

**Vorlesung**

# **Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I**

**Bernhard Beckert**

**Institut für Informatik**



**Sommersemester 2007**

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

**Katrin Erk** (gehalten an der Universität Koblenz-Landau)

**Jürgen Dix** (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

*– Bernhard Beckert, April 2007*

# Teil V

- 1 Determinierte Turing-Maschinen (DTMs)
- 2 Varianten von Turing-Maschinen
- 3 Indeterminierte Turing-Maschinen (NTMs)
- 4 Universelle determinierte Turing-Maschinen
- 5 Entscheidbar/Aufzählbar**
- 6 Determinierte Turing-Maschinen entsprechen Typ 0
- 7 Unentscheidbarkeit

## Akzeptieren

Eine DTM **akzeptiert** eine Sprache  $L$ , wenn sie

- für jedes Eingabe-Wort  $w \in L$  irgendwann hält
- für jedes Wort  $v \notin L$  unendlich lang rechnet oder hängt

## Entscheiden

Eine DTM **entscheidet** eine Sprache  $L$ , wenn sie

- für jedes Eingabe-Wort  $w \in L$  hält mit dem Bandinhalt  $Y$  ("Yes")
- für jedes Wort  $v \notin L$  hält mit dem Bandinhalt  $N$  ("No")

## Definition 13.1 (Entscheidbar)

$L$  sei eine Sprache über  $\Sigma_0$  mit  $\#, N, Y \notin \Sigma_0$ .

$M = (K, \Sigma, \delta, s)$  sei eine DTM mit  $\Sigma_0 \subseteq \Sigma$ .

$M$  **entscheidet**  $L$ , falls für alle  $w \in \Sigma_0^*$  gilt:

$$s, \#w\# \vdash_M^* \begin{cases} h, \#Y\# & \text{falls } w \in L \\ h, \#N\# & \text{sonst} \end{cases}$$

$L$  heißt **entscheidbar**, falls es eine DTM gibt, die  $L$  entscheidet.

## Definition 13.2 (Akzeptierbar)

$L$  sei eine Sprache über  $\Sigma_0$  mit  $\#, N, Y \notin \Sigma_0$ .

$M = (K, \Sigma, \delta, s)$  sei eine DTM mit  $\Sigma_0 \subseteq \Sigma$ .

$\mathcal{M}$  **akzeptiert** ein Wort  $w \in \Sigma_0^*$ ,  
falls  $\mathcal{M}$  bei Input  $w$  hält.

$\mathcal{M}$  **akzeptiert die Sprache  $L$** , falls für alle  $w \in \Sigma_0^*$  gilt:

$\mathcal{M}$  akzeptiert  $w$  *genau dann wenn*  $w \in L$

$L$  heißt **akzeptierbar** (oder auch **semi-entscheidbar**),  
falls es eine DTM gibt, die  $L$  akzeptiert.

## Definition 13.3 (Rekursiv Aufzählbar (recursively enumerable))

$L$  sei eine Sprache über  $\Sigma_0$  mit  $\#, N, Y \notin \Sigma_0$ .

$M = (K, \Sigma, \delta, s)$  sei eine DTM mit  $\Sigma_0 \subseteq \Sigma$ .

$\mathcal{M}$  **zählt  $L$  auf**, falls es einen Zustand  $q_B \in K$  gibt (den **Blinkzustand**), so daß:

$$L = \{w \in \Sigma_0^* \mid \exists u \in \Sigma^* : s, \# \vdash_{\mathcal{M}}^* q_B, \#w\#u\}$$

$L$  heißt **rekursiv aufzählbar**, falls es eine DTM gibt, die  $L$  aufzählt.

**Achtung:** aufzählbar  $\neq$  abzählbar.

## Unterschied

**$M$  abzählbar:** Es gibt eine surjektive Abbildung der natürlichen Zahlen auf  $M$

**$M$  aufzählbar:** Diese Abbildung kann von einer Turing-Maschine berechnet werden.

Wegen Endlichkeit der Wörter und des Alphabets sind alle Sprachen abzählbar.

Aber nicht alle Sprachen sind aufzählbar.



## Beispiel 13.4 (Rekursiv aufzählbar aber nicht entscheidbar)

Folgende Mengen sind rekursiv aufzählbar aber *nicht* entscheidbar:

- Die Menge der Gödelisierungen aller haltenden Turing-Maschinen
- Die Menge aller terminierenden Programme
- Die Menge aller allgemeingültigen prädikatenlogischen Formeln

## Satz 13.5 (Akzeptierbar = Rekursiv Aufzählbar)

*Eine Sprache ist genau dann rekursiv aufzählbar, wenn sie akzeptierbar ist.*

## Beweis

“ $\Rightarrow$ ”

Sei  $L$  rekursiv aufzählbar

Es gibt also eine DTM  $\mathcal{M}_L$ , die  $L$  aufzählt.

Zu zeigen:  $L$  ist akzeptierbar.

## Beweis (Fortsetzung)

Wir konstruieren aus  $\mathcal{M}_L$  eine 2-DTM  $\mathcal{M}$ , die  $L$  akzeptiert:

- $\mathcal{M}$  wird gestartet mit

$$s_0, \quad \#w\underline{\#}$$
$$\quad \underline{\#}$$

- $\mathcal{M}$  simuliert auf Band 2 die Maschine  $\mathcal{M}_L$ .
- Wenn  $\mathcal{M}_L$  den **Blinkzustand**  $q_B$  erreicht, dann enthält Band 2 von  $\mathcal{M}$  ein Wort

$$\#w'\underline{\#}u$$

mit  $w' \in L$ .

## Beweis (Fortsetzung)

- Nach Erreichen des Blinkzustands:  
 $\mathcal{M}$  vergleicht  $w$  und  $w'$ .
  - Falls  $w = w'$ , dann hält  $\mathcal{M}$ :  $w \in L$ .
  - Ansonsten simuliert  $\mathcal{M}$  auf Band 2 weiter die Arbeit von  $\mathcal{M}_L$ .
- Wenn  $\mathcal{M}_L$  hält, ohne das Wort  $w$  auf Band 2 erzeugt zu haben, gerät  $\mathcal{M}$  in eine Endlosschleife.

## Beweis (Fortsetzung)

“ $\Leftarrow$ ”

Sei  $L$  akzeptierbar

Es gebe also eine DTM  $\mathcal{M}_L$ , die  $L$  akzeptiert.

Zu zeigen:  $L$  ist rekursiv aufzählbar.

Wir konstruieren eine DTM  $\mathcal{M}$ , die  $L$  rekursiv aufzählt.

Grundidee:

- die Wörter aus  $\Sigma^*$  der Reihe nach aufzählen
- jedes Wort der Maschine  $\mathcal{M}_L$  vorlegen
- wenn  $\mathcal{M}_L$  das Wort akzeptiert, in den Blinkzustand  $q_B$  gehen

## Beweis (Fortsetzung)

**Problem:**  $\mathcal{M}_L$  akzeptiert, sie entscheidet nicht.

Wenn  $\mathcal{M}_L$  ein Wort nicht akzeptiert, rechnet sie unendlich.

**Lösung:** Wir betrachten die Rechnung von  $\mathcal{M}_L$  zu allen Wörtern aus  $\Sigma^*$   
**gleichzeitig.**

## Beweis (Fortsetzung)

$\Sigma^* = \{w_1, w_2, w_3, \dots\}$  (in lexikalischer Reihenfolge aufgezählt).

Dann rechnet  $\mathcal{M}$  so:

- Im ersten Durchlauf berechne den ersten Rechenschritt von  $\mathcal{M}_L$  für  $w_1$ .
- Im zweiten Durchlauf berechne
  - die ersten zwei Rechenschritte von  $\mathcal{M}_L$  für  $w_1$ ,
  - einen Rechenschritt für  $w_2$ .
- Im dritten Durchlauf berechne drei Rechenschritte für  $w_1$ , zwei für  $w_2$  und einen für  $w_3$  und so weiter.

Immer wieder bei der Startkonfiguration anfangen: So müssen wir uns den Bandinhalt der Rechnung von  $\mathcal{M}_L$  zu  $w_i$  nach  $j$  Schritten nicht merken.



## Beweis (Fortsetzung)

Wenn  $\mathcal{M}$  so rechnet, dann gilt:

- $\mathcal{M}$  fängt für jedes  $w_i \in \Sigma^*$  in endlicher Zeit (nämlich im  $i$ -ten Durchlauf) an, die Arbeit von  $\mathcal{M}_L$  zu  $w_i$  zu simulieren, und
- falls  $w_i \in L$  und falls  $\mathcal{M}_L$ , gestartet mit  $s, \#w_i\#$ , in  $j$  Schritten einen Haltezustand erreicht, dann erreicht  $\mathcal{M}$  nach endlicher Zeit (nämlich im  $i + j$ -ten Durchlauf) den Haltezustand von  $\mathcal{M}_L$  in der Rechnung zu  $w_i$ .

## Beweis (Ende)

- Wenn  $\mathcal{M}$  bei Simulation von  $\mathcal{M}_L$  zur Eingaben  $w_i$  auf eine Haltekonfiguration trifft, dann ist  $w_i \in L$ .
- $\mathcal{M}$  nimmt dann eine Konfiguration

$$q_B, \#w_i\#u_i$$

ein –  $q_B$  ist der Blinkzustand.

- In der Nebenrechnung  $u_i$  steht, welche Teilrechnung von  $\mathcal{M}_L$  als nächste zu simulieren ist.

Also zählt  $\mathcal{M}$  die  $w \in \Sigma^*$  auf, für die  $\mathcal{M}_L$  hält, und das sind gerade die  $w \in L$ . □

## Satz 13.6 (Entscheidbar und akzeptierbar)

*Jede entscheidbare Sprache ist akzeptierbar.*

## Beweis

Sei  $L$  eine entscheidbare Sprache und  $\mathcal{M}$  eine DTM, die  $L$  entscheidet.

Dann wird  $L$  akzeptiert von der DTM  $\mathcal{M}'$ ,  
die zunächst  $\mathcal{M}$  simuliert und danach in eine Endlosschleife geht, falls  $\mathcal{M}$  mit  $h, \#N\#$  endet.

## Satz 13.7 (Komplement einer entscheidbaren Sprache ist entscheidbar)

*Das Komplement einer entscheidbaren Sprache ist entscheidbar.*

### Beweis

Sei  $L$  eine entscheidbare Sprache und  $\mathcal{M}$  eine DTM, die  $L$  entscheidet.

Dann wird  $\bar{L}$  entschieden von einer DTM  $\mathcal{M}'$ ,

die genau wie  $\mathcal{M}$  rechnet

und nur am Schluß die Antworten  $Y$  und  $N$  vertauscht. □

## Satz 13.8 (Charakterisierung von Entscheidbarkeit)

*Eine Sprache  $L$  ist genau dann entscheidbar, wenn sie und ihr Komplement akzeptierbar sind.*

## Beweis

“ $\Rightarrow$ ”

Sei  $L$  ist entscheidbar.

Zu zeigen:  $L$  und  $\bar{L}$  sind akzeptierbar.

- $L$  ist entscheidbar, also ist  $L$  akzeptierbar
- $L$  ist entscheidbar, also ist  $\bar{L}$  entscheidbar
- $\bar{L}$  ist entscheidbar, also ist  $\bar{L}$  akzeptierbar

## Beweis (Fortsetzung)

“ $\Leftarrow$ ”

Seien  $L$  und  $\bar{L}$  akzeptierbar.

Zu zeigen:  $L$  ist entscheidbar.

- Sei  $\mathcal{M}_1$  eine DTM, die  $L$  akzeptiert.
- Sei  $\mathcal{M}_2$  eine DTM, die  $\bar{L}$  akzeptiert.

Daraus konstruieren wir eine 2-DTM  $\mathcal{M}$ , die  $L$  entscheidet:

- $\mathcal{M}$  wird gestartet mit

$$\begin{array}{c} s_0, \#w\# \\ \# \end{array}$$

- $\mathcal{M}$  kopiert  $w$  auf Band 2.


## Beweis (Ende)

- $M$  simuliert abwechselnd
  - einen Schritt von  $\mathcal{M}_1$  auf Band 1 und
  - einen Schritt von  $\mathcal{M}_2$  auf Band 2.
- Das tut  $\mathcal{M}$ , bis entweder  $\mathcal{M}_1$  oder  $\mathcal{M}_2$  hält.
- Eine von beiden muss halten:  $w$  gehört entweder zu  $L$  oder zu  $\bar{L}$ .
- Wenn  $\mathcal{M}_1$  hält, dann hält  $\mathcal{M}$  mit
  - $\#Y\#$  auf Band 1 und
  - $\#$  auf Band 2.
- Wenn  $\mathcal{M}_2$  hält, dann hält  $\mathcal{M}$  mit
  - $\#N\#$  auf Band 1 und
  - $\#$  auf Band 2.





# Teil V

- 
- 1 Determinierte Turing-Maschinen (DTMs)
  - 2 Varianten von Turing-Maschinen
  - 3 Indeterminierte Turing-Maschinen (NTMs)
  - 4 Universelle determinierte Turing-Maschinen
  - 5 Entscheidbar/Aufzählbar
  - 6 Determinierte Turing-Maschinen entsprechen Typ 0**
  - 7 Unentscheidbarkeit

## Zur Erinnerung

Formale Sprachen sind vom **Typ 0**, wenn sie durch beliebige Grammatiken (keinerlei Einschränkungen) erzeugt werden können.

## Satz 14.1 (Rekursiv aufzählbar = Typ 0)

*Die rekursiv aufzählbaren Sprachen  
(also die durch DTMn akzeptierbaren Sprachen)  
sind genau die durch beliebige Grammatiken erzeugten Sprachen  
(also die vom Typ 0).*

## Beweisidee

- Zu jeder Turing-Maschine kann eine Grammatik konstruiert werden, deren Ableitungsschritte die Rechenschritte der TM simulieren (spezielle Variable markiert Position des Schreib-/Lesekopfes).
- Zu jeder Grammatik kann eine indeterminierte Turing-Maschine (und damit auch eine DTM) konstruiert werden, deren Rechenschritte den Ableitungsschritten der Grammatik entsprechen.