} = \{\langle M \rangle \mid M \text{ ist eine TM und } L(M) = \emptyset\}$$

- **Beweis**
  - Für gegebene TM M und ein gegebenes Wort w kann man eine TM $A_{M,w}$ bauen mit folgender Funktionalität
  - $A_{M,w}$ = "Auf Eingabe x:
    - Falls $x \neq w$, verwerfe
    - Falls x = w, simuliere M auf Eingabe w
      - akzeptiere falls M das Wort w akzeptiert"
  - Angenommen TM R entscheidet die Sprache E_{TM}
    - Betrachte nun TM S:
    - S = "Auf Eingabe <M,w> (= Kodierung einer TM und Zeichenkette w)
      - Berechne die Beschreibung der TM $A_{M,w}$
      - Simuliere R auf Eingabe <$A_{M,w}$>
      - Falls R akzeptiert, verwirft S
      - Falls R verwirft, akzeptiert S"
  - S akzeptiert w gdw. R die Eingabe <$A_{M,w}$> verwirft
  - R verwirft gdw. wenn $A_{M,w}$ mindestens ein Wort akzeptiert
  - $A_{M,w}$ akzeptiert gdw. wenn M(w) akzeptiert.
  - Also: S entscheidet das (unentscheidbare) TM-Wortproblem
- **Widerspruch ⇒ Annahme ist falsch**

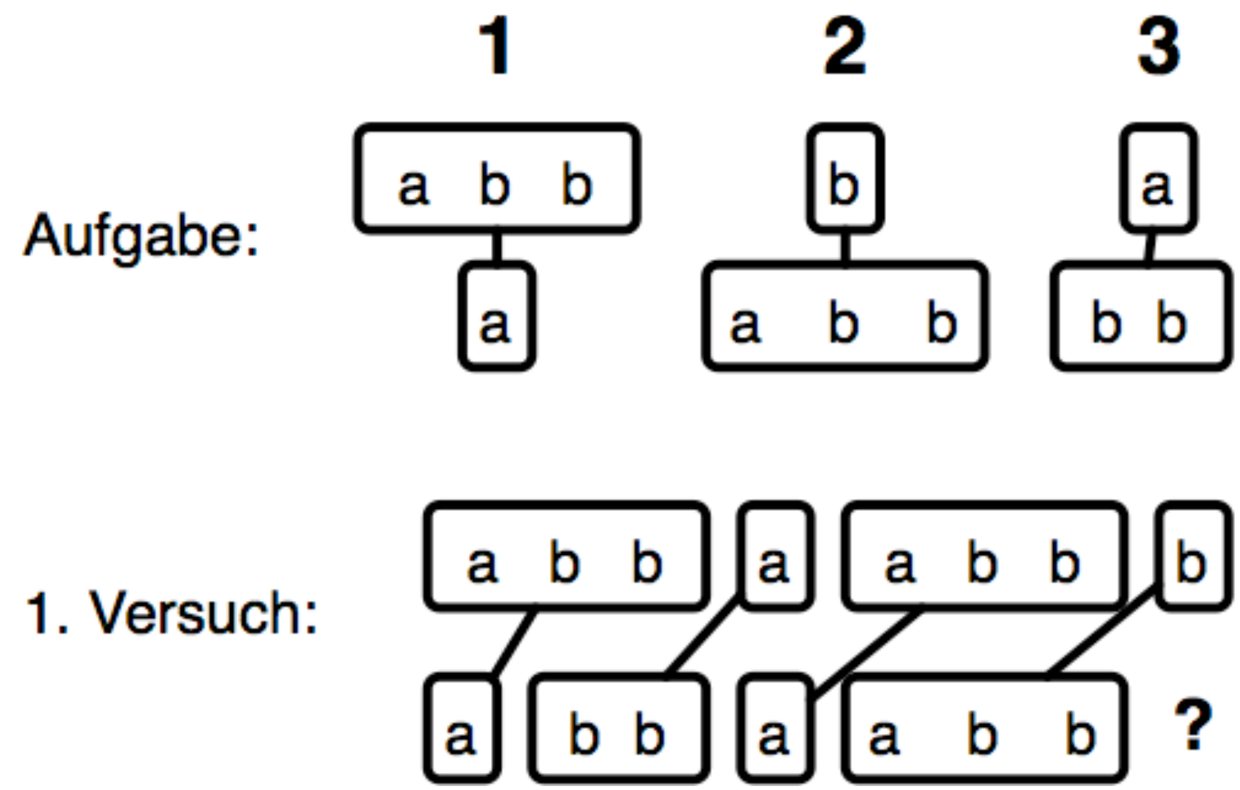
Entscheidbare und unentscheidbare Sprachen

- **Reguläre Sprachen**
  - Wortproblem entscheidbar
  - Leerheitsproblem entscheidbar
  - Äquivalenzproblem entscheidbar
- **Kontextfreie Sprachen**
  - Wortproblem entscheidbar
  - Leerheitsproblem entscheidbar
  - Äquivalenzproblem **nicht entscheidbar**
- **Turing-Maschinen**
  - Wortproblem der TM **nicht entscheidbar**
  - Halteproblem **nicht entscheidbar**
  - Leerheitsproblem der TM **nicht entscheidbar**

Ein einfaches unentscheidbares Problem

- **Das Postsche Korrespondenzproblem**
  - Gegeben
    - die Worte $x_1, x_2, .., x_n$ und
    - die Worte $y_1, y_2, .., y_n$
  - Gesucht
    - Gibt es eine Folge von Indizes $i_1, i_2, ..., i_m$ mit $m \geq 1$
    - so dass:

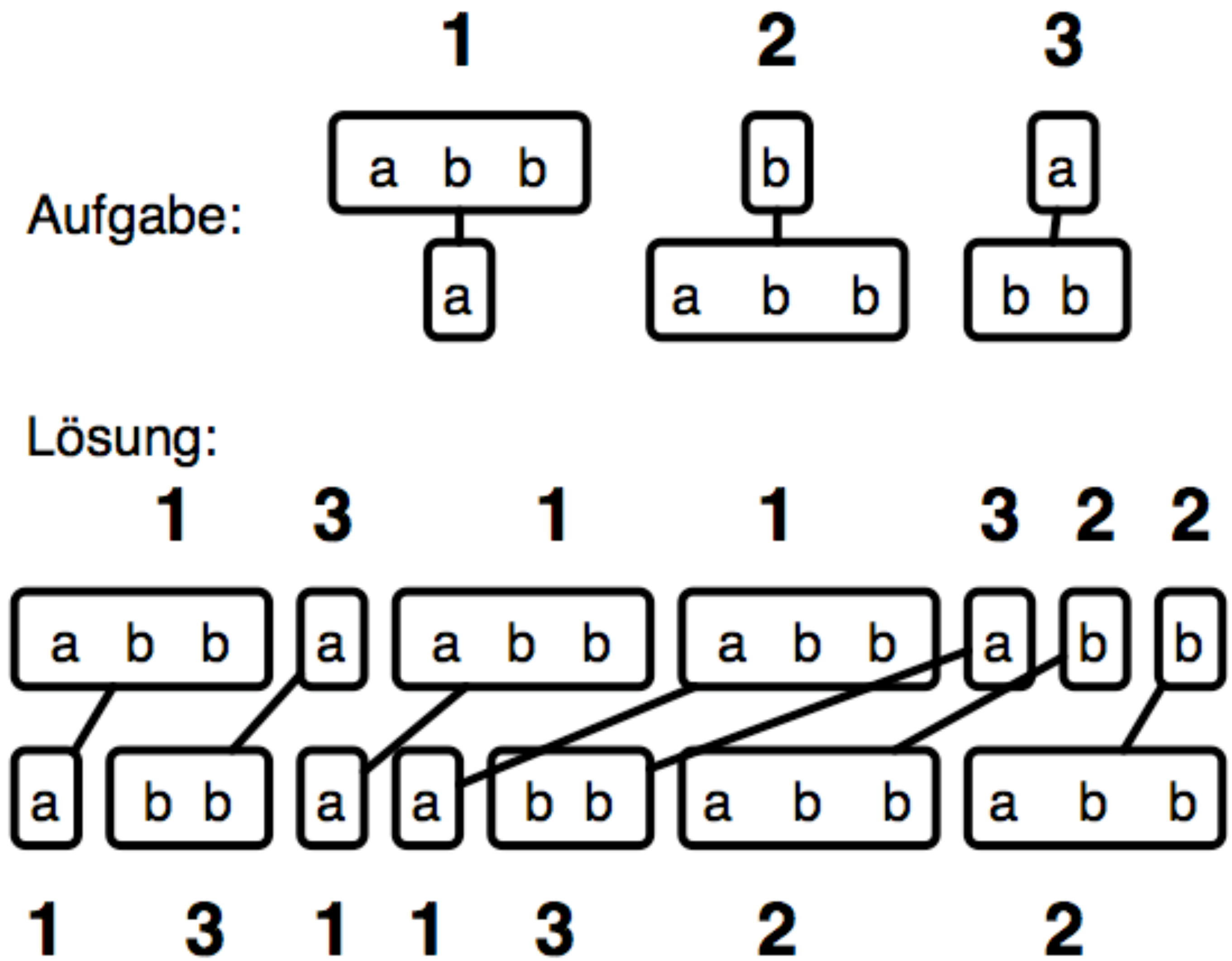
$$x_{i_1} x_{i_2} \dots x_{i_m} = y_{i_1} y_{i_2} \dots y_{i_m}$$



- **Theorem**
  - Das Postsche Korrespondenzproblem ist nicht entscheidbar.

Eine Lösung und eine Anregung

- **Der Beweis der Nichtentscheidbarkeit des Postschen Korrespondenz-Problem**
  - kann in Sipsers Buch “Introduction to the Theory of Computation” nachgelesen werden
  - sehr empfehlenswert



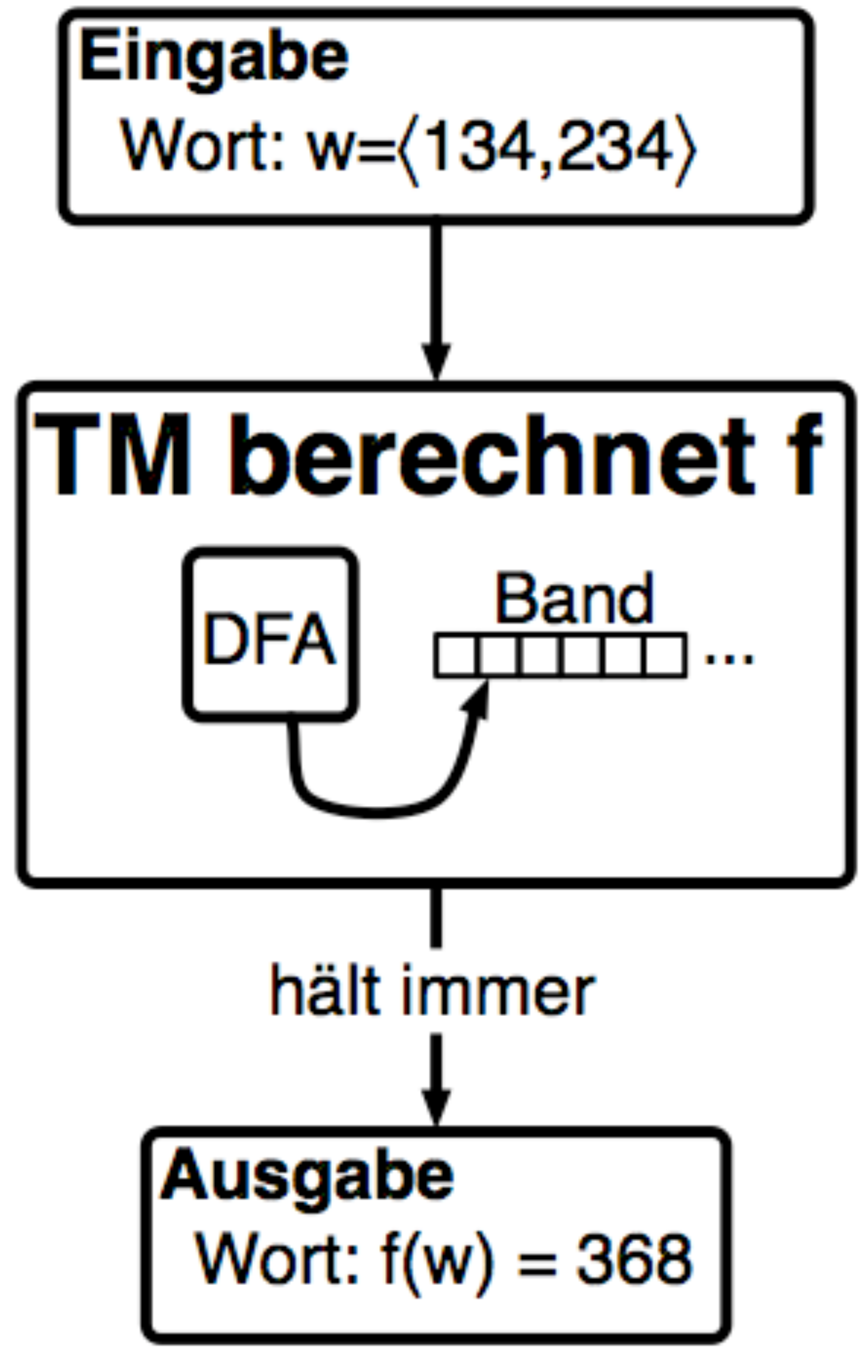
Berechenbarkeitstheorie

Reduktion und Entscheidbarkeit

35

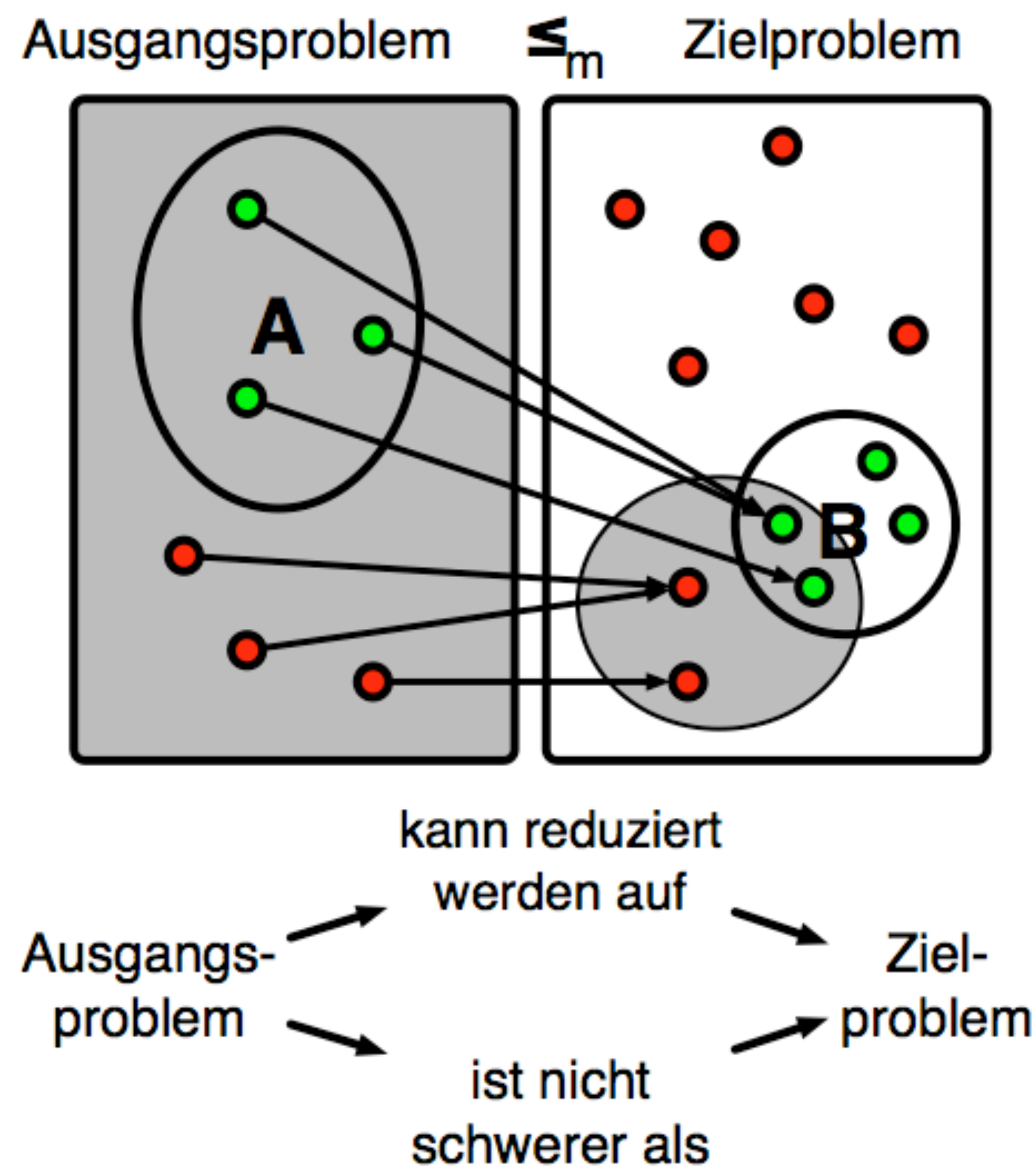
Berechenbare Funktionen

- **Definition**
  - Eine Funktion f: $\Sigma^* \rightarrow \Sigma^*$ ist berechenbar, falls eine Turing-Maschine für jede Eingabe w mit dem Ergebnis f(w) auf dem Band hält.
- **Beispiele für berechenbare Funktionen:**
  - Addition, Division, Multiplikation, Vergleich, Sortieren, Division, ...
  - Automatische Generierung von Kodierungen für bestimmte Turingmaschinen
  - Modifizierung der Kodierung einer TM:
    - Kartesisches Produkt
    - Invertierung von Zuständen
    - Initialisierung der Eingabe
    - Verknüpfung mehrerer kodierter TMs



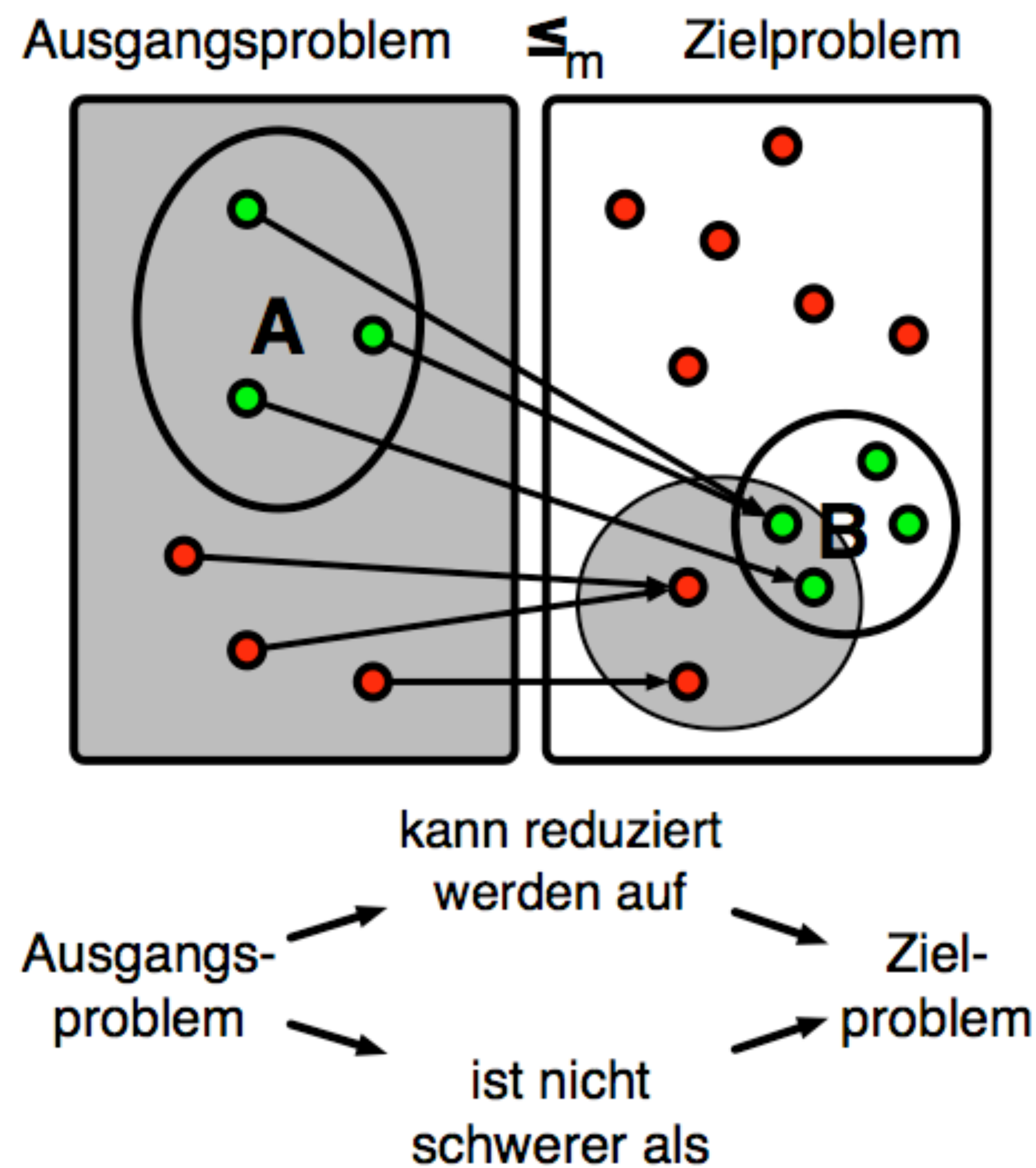
Abbildungsreduktion

- **Definition (Abbildungsreduktion, Mapping Reduction, Many-one)**
  - Eine Sprache A kann durch Abbildung auf eine Sprache B reduziert werden: **A** $\leq_m$ **B**,
    - falls es eine berechenbare Funktion f: $\Sigma^* \rightarrow \Sigma^*$ gibt,
    - so dass für alle w: $w \in A \Leftrightarrow f(w) \in B$
  - Die Funktion f heißt die **Reduktion** von A auf B.



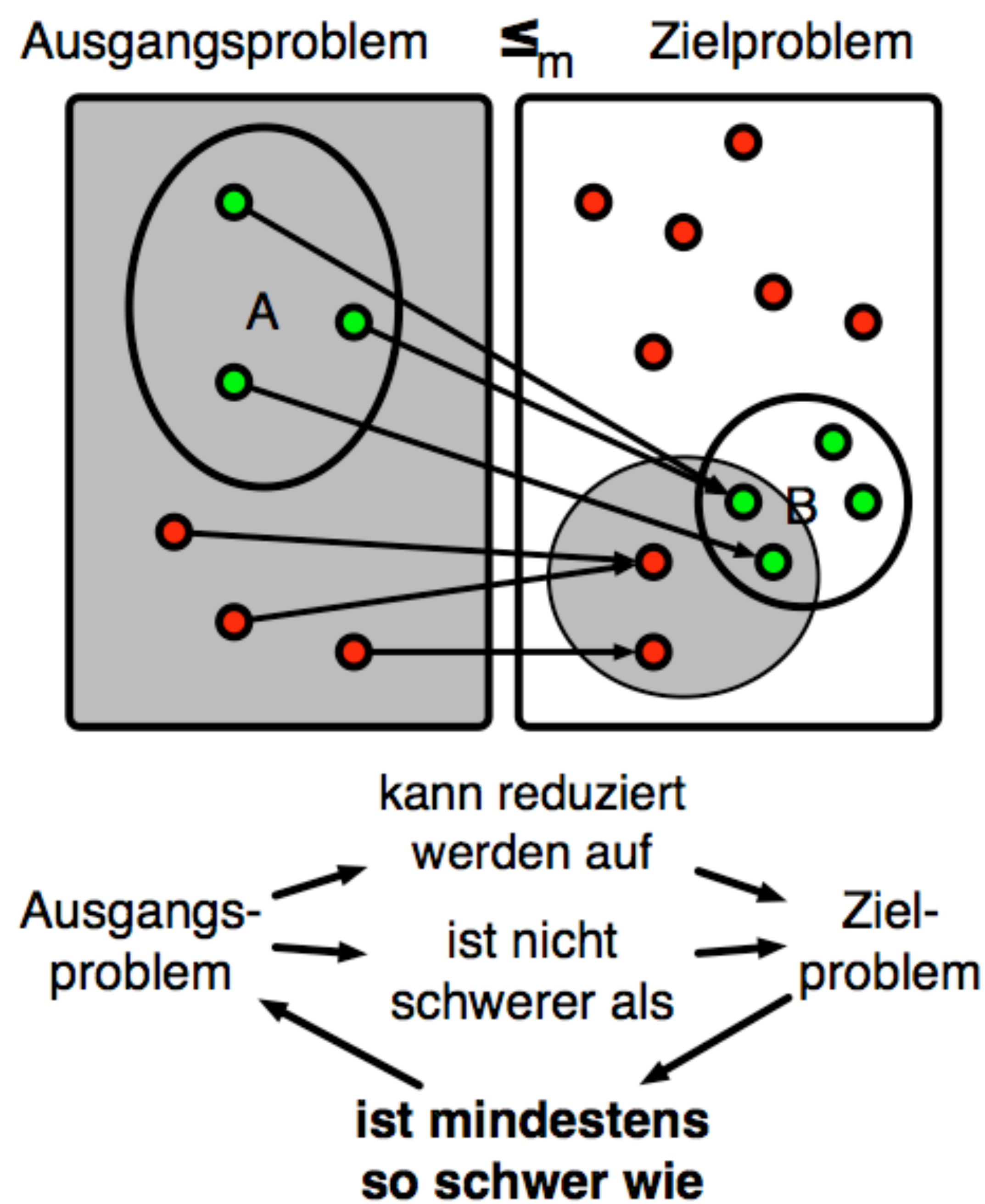
Der Nutzen der Reduktionen

- **Theorem**
  - Falls A $\leq_m$ B und B ist entscheidbar, dann ist A entscheidbar.
- **Beweis**
  - Sei M, eine Turing-Maschine, die B entscheidet.
  - Betrachte die Entscheider-TM:
  - N = “Auf Eingabe w:
    - Berechne f(w)
    - Führe die Berechnung von M auf Eingabe f(w) durch
    - N gibt aus, was M ausgibt”
  - Falls f(w) $\in$ B,
    - dann akzeptiert M
    - dann ist auch w $\in$ A
  - Falls f(w) $\notin$ B,
    - dann akzeptiert M nicht
    - dann ist auch w $\notin$ A



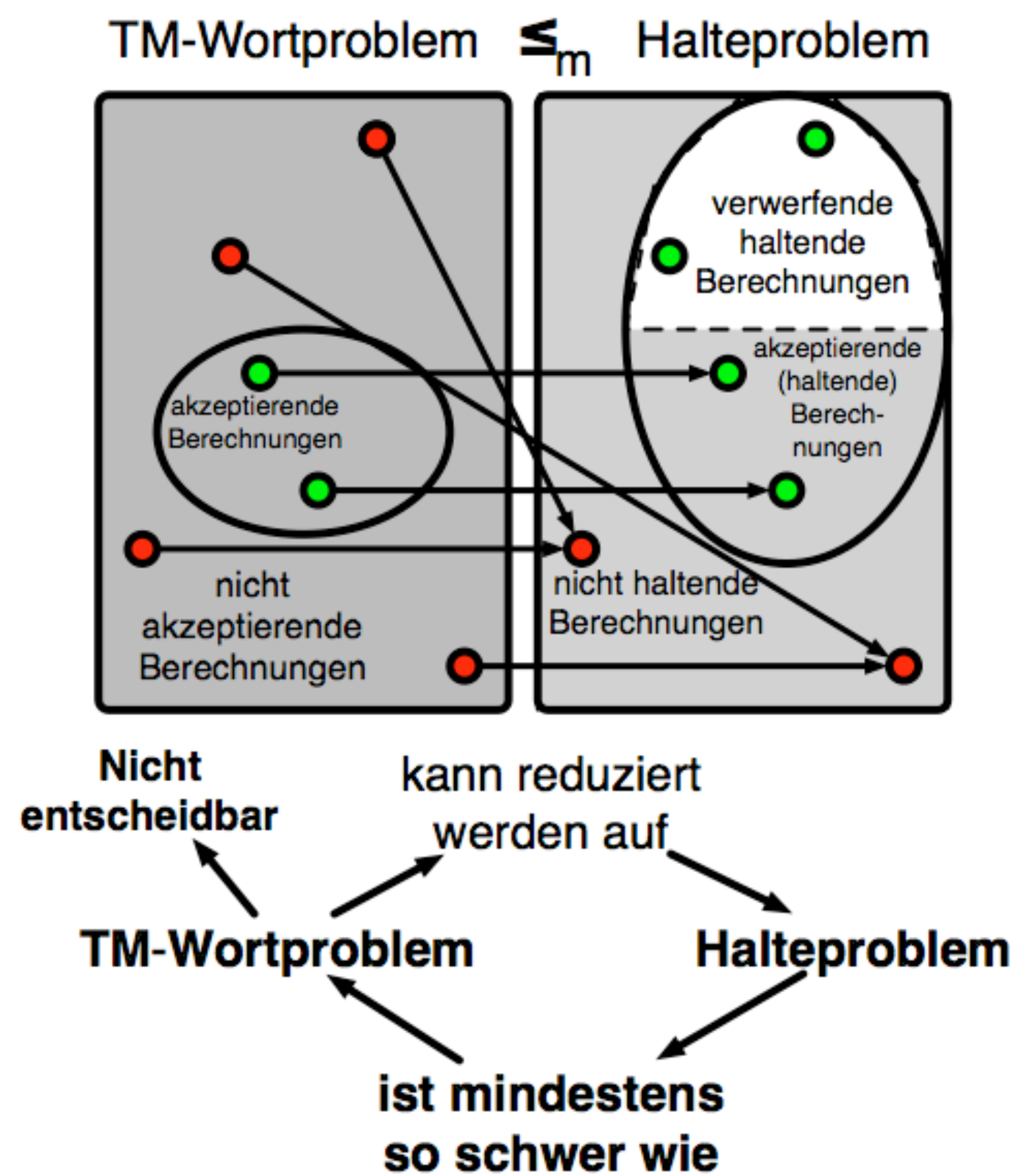
Der Nutzen der Reduktionen

- **Korollar**
  - Falls A $\leq_m$ B und A ist nicht entscheidbar, dann ist B auch nicht entscheidbar.
- **Folgt aus:**
  - $X \wedge Y \Rightarrow Z$ und $X \wedge \neg Y \Rightarrow \neg Z$ sind äquivalent
- **Dieses Korollar ist unser Hauptwerkzeug für den Beweis der Nichtberechenbarkeit**



Ein alternativer Beweis für die Nichtberechenbarkeit des Halteproblems

- **Theorem:**
  - $A_{TM} \leq_m HALT_{TM}$
- **Beweis**
  - Betrachte Reduktionsfunktion F:
  - F = “Auf Eingabe <M, w>:
    - Konstruiere TM M’:
      - M’ = “Auf Eingabe x:
        - Führe M auf Eingabe x aus
        - Falls M akzeptiert, akzeptiert M’
        - Falls M verwirft, hält M’ nicht”
    - F gibt <M’,w> aus”



Ein alternativer Beweis für das Leerheitsproblem der TM

- **Theorem**

  $E_{TM} = \{\langle M \rangle \mid M \text{ ist eine TM und } L(M) = \emptyset\}$

  - E_{TM} ist nicht entscheidbar
- **Beweis: $A_{TM} \leq_m \overline{E_{TM}}$**
  - Betrachte Reduktionsfunktion F:
  - F = “Auf Eingabe <M, w>:
    - Konstruiere TM M’:
      - M’ = “Für jede Eingabe x:
        - Führe M auf w aus
        - Falls M akzeptiert, akzeptiert M’
        - Falls M verwirft, verwirft M’ ”
    - F gibt M’ aus”
- **Zu zeigen:**
  - F ist eine berechenbare Funktion
    - M’ kann effektiv aus M und w konstruiert werden
- **$\langle M, w\rangle \in A_{TM} \Leftrightarrow F(\langle M, w\rangle) \in \overline{E_{TM}}$**
  - Falls $\langle M, w\rangle \in A_{TM}$
    - dann ist $L(M') = \Sigma^*$
    - dann ist $F(\langle M, w\rangle) \notin E_{TM}$
  - Falls $\langle M, w\rangle \notin A_{TM}$
    - dann ist $L(M') = \emptyset$
    - dann ist $F(\langle M, w\rangle) \in E_{TM}$
- **Nun ist $A_{TM} \leq_m \overline{E_{TM}}$**
  - da A_{TM} nicht rekursiv ist
  - ist auch $\overline{E_{TM}}$ nicht rekursiv
- **damit ist auch E_{TM} nicht rekursiv**

Berechenbarkeitstheorie

Reduktion und Aufzählbarkeit

42

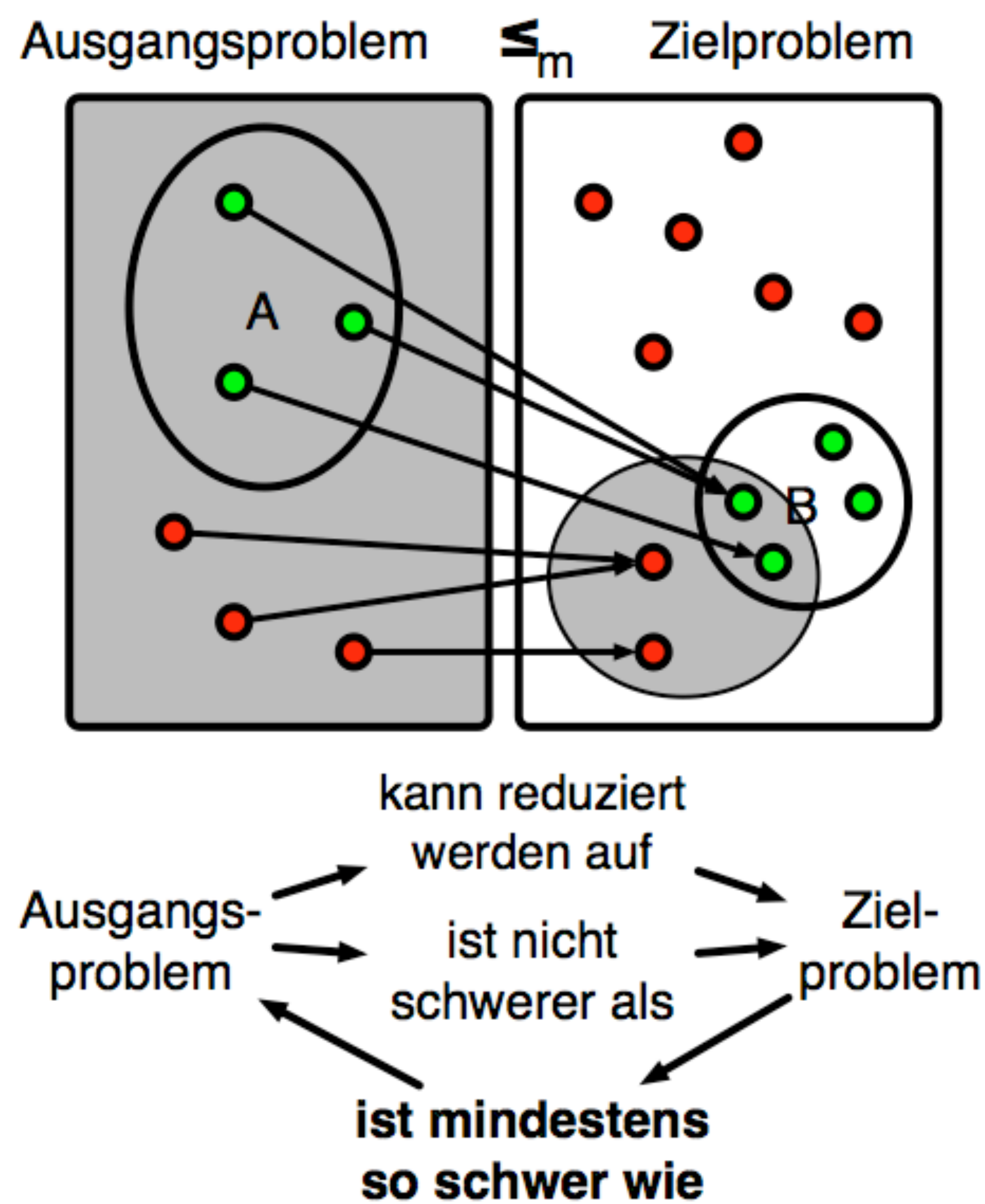
Reductionen und Rekursive Aufzählbarkeit

- **Theorem**
  - Falls A $\leq_m$ B und B ist rekursiv aufzählbar, dann ist A rekursiv aufzählbar.
- **Beweis**
  - Sei M, eine Turing-Maschine, die B akzeptiert.
  - Betrachte die Akzeptor-TM N:
  - N = “Auf Eingabe w:
    - Berechne f(w)
    - Führe die Berechnung von M auf Eingabe f(w) durch
    - N gibt aus, was M ausgibt”
  - Falls f(w) $\in$ B,
    - dann akzeptiert M
    - dann ist auch w $\in$ A
  - Falls f(w) $\notin$ B,
    - dann akzeptiert M nicht
    - dann ist auch w $\notin$ A

Nicht-Rekursive Aufzählbarkeit und Reduktionen

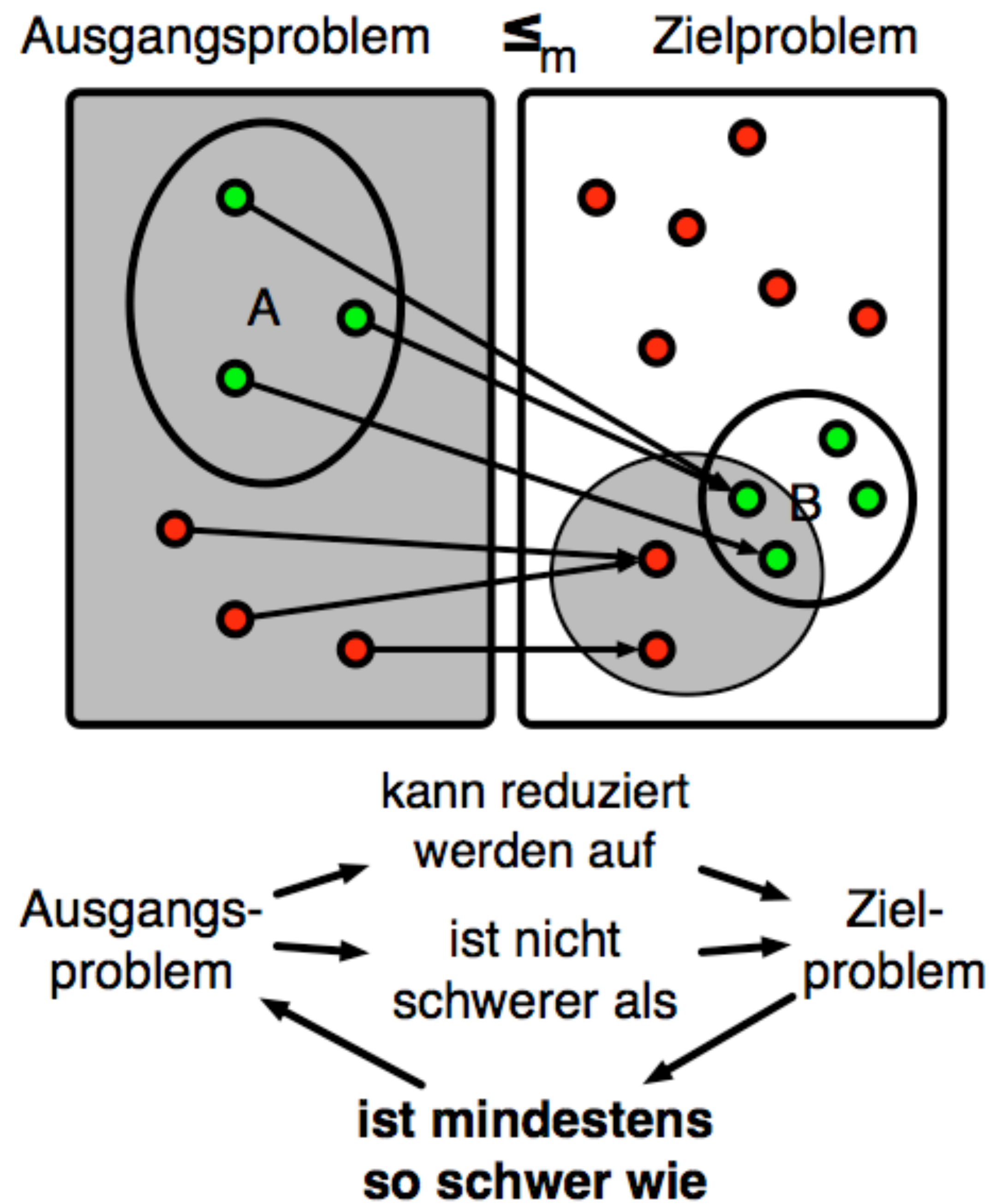
- **Theorem**
  - Falls A $\leq_m$ B und B ist rekursiv aufzählbar, dann ist A rekursiv aufzählbar.

- **Korollar**
  - Falls A $\leq_m$ B und A ist nicht rekursiv aufzählbar, dann ist B nicht rekursiv aufzählbar.



Zusammenfassung: Abbildungsreduktionen

- **Eine Sprache A ist kann durch Abbildung auf eine Sprache B reduziert werden: $A \leq_m B$,**
  - falls es eine berechenbare Funktion
    $f: \Sigma^* \rightarrow \Sigma^*$ gibt, so dass für alle w:
    $w \in A \Leftrightarrow f(w) \in B$
  - f ist die Reduktionsfunktion von A auf B.
- **Es gilt**
  - Falls $A \leq_m B$ und B ist entscheidbar, dann ist A entscheidbar.
  - Falls $A \leq_m B$ und A ist nicht entscheidbar, dann ist B auch nicht entscheidbar.
  - Falls $A \leq_m B$ und B ist rekursiv aufzählbar, dann ist A rekursiv aufzählbar.
  - Falls $A \leq_m B$ und A ist nicht rekursiv aufzählbar, dann ist B nicht rekursiv aufzählbar.



Berechenbarkeitstheorie

Das Äquivalenz-problem

46

Ein nicht rekursiv aufzählbares und nicht rekursiv ko-aufzählbares Problem

- **Definition**
  - Das TM-Äquivalenzproblem
    - Gegeben: TM M_1 und TM M_2
    - Gesucht: Ist $L(M_1) = L(M_2)$?
  - Definition als Sprache:

$$\mathrm{EQ}_{\mathrm{TM}} = \{\langle \mathsf{M}_1, \mathsf{M}_2\rangle \mid \mathsf{M}_1, \mathsf{M}_2 \text{ sind TMs und } L(M_1) = L(M_2)\}$$

- **Theorem**
  - EQ_{TM} ist weder rekursiv aufzählbar noch rekursiv ko-aufzählbar.
- **Beweisidee:**
  - Reduktion: $A_{TM} \leq_m EQ_{TM}$
    - äquivalent zu $\overline{A_{TM}} \leq_m \overline{EQ_{TM}}$
    - beweist, dass $\overline{EQ_{TM}}$ nicht rekursiv aufzählbar ist
  - Reduktion: $\overline{A_{TM}} \leq_m EQ_{TM}$
    - äquivalent zu $A_{TM} \leq_m \overline{EQ_{TM}}$
    - beweist, dass EQ_{TM} nicht rekursiv aufzählbar ist

$A_{TM} \leq_m \overline{EQ_{TM}}$

- **Reduktionsfunktion: F**
- **F = “Auf Eingabe <M,w>, wobei M eine TM ist und w ein Wort**
  - Konstruiere Maschinen M_1 und M_2 wie folgt
    - M_1 = *“Für jede Eingabe:*
      - *Verwerfe”*
    - M_2 = *“Für jede Eingabe:*
      - *Führe M auf w aus*
      - *Falls M akzeptiert, akzeptiert M_2*
      - *Ansonsten verwirft M_2”*
  - F gibt $\langle M_1, M_2 \rangle$ aus”

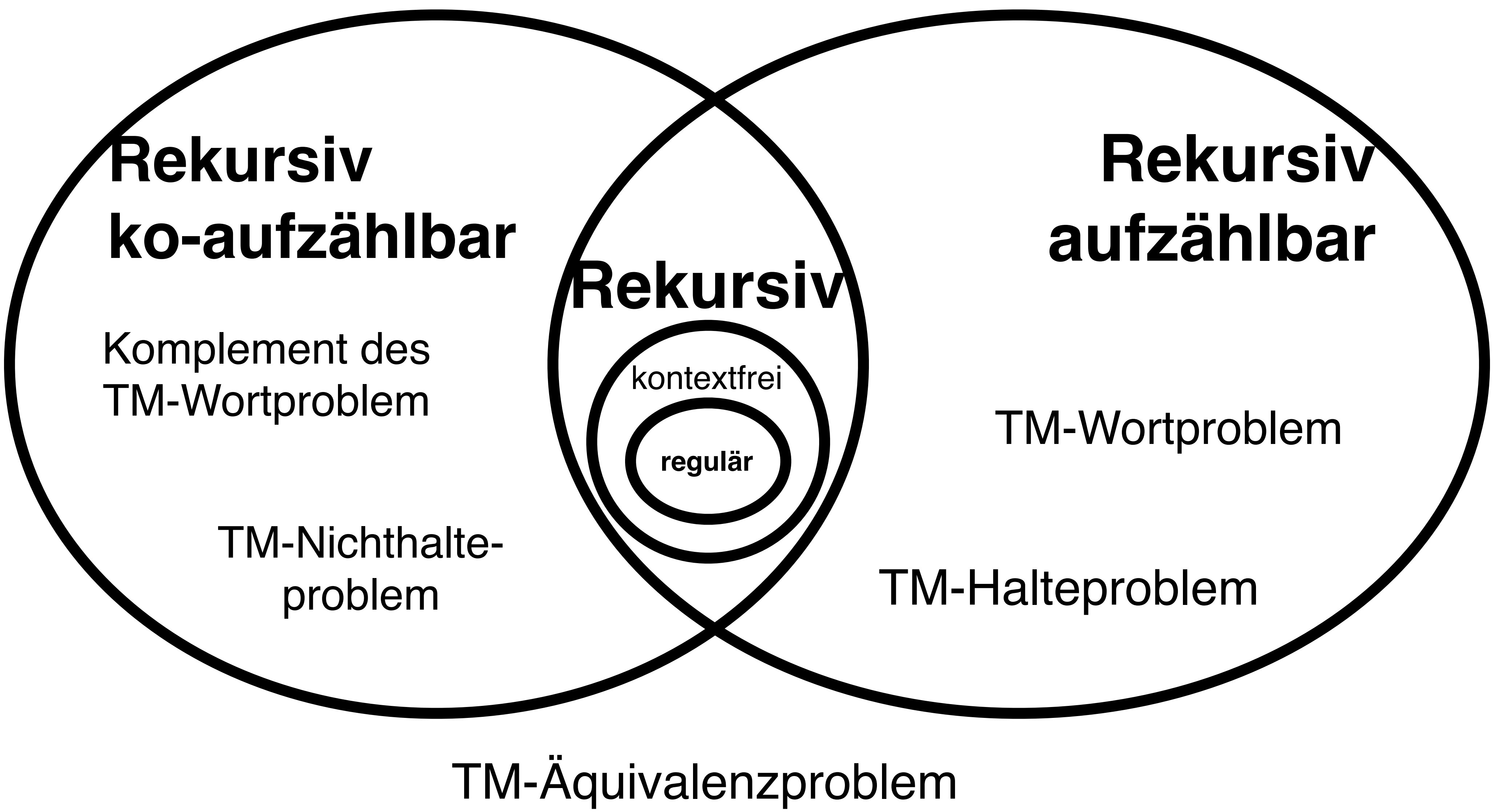
- **Zu beweisen:**
  - F ist berechenbar
    - die Kodierung der TM kann automatisch erfolgen
  - $\langle M,w \rangle \in A_{TM} \Leftrightarrow F(\langle M,w \rangle) \in \overline{EQ_{TM}}$
  - M ist TM und $F(\langle M,w \rangle) = \langle M_1, M_2 \rangle$
    - wobei $L(M_1) = \emptyset$ und
  - $L(M_2) = \Sigma^*$, falls M(w) akzeptiert
  - $L(M_2) = \emptyset$, falls M(w) nicht akzeptiert
- **Daraus folgt:**
  - $\langle M,w \rangle \in A_{TM} \Leftrightarrow F(\langle M,w \rangle) \notin EQ_{TM}$

$A_{TM} \leq_m EQ_{TM}$

- **Reduktionsfunktion: F**
- **F = “Auf Eingabe <M,w>, wobei M eine TM ist und w ein Wort**
  - Konstruiere Maschinen M_1 und M_2 wie folgt
    - M_1 = *“Für jede Eingabe:*
      - *Akzeptiere”*
    - M_2 = *“Für jede Eingabe:*
      - *Führe M auf w aus*
      - *Falls M akzeptiert, akzeptiert M_2”*
  - F gibt $\langle M_1, M_2\rangle$ aus”

- **Zu beweisen:**
  - F ist berechenbar
    - die Kodierung der TM kann automatisch erfolgen
  - $\langle M,w\rangle \in A_{TM} \Leftrightarrow F(\langle M,w\rangle) \in EQ_{TM}$
  - M ist TM und $F(\langle M,w\rangle) = \langle M_1, M_2\rangle$
    - wobei $L(M_1) = \Sigma^*$ und
  - $L(M_2) = \Sigma^*$, falls M(w) akzeptiert
  - $L(M_2) = \emptyset$, falls M(w) nicht akzeptiert
- **Daraus folgt:**
  - $\langle M,w\rangle \in A_{TM} \Leftrightarrow F(\langle M,w\rangle) \in EQ_{TM}$

Überblick


Rekursiv
ko-aufzählbar
Rekursiv
Rekursiv
aufzählbar
Komplement des
TM-Wortproblem
kontextfrei
regulär
TM-Wortproblem
TM-Nichthalte-
problem
TM-Halteproblem
TM-Äquivalenzproblem

3.2. Berechenbarkeit

Ende

51