Grundlagen der Theoretischen Informatik

Sommersemester 2015

23.04.2015

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Bis jetzt

1. Terminologie
2. Endliche Automaten und reguläre Sprachen
3. Kellerautomaten und kontextfreie Sprachen
4. Turingmaschinen und rekursiv aufzählbare Sprachen
5. Berechenbarkeit, (Un-)Entscheidbarkeit
6. Komplexitätsklassen P und NP

Bis jetzt

- **Alphabete, Wörter**
  - Operationen auf Wörtern

    Konkatenation, i-te Potenz, Reverse
- **Sprache**
  - Operationen auf Sprachen

    Konkatenation, i-te Potenz, Reverse, Kleene-Hülle
- **Reguläre Ausdrücke**
- **Grammatiken**
  - Ableitung
  - Die von einer Grammatik erzeugte Sprache.

Grammatik

Menge von Regeln, mit deren Hilfe man Wörter ableiten kann

Definition. Eine Grammatik G über einem Alphabet Σ ist ein Tupel

$$G = (V, T, R, S)$$

Dabei ist

- V eine endliche Menge von **Variablen**
- $T \subseteq \Sigma$ eine endliche Menge von **Terminalen** mit $V \cap T = \emptyset$
- R eine endliche Menge von **Regeln**
- $S \in V$ das **Startsymbol**

Konvention (meistens)

- **Variablen** als Großbuchstaben
- **Terminale** als Kleinbuchstaben

Rechnung einer Grammatik

Eine Regel ist ein Element $(P, Q) \in \big((V \cup T)^* \; V \; (V \cup T)^*\big) \times (V \cup T)^*$.

- P und Q sind Wörter über $(V \cup T)$
- P muss mindestens eine Variable enthalten
- Q ist beliebig

Schreibweise: $P \to_G Q$, $P \to Q$ (P: Prämisse, Q: Conclusio)

Ableitung, Rechnung.

- $w \Longrightarrow_G w'$ („w geht über in w'“) falls

$$\exists u, v \in (V \cup T)^* \; \exists P \to Q \in R \;\; \big(w = uPv \text{ und } w' = uQv\big)$$

- $w \Longrightarrow_G^* w'$ falls es Wörter $w_0, \ldots, w_n \in (V \cup T)^*$ $(n \geq 0)$ gibt mit
  - $w = w_0$
  - $w_m = w'$
  - $w_i \Longrightarrow_G w_{i+1}$ für $0 \leq i < n$

Erzeugte Sprache, Äquivalenz

Definition (Erzeugte Sprache) Gegeben: Eine Grammatik G

Die von G erzeugte Sprache $L(G)$ ist die Menge aller **terminalen** Wörter, die durch G vom Startsymbol S aus erzeugt werden können:

$$L(G) \quad := \quad \{w \in T^* \mid S \Longrightarrow_G^* w\}$$

Definition (Äquivalenz)

Zwei Grammatiken G_1, G_2 heißen **äquivalent** gdw

$$L(G_1) = L(G_2)$$

Beispiel: Dycksprache

Definition (Dycksprache)

Gegeben:

- $k \in \mathbb{N}$
- $\Sigma_k := \{x_1, \overline{x_1}, x_2 \ldots, x_k, \overline{x_k}\}$ ein Alphabet mit $2k$ Symbolen

Beispiel: Dycksprache

Definition (Dycksprache)

Gegeben:

- $k \in \mathbb{N}$
- $\Sigma_k := \{x_1, \overline{x_1}, x_2 \ldots, x_k, \overline{x_k}\}$ ein Alphabet mit $2k$ Symbolen

Die Dycksprache D_k ist die kleinste Menge die folgende Bedingungen erfüllt:

1. $\epsilon \in D_k$,
2. Falls $w \in D_k$, so auch $x_i w \overline{x_i}$.
3. Falls u, $v \in D_k$, so auch uv.

Beispiel: Dycksprache

Definition (Dycksprache)

Gegeben:

- $k \in \mathbb{N}$
- $\Sigma_k := \{x_1, \overline{x_1}, x_2 \ldots, x_k, \overline{x_k}\}$ ein Alphabet mit $2k$ Symbolen

Die Dycksprache D_k ist die kleinste Menge die folgende Bedingungen erfüllt:

1. $\epsilon \in D_k$,
2. Falls $w \in D_k$, so auch $x_i w \overline{x_i}$.
3. Falls u, $v \in D_k$, so auch uv.

Interpretiert man die x_i als öffnende, die $\overline{x_i}$ als zugehörige schließende Klammern, so kann man die Dycksprache als die **Menge aller korrekten Klammerausdrücke** sehen.

Beispiel: Dycksprache

$k = 2$

$\Sigma_2 := \{\ [,], (,)\ \}$

- $[[()[]]()] \in D_2$
- $([)] \notin D_2$
- $)) \notin D_2$

Beispiel: Dycksprache

Definition (Dycksprache)

Gegeben:

– $k \in \mathbb{N}$, $\Sigma_k := \{x_1, \overline{x_1}, x_2 \ldots, x_k, \overline{x_k}\}$ ein Alphabet mit $2k$ Symbolen

Die Dycksprache D_k ist die kleinste Menge die folgende Bedingungen erfüllt:

1. $\epsilon \in D_k$,
2. Falls $w \in D_k$, so auch $x_i w \overline{x_i}$.
3. Falls $u, v \in D_k$, so auch uv.

Grammatik $G = (\{S\}, \{x_1, \overline{x_1}, x_2 \ldots, x_k, \overline{x_k}\}, R, S)$ mit $L(G) = D_k$

R_1 : $S \to \varepsilon$

R_2 : $S \to x_1 S \overline{x_1} \mid \ldots \mid x_k S \overline{x_k}$

R_3 : $S \to SS$

Beispiel: Dycksprache

Um zu zeigen, dass $L(G) = D_k$, zeigen wir dass $L(G)$ die kleinste Menge ist die folgende Bedingungen erfüllt:

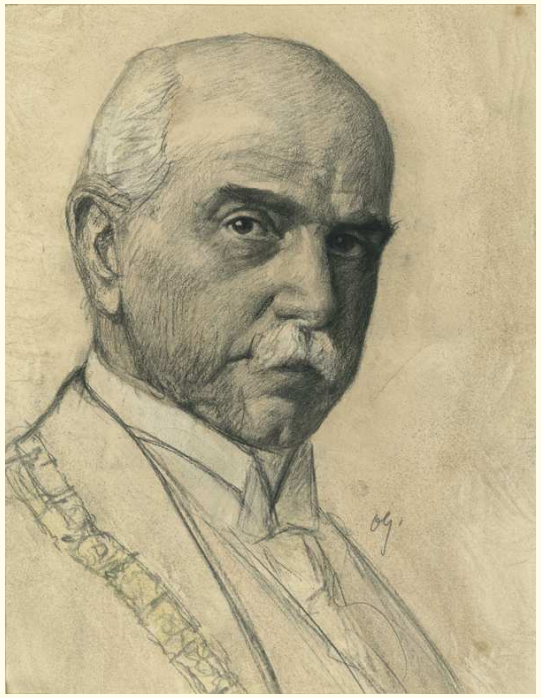
1. $\epsilon \in L(G)$,
2. Falls $w \in L(G)$, so auch $x_i w \overline{x_i}$.
3. Falls u, $v \in L(G)$, so auch uv.

Beweis: an der Tafel.

Beispiel: Dycksprache

Walther von Dyck (1856-1934)

- Mathematiker
- Hochschulpolitiker
- Erster Rektor der TU München
- Einer der Gründungsväter des Deutschen Museums



[Foto: Deutsches Museum]

Warum Sprachen?

Darstellung von Problemen

Fakt: So ziemlich alle Probleme können als Probleme über Sprachen formuliert werden.

Warum Sprachen?

Darstellung von Problemen

Fakt: So ziemlich alle Probleme können als Probleme über Sprachen formuliert werden.

Beispiel: Primzahlen

Alphabet $\Sigma_{\text{num}} := \{|\}$

Sprache $L_{\text{primes}} := \{\underbrace{||\ldots|}_{p \text{ mal}} \mid p \text{ prim}\}$

Darstellung von Problemen

Eingabealphabet

$$\Sigma = \{0, 1, \ldots, n-1\}$$

erlaubt Darstellung einer Ganzzahl zur Basis n

Darstellung von Problemen

Eingabealphabet

$$\Sigma = \{0, 1, \ldots, n-1\}$$

erlaubt Darstellung einer Ganzzahl zur Basis n

Beispiel:

5 binär: 101

5 unär: ||||| (oder auch 11111)

Darstellung von Problemen

Speicheraufwand

n-äre Darstellung ($n > 1$) einer Zahl k führt zu einer Speicherersparnis:

$$\log_n k \quad (n\text{-är}) \qquad \text{statt} \qquad k \quad (\text{unär})$$

Nur der Schritt von unär auf binär ist wesentlich, denn

$$\log_n k \;=\; \frac{1}{\log_2 n} \cdot \log_2 k = c \cdot \log_2 k$$

(von binär auf n-är nur lineare Einsparung)

Darstellung des Erfüllbarkeitsproblems SAT

Problem SAT

Gegeben: Eine aussagenlogische Formel w

Frage: Gibt es eine Belegung der booleschen Variablen in w,
so dass w zu *true* auswertet?

Darstellung des Erfüllbarkeitsproblems SAT

Problem SAT

Gegeben: Eine aussagenlogische Formel w

Frage: Gibt es eine Belegung der booleschen Variablen in w,
so dass w zu *true* auswertet?

Signatur für aussagenlogische Formeln

Signatur: $\Sigma_{\text{sat}} := \{\wedge, \vee, \neg, (,), x, 0, 1\}$

Dabei Darstellung von boolscher Variablen x_i als x gefolgt von i binär kodiert.
Dadurch Formel der Länge n um (unerheblichen) Faktor $\log n$ länger.

Darstellung des Erfüllbarkeitsproblems SAT

Satisfiability

Sprache

$$L_{\text{sat}} := \{w \in \Sigma^*_{sat} : \quad w \text{ ist eine aussagenlogische Formel,}$$
$$\text{und es gibt eine Belegung fÃ¼r die } x_i,$$
$$\text{so dass die Formel } w \text{ zu } \textit{true} \text{ auswertet } \}$$

Darstellung des Erreichbarkeitsproblems in Graphen

Erreichbarkeitsproblem

Gegeben: Ein Graph mit Ecken v_1 bis v_n

Frage: Gibt es einen Weg von Ecke v_1 zu Ecke v_n?

Signatur für Graphen

Signatur: $\Sigma_{\text{graph}} := \{v, e, 0, 1, (,), \#\}$

Darstellung von

Ecke v_i als v gefolgt i binär kodiert

Kante $e_{i,j}$ als $e(string_1 \# string_2)$, wobei

- $string_1$ die binäre Darstellung von i,
- $string_2$ die binäre Darstellung von j

Darstellung des Erreichbarkeitsproblems in Graphen

Erreichbarkeitsproblem

Sprache

$$L_{\text{reach}} := \{w \in \Sigma^*_{\text{graph}} : \quad \text{es gibt einen Weg in } w \text{ von der ersten Ecke } v_1 \text{ zur letzten Ecke } v_n\}$$

Die Chomsky-Hierarchie

Die Chomsky-Hierarchie

Noam Chomsky (born 1928)

- Professor für Linguistik und Philosophie am MIT
- Bedeutender Linguist
- Bedeutender Beitrag zur Informatik:
  Erste Beschreibung der Chomsky-Hierarchie (1956)
- Bedeutender linker Intellektueller und Globalisierungskritiker

Die Chomsky-Hierarchie

Was muss eine Grammatik erfüllen?

- Sie darf nur **endlich viele Regeln** haben
- Jede Regelprämisse muss **mindestens eine Variable** enthalten

Die Chomsky-Hierarchie

Was muss eine Grammatik erfüllen?

- Sie darf nur **endlich viele Regeln** haben
- Jede Regelprämisse muss **mindestens eine Variable** enthalten

Das Wort kann im Lauf der Ableitung beliebig wachsen und wieder schrumpfen.

(Weitere) Beschränkung der Form, die Regeln haben dürfen, führt zu
– **Grammatiktypen** und damit auch zu
– **Sprachtypen**
von verschiedenen Schwierigkeitsgraden.

Die Chomsky-Hierarchie

Definition (Rechtlineare Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **rechtslinear** gdw

$$\forall (P \to Q) \in R \ \left(P \in V \text{ und } Q \in T^* \cup T^+ V\right)$$

Die Chomsky-Hierarchie

Definition (Rechtlineare Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **rechtslinear** gdw

$$\forall (P \to Q) \in R \ \left(P \in V \text{ und } Q \in T^* \cup T^+ V\right)$$

Das heißt, bei jeder Regelanwendung:

- Links eine **einzelne Variable**
- Rechts **höchstens eine Variable**
- Wenn rechts eine Variable steht, steht sie **ganz rechts im Wort**.

Die Chomsky-Hierarchie

Definition (Kontextfreie Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **kontextfrei** gdw

$$\forall (P \to Q) \in R \ \left(P \in V \text{ und } Q \in (V \cup T)^*\right)$$

Die Chomsky-Hierarchie

Definition (Kontextfreie Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **kontextfrei** gdw

$$\forall (P \to Q) \in R \ \left(P \in V \text{ und } Q \in (V \cup T)^*\right)$$

Das heißt, bei jeder Regelanwendung:

- Links eine **einzelne Variable**
- Die Prämisse macht keine Aussage, was der Kontext dieser Variablen ist („kontextfrei“)
- Rechts steht etwas beliebiges

Die Chomsky-Hierarchie

Definition (Kontextsensitive Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **kontextsensitiv** gdw

$\forall (P \to Q) \in R$:

1. $\exists u, v, \alpha \in (V \cup T)^* \; \exists A \in V \; \big(P = uAv \text{ und } Q = u\alpha v \text{ mit } |\alpha| \geq 1\big)$,
   oder die Regel hat die Form $S \to \varepsilon$
2. S nicht in Q

Die Chomsky-Hierarchie

Definition (Kontextsensitive Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **kontextsensitiv** gdw

$\forall (P \to Q) \in R$:

1. $\exists u, v, \alpha \in (V \cup T)^* \; \exists A \in V \; \big(P = uAv$ und $Q = u\alpha v$ mit $|\alpha| \geq 1\big)$, **oder** die Regel hat die Form $S \to \varepsilon$
2. S nicht in Q

Das heißt, bei jeder Regelanwendung:

- Eine Variable A wird in einen String α mit $|\alpha| \geq 1$ überführt
- Die Ersetzung von A durch α findet nur statt, wenn der in der Regel geforderte **Kontext (u und v), vorhanden ist**
- **Das Wort wird nicht kürzer, außer bei $\varepsilon \in L$**

Beschränkte Grammatik

Definition (Beschränkte Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **beschränkt** gdw

$\forall (P \to Q) \in R$:

1. $|P| \leq |Q|$, **oder**
   die Regel hat die Form $S \to \varepsilon$

2. S nicht in Q

Beschränkte Grammatik

Definition (Beschränkte Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **beschränkt** gdw

$\forall (P \to Q) \in R$:

1. $|P| \leq |Q|$, **oder**
   die Regel hat die Form $S \to \varepsilon$

2. S nicht in Q

Das heißt, bei jeder Regelanwendung:

- Die Conclusio ist mindestens so lang wie die Prämisse, außer bei $\varepsilon \in L$.
- Das Wort wird nicht kürzer, außer bei $\varepsilon \in L$

Die Chomsky-Hierarchie

Aufbauend auf den Grammatikarten kann man Sprachklassen definieren.

Definition (Sprachklassen)

Klasse	**definiert als**	**Sprache heißt**
$\mathbf{L_3}$, REG	$\{L(G) \mid G$ ist rechtslinear$\}$	Typ 3, **regulär**
$\mathbf{L_2}$, CFL	$\{L(G) \mid G$ ist kontextfrei$\}$	Typ 2, **kontextfrei**
$\mathbf{L_1}$, CSL	$\{L(G) \mid G$ ist kontextsensitiv$\}$	Typ 1, **kontextsensitiv**
$\mathbf{L_1}$, CSL	$\{L(G) \mid G$ ist beschränkt$\}$	Typ 1, **beschränkt**
$\mathbf{L_0}$, r.e.	$\{L(G) \mid G$ beliebig$\}$	Typ 0, **aufzählbar**
$\mathbf{L}$	$\{L \mid \ L \subseteq \Sigma^* \ \}$	**beliebige** Sprache

Die Chomsky-Hierarchie

Grammatiken können kompliziert sein!

Beispiel (Grammatik für $\{a^n b^n c^n \mid n \geq 1\}$):
Grammatik $G_{abc} = (\{S, X_1, X_2\}, \{a, b, c\}, \{R_1, \ldots R_5\}, S)$ mit

$$
\begin{array}{lllll}
R_1 = & S & \rightarrow & abc \mid aX_1bc \\
R_2 = & X_1b & \rightarrow & bX_1 \\
R_3 = & X_1c & \rightarrow & X_2bcc \\
R_4 = & bX_2 & \rightarrow & X_2b \\
R_5 = & aX_2 & \rightarrow & aa \mid aaX_1
\end{array}
$$

- Ist diese Grammatik **kontextsensitiv?**
- **Ist sie beschränkt?**

Probleme über Sprachen

Probleme über Sprachen

Interessante Probleme (informell)

- Ist ein gegebenes Wort in einer Sprache (definiert durch eine Grammatik) enthalten?
- Erzeugen zwei gegebene Grammatiken dieselbe Sprache?

Mit welchen Algorithmen können diese Probleme gelöst werden?

Probleme und Algorithmen im allgemeine

Definition (Problem, Algorithmus)
Ein **Problem** P ist die Frage, ob eine bestimmte Eigenschaft auf gegebene Objekte zutrifft.

Dabei ist eine bekannte, abzählbaren Grundmenge solcher Objekte gegeben.

Für jedes Objekt o gilt: die Eigenschaft trifft auf o zu oder nicht.

Ein **Algorithmus** für ein Problem P ist eine Vorschrift (ein Programm), die zu beliebigem Objekt o berechnet, ob die Eigenschaft für o zutrifft oder nicht.

Probleme und Algorithmen im allgemeinen

Beispiel (Einige Probleme)

- Für $n \in \mathbb{N}$:
  Ist n eine Primzahl?
- Für ein Wort $w \in \Sigma^*$ und
  ein Element G aus der Menge aller Grammatiken über Σ:
  Gilt $w \in L(G)$?
- Für ein Element G aus der Menge aller Grammatiken:
  Ist $L(G)$ leer (endlich, unendlich)?
- Für $(a, b, c) \in \mathbb{N}^3$:
  Hat $a^n + b^n = c^n$ eine Lösung in den natürlichen Zahlen?
- Für ein Programm p aus der Menge aller Java-Programme:
  Terminiert p?