Berechenbarkeit und Komplexität

Dr. Eva Richter

13. April 2012

Inhalt Berechenbarkeitstheorie

- Church-Turing-These: Turingmaschinen, Algorithmen, funktionale Programmierung und λ-Kalkül
- Entscheidbarkeit: Probleme von regulären und kontextfreien Sprachen, Halteproblem
- Reduzierbarkeit: unentscheidbare Probleme der Sprachtheorie, Korrespondenzproblem von Post, Abbildungsreduzierbarkeit
- Rekursionssatz, Beschreibungskomplexität

- Zeitkomplexität: asymptotisches Verhalten, Klassen P und NP, NP-Vollständigkeit
- Platzkomplexität: Savitch-Theorem, Klasse PSPACE
- Probabilistische Algorithmen

Organisatorisches

- Vorlesung freitags 8:30-10:00 Uhr, Tutorial donnerstags 10:00-11:30 Uhr, Übungen montags, dienstags, mittwochs
- Tutoren: Nuria Brede, Margrit Dittmann, Mario Frank, Charlotte Gerlitz, Michael Görner, Anna Melzer, Hannnes Schröder, Alexander Schulze, Malte Swart
- Sprechzeiten: immer, wenn jemand da ist oder nach Vereinbarung
- Prüfung: Klausur am Donnerstag, 19.07. 2012, bei Bestehen 6 Leistungspunkte
- Hausaufgaben: wöchentlich von Freitag zu Freitag, freiwillige Abgabe
- Literatur: Michael Sipser: Introduction to the Theory of Computation, weitere Literatur siehe Website
- Videoaufzeichnung von Prof. Kreitz SS2007: http://www.tele-task.de/archive/series/overview/597/

- 1936 von Alan Turing als Gedankenmodell vorgestellt
- kann alles, was ein realer Computer kann (auch dafür gibt es Grenzen)
- Zutaten:
  1. unbeschränktes Band
  2. Kopf zum Lesen und Schreiben, der sich auf dem Band hin und her bewegen kann
  3. interne Zustände und Haltezustände für Akzeptieren und Zurückweisen
  4. kann unendlich lange laufen

Turingmaschine für die Sprache $\{w\sharp w \mid w \in (0+1)^*\}$

1. Überprüfe die Eingabe, ob sie genau ein $\sharp$ enthält
   (falls nicht: **ablehnen**)
2. Hin-und Herbewegen, um zu prüfen, ob links und rechts
   dieselben Symbole stehen
   dabei Absteichen der Symbole, die schon geprüft wurden
3. wenn alle Symbole links von $\sharp$ geprüft worden sind, nachsehen,
   ob rechts von $\sharp$ noch etwas steht,
   wenn nicht: **akzeptieren**, sonst: **ablehnen**

Blick aufs Band bei Eingabe des Wortes 011000♯011000

0	1	1	0	0	0	♯	0	1	1	0	0	0	␣ . . .
x	1	1	0	0	0	♯	0	1	1	0	0	0	␣ . . .
													. . .
0	1	1	0	0	0	♯	x	1	1	0	0	0	␣ . . .
x	1	1	0	0	0	♯	x	1	1	0	0	0	␣ . . .
x	x	1	0	0	0	♯	0	1	1	0	0	0	␣ . . .
													. . .
x	x	x	x	x	x	♯	x	x	x	x	x	x	␣ . . .

accept

Definition

Eine Turingmaschine ist ein 7-Tupel $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, wobei Q, Σ und Γ endliche Mengen sind und folgende Bedingungen erfüllt werden:

1. Q ist die Menge der Zustände
2. Σ ist das Eingabealphabet, das nicht das Leersymbol $\sqcup$ enthält
3. Γ ist das Bandalphabet, wobei $\sqcup \in \Gamma$ und $\Sigma \subset \Gamma$
4. $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ ist die Übergangsfunktion
5. q_0 ist der Startzustand
6. q_{accept} ist der akzeptierende Zustand
7. q_{reject} ist der ablehnende Zustand.

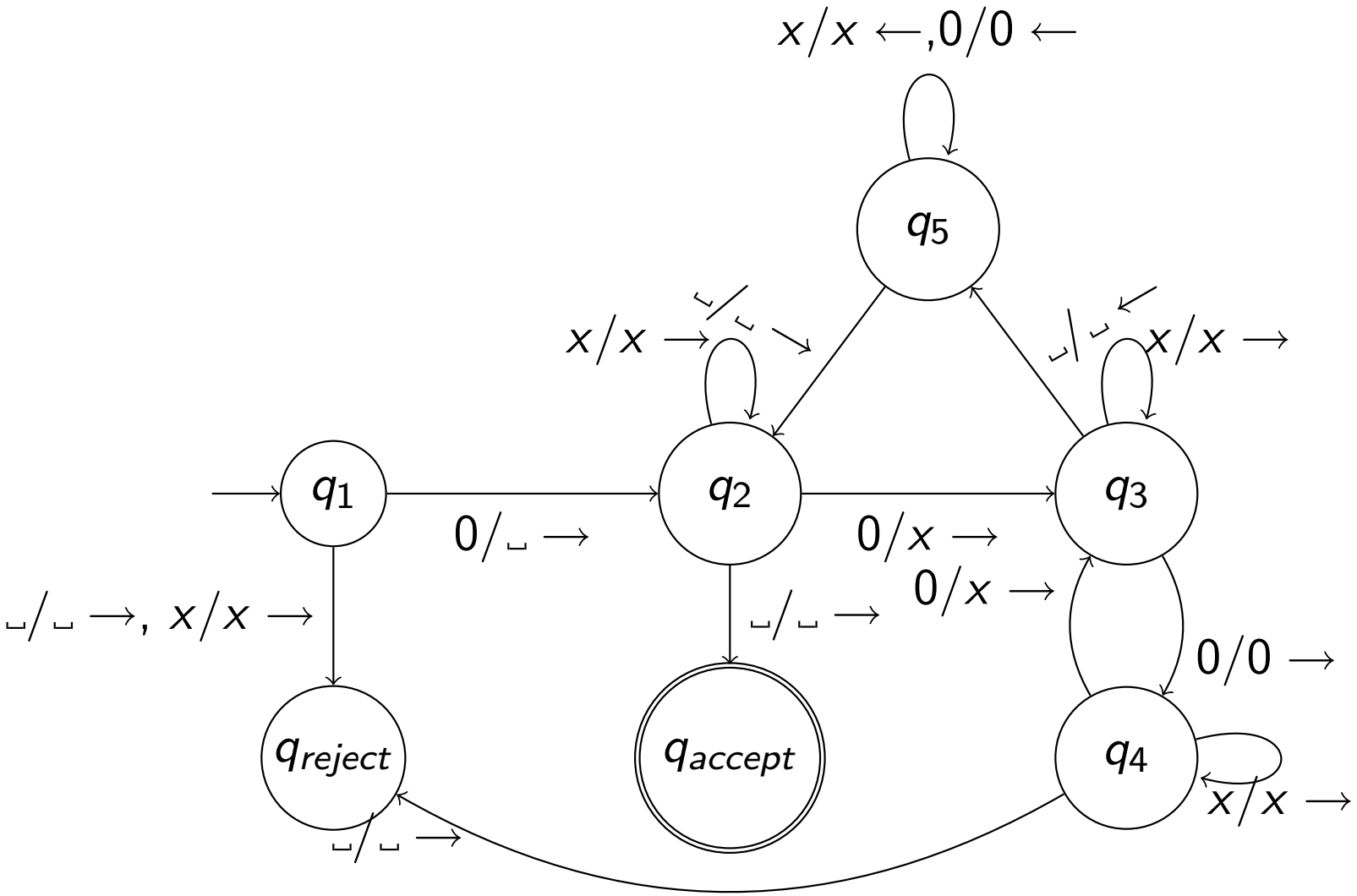
Verhalten von $M = (Q, \Sigma, \Gamma\delta, q_0, q_{accept}, q_{reject})$

- Eingabe $w = w_1 ... w_n \in \Sigma^*$ steht auf den ersten(linkesten) n Zellen des Bandes, Rest des Bandes ist mit Blanks gefüllt, beachte: $\sqcup \in \Gamma \setminus \Sigma$
- Kopf steht auf w_1 und die Maschine ist im Zustand q_0
- Maschine arbeitet entsprechend den Regeln von δ
- falls der Kopf ganz links steht und die nächste Anweisung lautet nach links zu gehen, so bleibt der Kopf in der ersten Position
- Berechnungen erfolgen solange, bis q_{accept} oder q_{reject} erreicht wird oder unendlich lange weiter.

Übergangsdiagramme zur Darstellung des Verhaltens

- Zustände werden durch Knoten dargestellt
- Startknoten wird durch einen Startpfeil markiert, akzeptierender Endzustand durch einen doppelten Kreis
- Übergänge werden markiert durch: das gelesene Zeichen, Schrägstrich, das zu schreibende Zeichen, Bewegungsrichtung des Kopfes

Übergangsdiagramm für M mit $L(M) = \{0^{2^n} \mid n \in \mathbb{N}\}$



Beschreibung für M mit $L(M) = \{0^{2^n} \mid n \in \mathbb{N}\}$

$M =$„Auf Eingabe von w:

1. gehe von links nach rechts übers Band und streiche jede zweite 0 ab
2. falls in 1 nur eine 0 vorhanden war, **akzeptiere**
3. falls in 1 mehr als eine Null vorhanden war und deren Anzahl ungerade ist, **lehne ab**
4. gehe zum linken Bandende
5. kehre zu 1 zurück

In jedem Schritt kann sich der aktuelle Zustand, der aktuelle Bandinhalt und die aktuelle Position des Kopfes ändern.

Definition

Die Konfiguration einer TM ist ein Tripel (u, q, v), geschrieben als (uqv), wobei uv der aktuelle Inhalt des Bandes, q der aktuelle Zustand und das erste Zeichen von v die aktuelle Position des Kopfes ist.

Beispiel Konfiguration $(1011q_701111)$, wenn der aktuelle Bandinhalt 101101111 ist, die Maschine sich im Zustand q_7 befindet und der Kopf über der zweiten Null steht.

Definition

Eine Konfiguration K_2 ist Nachfolgekonfiguration von K_1, falls die Turingmaschine in einem Schritt von K_1 nach K_2 gelangt.

Beispiel

- uq_jacv ist die Nachfolgekonfiguration von uaq_ibv, falls $\delta(q_i, b) = (q_j, c, L)$
- $uacq_iv$ Nachfolgekonfiguration falls $\delta(q_ib) = (q_jcR)$
- Sonderfälle am linken Bandende: q_jcv ist die Nachfolgerkonfiguration von q_ibv, wenn $\delta(q_i, b) = (q_j, c, L)$
- am rechten Ende: (uaq_i) ist äquivalent zu $(uaq_i\sqcup)$, d.h der Kopf geht über das rechte Ende des Wortes hinaus.

Definition

M akzeptiert eine Eingabe w, falls eine Folge $K_1 \dots K_k$ von Konfigurationen existiert, sodass

1. *K_1 ist Startkonfiguration von M bei Eingabe von w,*
2. *jedes K_{i+1} ist Nachfolgekonfiguration von K_i und*
3. *K_k ist eine akzeptierende Konfiguration.*

Die Menge der Eingaben, die von M akzeptiert werden, heißt Sprache von M, bezeichnet mit $L(M)$.

Definition

Eine Sprache heißt Turing-akzeptierbar, rekursiv aufzählbar, semi-entscheidbar, wenn es eine Turingmaschine gibt, die diese Sprache akzeptiert.

- M hat auf einer Eingabe drei Möglichkeiten: kommt in einen der Haltezustände (accept oder reject) oder hält nicht an
- praktisch lässt sich eine Schleife nicht von einer langen Laufzeit unterscheiden

Definition

1. *Ein Entscheider ist eine TM, die auf jeder Eingabe anhält.*
2. *Eine Sprache heißt (Turing)-entscheidbar oder rekursiv, wenn es eine Turingmaschine gibt, die sie entscheidet.*

Jede entscheidbare Sprache ist auch Turing-akzeptierbar, aber nicht notwendig umgekehrt.

Beispiel für entscheidbare Sprachen

1. $L_1 = \{0^{2^n}\}$, $L_2 = \{a^i b^j c^k \mid i \times j = k \text{ und } i, j, k \geq 1\}$,
2. $L_3 = \{\sharp x_1 \sharp x_2 \sharp \ldots \sharp x_n \mid x_i \in \{0,1\}^* \text{ und } x_i \neq x_j \text{ wenn } i \neq j\}$

- statt einem gibt es mehrere rechtsseitig unendliche Bänder, jedes Band hat einen eigenen Kopf zum Lesen und Schreiben
- Übergangsfunktion darf alle Köpfe gleichzeitig bewegen und lesen bzw. schreiben.

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k,$$

- $\delta(q_i, a_1, \ldots a_k) = (q_j, b_1, \ldots b_k, L, R, \ldots, L)$: wenn die Maschine im Zustand q_i mit den Köpfen 1 bis k die Zeichen $a_1 \ldots a_k$ von den k Bändern liest, geht sie in den Zustand q_j über, schreibt auf die i-ten Bänder die Zeichen b_i und bewegt die Köpfe gemäß der Folge $L, R, \ldots, L$

Satz

Zu jeder Mehrband-Turingmaschine gibt es eine äquivalente Einband-Turingmaschine

$S =$ „Bei Eingabe von $w = w_1, \ldots w_n$:

1. schreibe Inhalt der k Bänder aufs Band:

$$\Diamond \dot{w}_1 w_2 \ldots w_n \Diamond \dot{\sqcup} \Diamond \dot{\sqcup} \Diamond \ldots \Diamond$$

2. für jeden M-Schritt: lies punktierte Symbole zwischen ersten und $k+1$-ten $\Diamond$, vollziehe Übergänge nach δ_M
3. falls einer der virtuellen Köpfe nach rechts auf ein $\Diamond$ kommt, schreibe Blank und schiebe restlichen Bandinhalt um eine Zelle nach rechts fahre fort wie in 2 “

- ähnlich wie NEA, an jedem Punkt der Berechnung hat die Maschine mehrere (festlgelegte) Möglichkeiten

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R\}).$$

- NTM akzeptiert eine Eingabe, wenn mindestens ein Zweig zu einem akzeptierenden Zustand führt
- NTM ist ein Entscheider, wenn jeder Pfad zu einem Haltezustand führt.

Satz

Für jede nichtdeterministische Turingmaschine gibt es eine äquivalente deterministische Turingmaschine.

Beweisidee:

- wir simulieren NTM N durch DTM D
- D probiert alle möglichen Zweige der nichtdeterministischen Berechnung von N aus
- findet D einen akzeptierenden Zustand, akzeptiert sie, sonst läuft D unendlich lange weiter
- im Berechnungsbaum ist jeder Knoten eine Konfiguration von N, D führt Breitensuche durch

die deterministische TM D hat 3 Bänder

Band 1: enthält den Eingabestring und wird nie verändert

Band 2: verwaltet die Kopie vom Band von N, während eines Zweiges der nichtdeterministischen Berechnung

Band 3: „merkt sich“ an welcher Stelle vom Berechnungsbaum von N sich D befindet

Beweis Äquivalenz NTM und DTM II

n:= maximale Anzahl an möglichen Verzweigungen für δ_N

1. Band 1 enthält Eingabe w, Band 2 und 3 sind leer,
2. Kopiere Band 1 auf Band 2
3. simuliere mit Hilfe von Band 2 den Berechnungszweig, der durch die Knotennummer auf Band 3 bestimmt ist falls Kopf 3 ein Blank oder ungültige Konfiguration liest, brich ab und gehe über in 4, bei reject-Konfiguration gehe zu 4, bei accept-Kofiguration, **akzeptiere** w
4. ersetzte String auf Band 3 durch den lexikographisch nächsten String, gehe zu 2

Folgerung

Satz

1. *Eine Sprache ist genau dann Turing-akzeptierbar, wenn es eine NTM gibt, die sie akzeptiert.*
2. *Eine Sprache ist genau dann entscheidbar, wenn eine NTM existiert, die sie entscheidet.*

Aufzähler

- Turing-akzeptierbare Sprachen heißen auch rekursive oder aufzählbare Sprachen
- stammt von Aufzähler, einer Turingmaschinenvariante
- Aufzähler E ist TM mit einem angehängten Drucker
- arbeitet ohne Eingabe und gibt von Zeit zu Zeit bestimmte Zeichenketten auf den Drucker aus
- Menge aller Strings, die E ausgibt, ist die Sprache von E. E darf die Menge in beliebiger Ordnung ausgeben, auch mit Wiederholungen.

Aufzähler sind äquivalent zu anderen TM

Satz

Eine Sprache ist genau dann Turing-akzeptierbar, wenn es einen Aufzähler gibt, der sie aufzählt.

für jeden Aufzähler E der Sprache A gibt es eine TM M, die A akzeptiert:

$M=$ „bei Eingabe von w

1. lasse E laufen, vergleiche alle Ausgaben von E mit w
2. falls w in der Liste auftaucht, **akzeptiere**“

Offensichtlich akzeptiert M genau die Wörter, die E aufzählt.

Aufzähler sind äquivalent zu anderen TM II

Sei M TM, die eine Sprache A akzeptiert, sei $s_1, s_2 \ldots$ Liste aller möglichen Strings in Σ^*
$E =$ „Ignoriere die Eingabe
Wiederhole für $i = 1, 2, \ldots$ die folgenden Schritte:

1. lasse M jeweils die ersten i Schritte auf den Eingaben $s_1 \ldots s_i$ arbeiten,
2. falls eine der Berechnungen akzeptiert wird, dann gib das entsprechende s_j aus. “

- jedes s, das von M akzeptiert wird, wird irgendwann auf der Liste von E ausgegeben (sogar unendlich oft)
- Effekt des Vorgehens: M läuft parallel auf allen möglichen Eingaben