

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

Bernhard Beckert

Institut für Informatik



Sommersemester 2007

Turing-Maschine

Definition 9.8 (Eingabe)

w heißt **Eingabe** (*input*) für \mathcal{M} , falls \mathcal{M} mit der **Startkonfiguration**

$$C_0 = s, \#w\#$$

startet.

(w_1, \dots, w_n) heißt **Eingabe** für \mathcal{M} , falls \mathcal{M} mit der **Startkonfiguration**

$$C_0 = s, \#w_1\# \dots \#w_n\#$$

startet.

Dank

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*

Turing-Maschine

Definition 9.9 (Halten, Hängen)

Sei \mathcal{M} eine Turing-Maschine.

- \mathcal{M} **hält** in $C = q, w\underline{a}u$ **gdw.** $q = h$.
- \mathcal{M} **hängt** in $C = q, w\underline{a}u$ **gdw.** es keine Nachfolgekonfiguration gibt
Insbesondere: wenn $w = \varepsilon \wedge \exists q' \delta(q, a) = (q', L)$.

Turing-Maschine

Definition 9.10 (Rechnung)

Sei \mathcal{M} eine Turing-Maschine. Man schreibt

$$C \vdash_{\mathcal{M}}^* C'$$

gdw.:

es gibt eine Reihe von Konfigurationen

$$C_0, C_1, \dots, C_n \quad (n \geq 0)$$

so daß

- $C = C_0$ und $C' = C_n$
- für alle $i < n$ gilt: $C_i \vdash_{\mathcal{M}} C_{i+1}$

Dann heißt C_0, C_1, \dots, C_n eine **Rechnung** der Länge n von C_0 nach C_n .

Turing-Maschine können Funktionen berechnen

Definition 9.11 (TM-berechenbare Funktion)

Sei Σ_0 ein Alphabet mit $\# \notin \Sigma_0$.

Eine (partielle) Funktion

$$f : (\Sigma_0^*)^m \rightarrow (\Sigma_0^*)^n$$

heißt **DTM-berechenbar**, falls:

Es existiert eine determinierte Turing-Maschine $\mathcal{M} = (K, \Sigma, \delta, s)$

- mit $\Sigma_0 \subseteq \Sigma$,
- so daß für alle $w_1, \dots, w_m, u_1, \dots, u_n \in \Sigma_0^*$ gilt:
 - $f(w_1, \dots, w_m) = (u_1, \dots, u_n)$ gdw
 $s, \#w_1\#\dots\#w_m\# \vdash_{\mathcal{M}}^* h, \#u_1\#\dots\#u_n\#$
 - $f(w_1, \dots, w_m)$ ist undefiniert gdw
 \mathcal{M} gestartet mit $s, \#w_1\#\dots\#w_m\#$ hält nicht (läuft unendlich oder hängt)

Turing-Maschine können Funktionen berechnen

Vorsicht

Wir betrachten Turing-Maschinen hier unter einem anderen Aspekt als alle bisherigen Automaten:

- Bei endlichen Automaten und Pushdown-Automaten haben wir untersucht, **welche Sprachen sie akzeptieren**.
- Bei Turing-Maschinen untersuchen wir,
 - welche Sprachen sie akzeptieren und
 - **welche Funktionen sie berechnen**.

Akzeptieren ist Spezialfall von Berechnen

Turing-Maschine: Akzeptierte Sprache

Definition 9.12 (Von einer DTM akzeptierte Sprache)

Ein Wort w wird **akzeptiert von einer DTM \mathcal{M}** ,

falls \mathcal{M} auf Eingabe von w hält

(wobei am Ende der Kopf auf dem ersten Blank rechts von w steht).

Eine Sprache $L \subseteq \Sigma^*$ **wird akzeptiert von einer DTM \mathcal{M}** , wenn genau die Wörter aus L aus \mathcal{M} und keine anderen akzeptiert werden.

Achtung

Bei nicht akzeptierten Wörtern muss die DTM nicht halten

Sie darf es sogar nicht!

Funktionen auf natürlichen Zahlen

- Wir verwenden die **Unärdarstellung**
Eine Zahl n wird auf dem Band der Maschine durch n senkrechte Striche dargestellt.
- Eine Turing-Maschine \mathcal{M} berechnet eine Funktion

$$f: \mathbb{N}^k \rightarrow \mathbb{N}^n$$

in Unärdarstellung wie folgt:

- Wenn $f(i_1, \dots, i_k) = (j_1, \dots, j_n)$ ist, dann rechnet \mathcal{M}

$$s, \#^{i_1} \# \dots \#^{i_k} \# \vdash_{\mathcal{M}}^* h, \#^{j_1} \# \dots \#^{j_n} \#$$

- Ist $f(i_1, \dots, i_k)$ undefiniert, dann hält \mathcal{M} bei Input $\#^{i_1} \# \dots \#^{i_k} \#$ nicht.

Definition 9.13

- **TM^{part}** ist die Menge der partiellen TM-berechenbaren Funktionen
 $f: \mathbb{N}^k \rightarrow \mathbb{N}$
- **TM** ist die Menge der totalen TM-berechenbaren Funktionen
 $f: \mathbb{N}^k \rightarrow \mathbb{N}$

Achtung: Einschränkung

In der Definition von TM und TM^{part} haben wir uns eingeschränkt:

- nur Funktionen über natürliche Zahlen
- nur Funktionen mit einstelligem Wertebereich

Das ist keine echte Einschränkung

Elemente (Wörter) aus anderen Definitions- und Wertebereiche können als natürliche Zahlen kodiert werden.

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

Variationen von Turing-Maschinen

Standard-DTM

Die Turing-Maschine, die wir bisher kennen, ...

- ist determiniert
- hat ein einseitig unbeschränktes Band (**Halbband**).

Ab jetzt nennen wir sie auch:

Standard-Turing-Maschine (Standard-DTM oder kurz DTM)

Variationen

- **zweiseitig** unbeschränktes Band (kein Hängen)
- **mehrere** Bänder
- **indeterminierte** Turing-Maschinen

Variationen von Turing-Maschinen

Turing-Maschinen, die nie hängen

Gegeben:

Eine Turing-Maschine \mathcal{M} , mit Eingabge $\#w\#$

Daraus konstruieren wir eine DTM \mathcal{M}' , die

- dasselbe berechnet wie \mathcal{M}
- **nie hängt.**

Variationen von Turing-Maschinen

Konstruktion der TM, die nie hängt

Das Bandende ist am Anfang ein Zeichen links vom Eingabewort

DTM \mathcal{M}' rechnet so:

- Sie verschiebt die Eingabe ein Zeichen nach rechts.
- Dann druckt sie ganz links ein **Sonderzeichen α** , das das **Bandende** anzeigt.
- Ab dann rechnet sie wie \mathcal{M} .
- Aber:
Wenn sie α erreicht, bleibt sie dort stehen und druckt immer wieder α .

Variationen von Turing-Maschinen

Eigenschaften der nicht-hängenden DTM

- \mathcal{M}' hält für Eingabe w gdw \mathcal{M} hält für Eingabe w .
- \mathcal{M}' hängt nie.
Wenn \mathcal{M} hängt, rechnet \mathcal{M}' unendlich lang.

O.B.d.A. sollen alle Turing-Maschinen, die wir von jetzt an betrachten, nie hängen.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

DTM mit zweiseitig unbeschränktem Band

- Die Definition der Maschine bleibt gleich.
- Die Definition der **Konfiguration** ändert sich.
- Sie hat immer noch die Form $q, w\underline{a}u$, aber:
 - w umfasst analog zu u alle Zeichen bis zum letzten nicht-Blank links vom Schreib-/Lesekopf.
 - $w = \varepsilon$ bzw. $u = \varepsilon$ bedeutet, daß links bzw. rechts vom Schreib-/Lesekopf nur noch Blanks stehen.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Definition 10.1 (DTM mit zweiseitig unbeschränktem Band, zw-DTM)

Eine **Turing-Maschine mit zweiseitig unbeschränktem Band (zw-DTM)** ist eine DTM, für die die Begriffe der Konfiguration und der Nachfolgekonfiguration wie folgt definiert sind:

Eine **Konfiguration** C einer zw-DTM $\mathcal{M} = (K, \Sigma, \delta, s)$ ist von der Form

$$C = q, w\underline{a}u$$

Dabei ist

- $q \in K \cup \{h\}$ der aktuelle Zustand,
- $w \in (\Sigma - \{\#\})\Sigma^* \cup \{\varepsilon\}$ der Bandinhalt links des Kopfes,
- $a \in \Sigma$ das Zeichen unter dem Kopf, und
- $u \in \Sigma^*(\Sigma - \{\#\}) \cup \{\varepsilon\}$ der Bandinhalt rechts des Kopfes.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Definition (Forts.)

$C_2 = q_2, w_2\underline{a_2}u_2$ heißt **Nachfolgekonfiguration** von $C_1 = q_1, w_1\underline{a_1}u_1$, in Zeichen $C_1 \vdash_{\mathcal{M}} C_2$,

falls es einen Übergang $\delta(q_1, a_1) = (q_2, b)$ gibt, mit:

Fall 1: $b \in \Sigma$. Dann $w_1 = w_2, u_1 = u_2$ und $a_2 = b$.

Fall 2: $b = L$. Für u_2 : Wenn $a_1 = \#$ und $u_1 = \varepsilon$ ist, dann $u_2 = \varepsilon$, sonst $u_2 = a_1 u_1$.

Für a_2 und w_2 : Wenn $w_1 = \varepsilon$ ist, dann $w_2 = \varepsilon$ und $a_2 = \#$; sonst $w_1 = w_2 a_2$.

Fall 3: $b = R$. Für w_2 : Wenn $a_1 = \#$ und $w_1 = \varepsilon$ ist, dann $w_2 = \varepsilon$, sonst $w_2 = w_1 a_1$.

Für a_2 und u_2 : Wenn $u_1 = \varepsilon$ ist, dann $u_2 = \varepsilon$ und $a_2 = \#$; ansonsten $u_1 = a_2 u_2$.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Theorem 10.2 (Simulation von zw-DTM durch DTM)

Zu jeder zw-DTM \mathcal{M} , die eine Funktion f berechnet oder eine Sprache L akzeptiert, existiert eine Standard-DTM \mathcal{M}' , die ebenfalls f berechnet bzw. L akzeptiert.

Beweis

Sei $w = a_1 \dots a_n$ die Eingabe für $M = (K, \Sigma, \delta, s)$.

Dann sieht das beidseitig unendliche Band zu Beginn der Rechnung so aus:

$$\dots \### a_1 \dots a_n \### \dots$$

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Idee:

- \mathcal{M} hat quasi zwei unendlich lange Halbbänder.
- Ziel ist, den Inhalt beider Halbbänder von \mathcal{M} auf einem unterzubringen.
- Dazu: Den Teil des Bandes, der zwei Zeichen links vom Input w beginnt, umklappen:
Spur 1 ## ...# #
Spur 2 # $a_1 \dots a_n \# \dots$
- Die DTM \mathcal{M}' hat zwei **Spuren**, d.h. zwei Reihen von Zeichen, die auf demselben Band untergebracht (kodiert) sind.
- Das Bandalphabet von \mathcal{M}' ist $\Sigma' \supseteq \Sigma \times \Sigma$.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Sei $\mathcal{M}' = (K', \Sigma', \delta', s)$. \mathcal{M}' rechnet so:

- \mathcal{M}' legt zunächst eine zweite Spur an,
- simuliert dann die Arbeit von \mathcal{M} , und
- transformiert dann das Ergebnis wieder auf nur eine Spur herunter.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Erste Phase der Rechnung:

\mathcal{M}' rechnet

$$s, \# a_1 \dots a_n \# \vdash_{\mathcal{M}'}^* q, \$ \begin{array}{c} \# \# \dots \# \# \\ \# a_1 \dots a_n \# \end{array} \# \dots$$

Die zweite Spur wird nur so weit wie nötig angelegt.

Mit dem Symbol \$ markiert \mathcal{M}' das Ende des Halbbands, damit sie nicht hängenbleibt.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Zweite Phase der Rechnung:

\mathcal{M}' simuliert \mathcal{M} .

Dabei muß sie sich immer merken, auf welcher der beiden Spuren sie gerade arbeitet.

Deshalb definieren wir $K' \supseteq K \times \{1, 2\}$.

(q, i) bedeutet, daß die simulierte Maschine \mathcal{M} im Zustand q ist und \mathcal{M}' auf Spur i arbeitet.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Dritte Phase der Rechnung:

Die Simulation von \mathcal{M} ist abgeschlossen.

\mathcal{M}' muß nun den Bandinhalt von zwei Spuren auf nur eine heruntertransformieren, um danach die Konfiguration $h, \#u\#$ zu erreichen.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

- \mathcal{M}' macht zunächst alle $\#$ rechts vom beschriebenen Bandteil zu $\#$. Für Fall (i) und (ii) löscht sie die $\#$ links von u^R bzw. u .
- Für Fall (iii) schiebt \mathcal{M}' dann die untere Spur nach links, bis sie eine Konfiguration $q, \# \dots \# u_2 \#$ erreicht.
- Für Fall (i) und (iii) muß \mathcal{M}' jetzt u_1^R bzw. u^R auf nur eine Spur transformieren und zugleich invertieren, sie muß also für den allgemeineren Fall (iii) $q, \$ \# \dots \# u_2 \# \vdash_{\mathcal{M}'}^* q', \$ u_1 u_2 \#$ rechnen.
- Danach muß \mathcal{M}' nur noch das $\$$ links löschen und nach rechts neben u laufen.

DTM mit zweiseitig unbeschränktem Band (zw-DTM)

Beweis (Forts.)

Damit hat die Standard-DTM \mathcal{M}' die Arbeitsweise der zw-DTM \mathcal{M} vollständig simuliert.

Also kann man mit zw-Turing-Maschinen nicht mehr berechnen als mit Standard DTM'n. \square

DTM mit k Halbbändern

Definition 10.3 (DTM mit k Halbbändern, k -DTM)

Eine **Turing-Maschine** $\mathcal{M} = (K, \Sigma_1, \dots, \Sigma_k, \delta, s)$ **mit k Halbbändern** (mit je einem Kopf) ist eine Turing-Maschine mit einer Übergangsfunktion

$$\delta : K \times \Sigma_1 \times \dots \times \Sigma_k \rightarrow (K \cup \{h\}) \times (\Sigma_1 \cup \{L, R\}) \times \dots \times (\Sigma_k \cup \{L, R\})$$

Eine **Konfiguration** einer k -Turing-Maschine hat die Form

$$C = q, w_1 \underline{a_1} u_1, \dots, w_k \underline{a_k} u_k.$$

DTM mit k Halbbändern

- Die Köpfe einer k -DTM können sich **unabhängig** bewegen (sonst hätten wir nur eine DTM mit k Spuren).
- Die Definition der Nachfolgekonfiguration verläuft analog zu der Definition bei Standard-DTM.
- Für eine k -DTM, die eine Funktion $f : \Sigma_0^m \rightarrow \Sigma_0^n$ berechnet, legen wir fest, daß sowohl die m Eingabewerte als auch – nach der Rechnung – die n Ergebniswerte auf dem ersten Band stehen sollen.
- Es übertragen sich alle Begriffe wie *berechenbar*, *entscheidbar* etc. kanonisch auf k -DTM.

Theorem 10.4 (Simulation von k -DTM durch DTM)

Zu jeder k -DTM \mathcal{M} , die eine Funktion f berechnet (resp. eine Sprache L akzeptiert), existiert eine DTM \mathcal{M}' , die ebenfalls f berechnet (resp. L akzeptiert).

Beweis (Skizze)

Wir arbeiten mit einer Turing-Maschine mit mehreren Spuren.

Um eine k -DTM zu simulieren, verwenden wir **$2k$ Spuren**, also Bandzeichen, die aus $2k$ übereinander angeordneten Buchstaben bestehen.

- In den Spuren mit ungerader Nummer stehen die Inhalte der k Bänder von \mathcal{M} .
- Die Spuren mit gerader Nummer verwenden wir, um die Positionen der Köpfe von \mathcal{M} zu simulieren:
Die $2i$ -te Spur enthält an genau einer Stelle ein \wedge , nämlich da, wo \mathcal{M} gerade seinen i -ten Kopf positioniert hätte, und ansonsten nur Blanks.

\mathcal{M}' kodiert zunächst die Eingabe von \mathcal{M} . Dann simuliert \mathcal{M}' die Maschine \mathcal{M} . Am Ende der Rechnung wird noch die Ausgabe dekodiert.

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

Indeterminierte Turing-Maschine

Definition 11.1 (Indeterminierte Turing-Maschine, NTM)

Eine **indeterminierte Turing-Maschine** \mathcal{M} ist ein Tupel

$$M = (K, \Sigma, \Delta, s)$$

Dabei sind K, Σ, s definiert wie bei determinierten Turing-Maschinen.

Übergangsrelation:

$$\Delta \subseteq (K \times \Sigma) \times ((K \cup \{h\}) \times (\Sigma \cup \{L, R\}))$$

Indeterminierte Turing-Maschine

Mehrere Nachfolgekongfigurationen

Kongfigurationen sind definiert wie bei DTMs.

Nun kann eine Konfiguration aber mehrere mögliche Nachfolgekongfigurationen haben.

Indeterminierte Turing-Maschine

Definition 11.2 (NTM: Halten, Hängen, Akzeptieren)

Sei $\mathcal{M} = (K, \Sigma, \Delta, s_0)$ eine indeterminierte Turing-Maschine.

- \mathcal{M} **hält** bei Input w , falls es **unter den möglichen Rechnungen**, die \mathcal{M} wählen kann, **eine gibt**, so daß \mathcal{M} eine Haltekonfiguration erreicht.
- \mathcal{M} **hängt** in einer Konfiguration, wenn es keine (durch Δ definierte) Nachfolgekongfiguration gibt.
- \mathcal{M} **akzeptiert** ein Wort w , falls sie **von $s, \#w\#$ aus einen Haltezustand erreichen kann**, und \mathcal{M} akzeptiert eine Sprache L , wenn sie genau alle Wörter $w \in L$ akzeptiert.

Indeterminierte Turing-Maschine

Bemerkung

Wenn es nicht nur darauf ankommt, ob die Maschine hält, sondern auch mit welchem Bandinhalt:

Welche der vielen Haltekonfigurationen sollte dann gelten?

Um dies Problem zu umgehen, übertragen wir die Begriffe des *Entscheidens* und *Aufzählens* **nicht** auf NTM. Im Allgemeinen verwendet man NTM auch nicht dazu, Funktionen zu berechnen.

Wie rechnet eine indeterminierte Turing-Maschine?

- Die Regeln einer determinierten DTM kann man sich als Programm (aus sehr einfachen Schritten) vorstellen.
- **Bei NTM ist das anders!**
- **Eine NTM ist nicht einfach eine Maschine, die immer richtig rät!**

Dieselbe Diskussion hatten wir bei indeterminierten endlichen Automaten.

Vorstellung von einer NTM

- Korrekte Vorstellung:
 - Übergänge von Konfiguration zu Nachfolgekonfiguration entspr. Regelmenge
 - **plus Suchverfahren!**oder: Eine NTM beschreitet alle möglichen Rechenwege parallel.
- **Eine NTM akzeptiert ein Wort, wenn es mindestens einen Berechnungsweg gibt, der in einer Haltekonfiguration endet.**
- Sprechweise **“Die NTM rät”**: Wir verwenden diese Sprechweise, sie ist aber mit Vorsicht zu genießen!