

Theorie der Informatik

Was bisher geschah... (Teil II)

Malte Helmert Christian Tschudin

Universität Basel

10. April 2013

Wegweiser

Überblick: Vorlesung

Vorlesungsteile

- I. Logik
- II. Automatentheorie und formale Sprachen
- III. Berechenbarkeitstheorie
- IV. Komplexitätstheorie

Überblick: Vorlesung

Vorlesungsteile

- I. ~~Logik~~
- II. **Automatentheorie und formale Sprachen** ← Wiederholung!
- III. Berechenbarkeitstheorie
- IV. Komplexitätstheorie

Überblick: Automatentheorie und formale Sprachen

II. Automatentheorie und formale Sprachen

II.1. Grammatiken

II.2. Chomsky-Hierarchie

II.3. Reguläre Sprachen/endliche Automaten

II.4. Kontextfreie Sprachen/Kellerautomaten

Grammatiken

Grammatiken

Definition (Grammatik)

Eine **Grammatik** ist ein 4-Tupel (Σ, V, P, S) mit

- 1 Σ endliches Terminalalphabet,
- 2 V endliche Menge von Variablen (mit $V \cap \Sigma = \emptyset$),
- 3 $P \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^*$ endliche Menge von Regeln, und
- 4 $S \in V$ Startvariable.

Grammatiken

Definition (Ableitung)

Für $u, v \in (V \cup \Sigma)^*$: v kann von u abgeleitet werden ($u \Rightarrow v$), falls

- 1 $u = xyz$, $v = xy'z$ mit $x, z \in (V \cup \Sigma)^*$ und
- 2 es existiert eine Regel $y \rightarrow y' \in P$.

Wir schreiben: $u \Rightarrow^* v$ falls v in endlich vielen Schritten (d.h. durch die Anwendung von n Regeln für $n \in \mathbb{N}$) von u abgeleitet werden kann.

Definition (Sprache)

Die von einer Grammatik $G = (\Sigma, V, P, S)$ erzeugte **Sprache**

$$\mathcal{L}(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}$$

ist die Menge aller Wörter aus Σ^* , die von S mit endlich vielen Regelanwendungen abgeleitet werden können.

Grammatiken

Beispiel: Tafel

Chomsky-Hierarchie

Chomsky-Hierarchie

Grammatiken werden in die **Chomsky-Hierarchie** eingegliedert.

Definition (Chomsky-Hierarchie)

- Jede Grammatik ist vom Typ 0 (beliebige Regeln erlaubt).
- Grammatik ist vom Typ 1 (kontextsensitiv), falls für alle Regeln $w_1 \rightarrow w_2$ gilt: $|w_1| \leq |w_2|$.
- Grammatik ist vom Typ 2 (kontextfrei), falls für alle Regeln $w_1 \rightarrow w_2$ gilt: $w_1 \in V$ (einzelne Variable).
- Grammatik ist vom Typ 3 (regulär), falls zusätzlich $w_2 \in \Sigma \cup \Sigma V$.

Spezialfall: Regel $S \rightarrow \varepsilon$ ist immer erlaubt, wenn S Startsymbol ist und auf keiner rechten Seite irgendeiner Regel auftritt.

Chomsky-Hierarchie

Definition (Typ-0–3 Sprachen)

Eine Sprache $L \subseteq \Sigma^*$ heisst vom Typ 0 (Typ 1, Typ 2, Typ 3), falls es eine Typ-0- (Typ-1-, Typ-2-, Typ-3-) Grammatik G gibt mit $\mathcal{L}(G) = L$.

Beispiel: Tafel

Chomsky-Hierarchie

Im Folgenden

Genauere Untersuchung von

- Typ-3- (regulären) Sprachen
- Typ-2- (kontextfreien) Sprachen

Wie können wir solche Sprachen algorithmisch erkennen?

Wir haben in der Vorlesung gesehen:

- Reguläre Sprachen werden von **endlichen Automaten** erkannt.
- Kontextfreie Sprachen werden von **Kellerautomaten** erkannt.

Reguläre Sprachen

Reguläre Sprachen

Definition (deterministischer endlicher Automat = DFA)

Ein **deterministischer endlicher Automat** (deterministic finite automaton, **DFA**) ist ein 5-Tupel $M = (\Sigma, Z, z_0, E, \delta)$ mit

- Σ das Eingabealphabet,
- Z die Menge der Zustände (mit $Z \cap \Sigma = \emptyset$),
- $z_0 \in Z$ der Startzustand,
- $E \subseteq Z$ die Menge der Endzustände,
- $\delta : Z \times \Sigma \rightarrow Z$ die Überföhrungsfunktion.

Reguläre Sprachen

Sei $M = (\Sigma, Z, z_0, E, \delta)$ endlicher Automat.

Definition (erkanntes Wort bei DFAs)

M **erkennt das Wort** $w = a_1 \dots a_n$ genau dann, wenn eine Folge von Zuständen $z'_0, \dots, z'_n \in Z$ existiert mit

- ① $z'_0 = z_0$,
- ② $\delta(z'_{i-1}, a_i) = z'_i$ für alle $i \in \{1, \dots, n\}$ und
- ③ $z'_n \in E$.

Definition (akzeptierte Sprache eines DFA)

Sei M ein endlicher Automat. Die von M **akzeptierte Sprache** ist definiert durch $T(M) = \{w \in \Sigma^* \mid w \text{ wird von } M \text{ erkannt}\}$.

Beispiel: Tafel

Reguläre Sprachen

Satz

Jede durch endliche Automaten erkennbare Sprache ist regulär.

Gilt auch die Rückrichtung? D.h., gibt es für jede reguläre Sprache $\mathcal{L}$ einen endlichen Automaten, der $\mathcal{L}$ erkennt?

Ja! Um dies zu zeigen $\rightsquigarrow$ **nichtdeterministische** endliche Automaten

Reguläre Sprachen

Definition (nichtdeterministischer endlicher Automat = NFA)

Ein **nichtdeterministischer** endlicher Automat (nondeterministic finite automaton, **NFA**) ist ein 5-Tupel $M = (\Sigma, Z, S, E, \delta)$ mit

- Σ das Eingabealphabet,
- Z die Menge der Zustände (mit $Z \cap \Sigma = \emptyset$),
- $S \subseteq Z$ **die Menge** der Startzustände,
- $E \subseteq Z$ die Menge der Endzustände,
- $\delta : Z \times \Sigma \rightarrow \mathcal{P}(Z)$ die Überföhrungsfunktion in die **Potenzmenge** von Z .

Unterschiede zu DFAs:

- 1 **mehrere** Startzustände möglich
- 2 δ kann für **ein** $a \in \Sigma$ in **mehrere** Nachfolgezustände föhren

Reguläre Sprachen

Sei $M = (\Sigma, Z, S, E, \delta)$ nichtdeterministischer endlicher Automat.

Definition (erkanntes Wort bei NFAs)

M **erkennt das Wort** $w = a_1 \dots a_n$ genau dann, wenn eine Folge von Zuständen $z'_0, \dots, z'_n \in Z$ existiert mit

- ① $z'_0 \in S$,
- ② $z'_i \in \delta(z'_{i-1}, a_i)$ für alle $i \in \{1, \dots, n\}$ und
- ③ $z'_n \in E$.

Definition (akzeptierte Sprache eines NFA)

Sei M ein NFA. Die von M **akzeptierte Sprache** ist definiert durch $T(M) = \{w \in \Sigma^* \mid w \text{ wird von } M \text{ erkannt}\}$.

Beispiel: Tafel

Reguläre Sprachen

Satz (Äquivalenz DFA und NFA)

Jede von einem NFA akzeptierbare Sprache ist auch durch einen DFA akzeptierbar.

(„NFAs und DFAs sind gleichmächtig.“)

Beweis: Zu jedem NFA $M = (\Sigma, Z, S, E, \delta)$ kann man einen DFA $M' = (\Sigma, Z', z'_0, E', \delta')$ bauen mit $T(M) = T(M')$. Hierbei ist M' wie folgt definiert:

- $Z' := \mathcal{P}(Z)$ (die Potenzmenge von Z)
- $z'_0 := S$
- $E' := \{Z \subseteq Z \mid Z \cap E \neq \emptyset\}$
- Für alle $Z \in Z'$: $\delta'(Z, a) := \bigcup_{z \in Z} \delta(z, a)$

Reguläre Sprachen

Satz

Für jede reguläre Grammatik G gibt es einen NFA M mit $\mathcal{L}(G) = T(M)$.

Insgesamt folgt damit:

Folgerung

$\mathcal{L}$ regulär $\iff \mathcal{L}$ wird von einem endlichen Automaten erkannt.

Reguläre Sprachen

Andere Möglichkeit, Sprachen zu definieren: **reguläre Ausdrücke**

Definition (regulärer Ausdruck)

Für ein Alphabet Σ sind reguläre Ausdrücke induktiv definiert:

- $\emptyset$ ist ein regulärer Ausdruck
- ε ist ein regulärer Ausdruck
- Für jedes $a \in \Sigma$ ist a ein regulärer Ausdruck
- Für reguläre Ausdrücke α und β sind auch $\alpha\beta$, $(\alpha|\beta)$ und $(\alpha)^*$ reguläre Ausdrücke.

Reguläre Sprachen

Jeder reguläre Ausdruck definiert eine Sprache.

Definition (durch regulären Ausdruck beschriebene Sprache)

Sei γ regulärer Ausdruck. Die Sprache $\mathcal{L}(\gamma)$ ist induktiv definiert:

- Falls $\gamma = \emptyset$, dann $\mathcal{L}(\gamma) := \emptyset$
- Falls $\gamma = \varepsilon$, dann $\mathcal{L}(\gamma) := \{\varepsilon\}$
- Falls $\gamma = a$, dann $\mathcal{L}(\gamma) := \{a\}$
- Falls $\gamma = \alpha\beta$, dann $\mathcal{L}(\gamma) := \{xy \mid x \in \mathcal{L}(\alpha) \text{ und } y \in \mathcal{L}(\beta)\}$
- Falls $\gamma = (\alpha|\beta)$, dann $\mathcal{L}(\gamma) := \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
- Falls $\gamma = (\alpha)^*$, dann $\mathcal{L}(\gamma) := \mathcal{L}(\alpha)^*$

Beispiel: Tafel

Reguläre Sprachen

Satz (Kleene)

Die Menge der durch reguläre Ausdrücke beschreibbaren Sprachen ist genau die Menge der regulären Sprachen.

Reguläre Sprachen

Wie kann man beweisen, dass eine Sprache $\mathcal{L}$ **nicht** regulär ist?

- Man kann versuchen, zu beweisen, dass keine reguläre Grammatik existiert, die $\mathcal{L}$ erzeugt $\rightsquigarrow$ i.A. schwierig
- **Pumping Lemma**: Liefert notwendige Eigenschaft, die für alle regulären Sprachen gelten muss

Satz (Pumping Lemma)

Sei $\mathcal{L}$ eine reguläre Sprache. Dann gibt es ein $n \in \mathbb{N}$, so dass sich alle Wörter $x \in \mathcal{L}$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass folgendes gilt:

- 1 $|v| \geq 1$,
- 2 $|uv| \leq n$, und
- 3 für alle $i = 0, 1, 2, \dots$ gilt: $uv^i w \in \mathcal{L}$.

Reguläre Sprachen

Pumping Lemma (PL): Verwendung

- Wenn $\mathcal{L}$ regulär, dann gilt PL für $\mathcal{L}$
- Wenn PL für $\mathcal{L}$ gilt, kann man nichts über $\mathcal{L}$ aussagen
- Aber: Wenn PL für $\mathcal{L}$ nicht gilt, kann $\mathcal{L}$ nicht regulär sein
- D.h.: Wenn es kein $n \in \mathbb{N}$ mit den geforderten Eigenschaften für PL gibt, dann kann $\mathcal{L}$ nicht regulär sein.

Beispiel: Ist $\mathcal{L} = \{a^n b^n \mid n \in \mathbb{N}\}$ regulär? $\rightsquigarrow$ Tafel

Kontextfreie Sprachen

Kontextfreie Sprachen

Wiederholung: Kontextfreie Sprache

- Eine Sprache $\mathcal{L}$ heisst **kontextfrei**, falls sie von einer kontextfreien Grammatik erzeugt wird.
- Eine Grammatik $G = (\Sigma, V, P, S)$ heisst kontextfrei, falls für alle Regeln $w_1 \rightarrow w_2 \in P$ gilt:
 $w_1 \in V$ (einzelne Variable) und $|w_1| \leq |w_2|$.

Wie können kontextfreie Sprachen algorithmisch erkannt werden?

↪ **Kellerautomaten**

Kontextfreie Sprachen

Definition (Kellerautomat = PDA)

Ein **Kellerautomat** (push-down automaton, **PDA**) ist ein 6-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ mit

- Z endliche Menge der Zustände,
- Σ das Eingabealphabet,
- Γ das Kelleralphabet,
- $\delta : Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow \mathcal{P}_e(Z \times \Gamma^*)$ die Überföhrungsfunktion (mit $\mathcal{P}_e$ Menge aller **endlichen** Teilmengen)
- $z_0 \in Z$ der Startzustand
- $\# \in \Gamma$ das unterste Kellerzeichen

Wichtigster Unterschied zu DFAs und NFAs:
potentiell unbeschränkter Speicher in Form eines Stacks („Keller“).

Kontextfreie Sprachen

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ Kellerautomat.

Was bedeutet Übergangsfunktion δ intuitiv?

- $(z', B_1, \dots, B_k) \in \delta(z, a, A)$: Wenn M im Zustand z das Zeichen a liest und A das oberste Kellerzeichen ist, dann **kann** M im nächsten Schritt in z' übergehen und A durch $B_1 \dots B_k$ ersetzen (danach B_1 oberstes Kellerzeichen)
- Spezialfall $a = \varepsilon$ zugelassen (spontaner Übergang)

weggelassen: formale Definition von Konfiguration und Übergang

Definition (erkanntes Wort bei PDAs)

M **erkennt das Wort** $w = a_0 \dots a_n$ genau dann, wenn M durch endliches Anwenden von δ in eine Konfiguration $(z, \varepsilon, \varepsilon)$ übergehen kann (**Wort verarbeitet** und **Keller leer**).

Kontextfreie Sprachen

Definition (akzeptierte Sprache eines PDAs)

Sei M ein Kellerautomat. Die von M **akzeptierte Sprache** ist definiert durch $T(M) = \{w \in \Sigma^* \mid w \text{ wird von } M \text{ erkannt}\}$.

Beispiel: Tafel

Kontextfreie Sprachen

Satz

*Eine Sprache $\mathcal{L}$ ist genau dann kontextfrei,
wenn $\mathcal{L}$ von einem Kellerautomaten erkannt wird.*

Zusammenfassung

Zusammenfassung

Mit welchen Beschreibungsmitteln kann welcher Sprachtyp dargestellt werden?

Typ 3: äquivalente Beschreibungen

- reguläre Grammatik
- deterministischer endlicher Automat
- nichtdeterministischer endlicher Automat
- regulärer Ausdruck

Typ 2: äquivalente Beschreibungen

- kontextfreie Grammatik
- Kellerautomat

Nächste Woche: Wiederholung Berechenbarkeitstheorie