Naturwissenschaftlich motivierte formale Modelle

Systematisierung Formale Sprachen

Institut für Informatik und Computational Science
Universität Potsdam

Henning Bordihn

1.2 Grammatiken

Regelgrammatiken (Typ-0-Grammatiken)

- $G = (N, T, P, S)$
- N Alphabet (der Nichtterminale), T Alpahbet (der Terminale), $N \cap T = \emptyset$,
- $V := N \cup T$ (Gesamtalphabet)
- P endliche Teilmenge von $(V^* \setminus T^*) \times V^*$
  (Menge der Regeln der Form $\alpha \to \beta$)
- $S \in N$ – Axiom/Startsymbol

Ableitungen und erzeugte Sprache

- $\gamma \Longrightarrow \gamma'$ gdw. $\gamma = \delta_1 \alpha \delta_2$, $\gamma' = \delta_1 \beta \delta_2$, $\alpha \to \beta \in P$, δ_1, $\delta_2 \in V^*$
- $\gamma \overset{n}{\Longrightarrow} \gamma'$ gdw. $\gamma \Longrightarrow \gamma_1 \Longrightarrow \gamma_2 \Longrightarrow \ldots \Longrightarrow \gamma_n = \gamma'$
- $\gamma \overset{*}{\Longrightarrow} \gamma'$ gdw. $\gamma \overset{n}{\Longrightarrow} \gamma'$ mit $n \geq 0$
- $\gamma \overset{+}{\Longrightarrow} \gamma'$ gdw. $\gamma \overset{n}{\Longrightarrow} \gamma'$ mit $n > 0$
- $L(G) = \{\, w \in T^* \mid S \overset{*}{\Longrightarrow} w \,\}$

Chomsky-Typen

- Eine Grammatik $G = N, T, P, S)$ heißt
  - **monoton** (Typ-1) gdw. $|\alpha| \leq |\beta|$,
  - **kontextsensitiv** (Typ-1) gdw. $\alpha = \alpha_1 A \alpha_2$, $\beta = \alpha_1 \nu \alpha_2$ mit $A \in N$, $\nu \in V^+$, $\alpha_1, \alpha_2 \in V^*$,
  - **kontextfrei** (Typ-2) gdw. $\alpha \in N$,
  - **regulär** (Typ-3) gdw. $\alpha \in N$ und $\beta \in T^* N \cup T^*$,
  - λ-frei gdw. $\beta \neq \lambda$

  für alle $\alpha \to \beta \in P$.

- $\mathcal{L}(\mathrm{REG}) \subseteq \mathcal{L}(\mathrm{CF}) \subseteq \mathcal{L}(\mathrm{RE})$ und $\mathcal{L}(\mathrm{CS}) \subseteq \mathcal{L}(\mathrm{MON}) \subseteq \mathcal{L}(\mathrm{RE})$

Chomsky-Hierarchie

$$\mathcal{L}(\mathrm{REG}) \subset \mathcal{L}(\mathrm{CF}) \subset \mathcal{L}(\mathrm{CS}) = \mathcal{L}(\mathrm{MON}) \subset \mathcal{L}(\mathrm{RE})$$

1. Jede λ-freie kontextfreie Grammatik ist kontextsensitiv.

2. Zu jeder kontextfreien Grammatik kann eine äquivalente kontextfreie Grammatik konstruiert werden, die λ-frei ist.

3. Zu jeder monotonen Grammatik kann eine äquivalente Grammatik konstruiert werden, die kontextsensitiv ist.

G und G' heißen **äquivalent**, wenn $L(G) = L(G')$ gilt.

Beseitigung löschender Regeln in Typ-2-Grammatiken

$$
\begin{aligned}
M_\lambda &= \{\, A \in N \mid A \stackrel{*}{\Longrightarrow} \lambda \,\} = \bigcup_{i \geq 1} M_i \\
M_1 &= \{\, A \in N \mid A \rightarrow \lambda \in P \,\} \\
M_{i+1} &= M_i \cup \{\, A \in N \mid A \rightarrow \beta \in P \text{ mit } \beta \in M_i^* \,\}
\end{aligned}
$$

Beseitigung löschender Regeln in Typ-2-Grammatiken

$$\begin{aligned} M_\lambda &= \{\, A \in N \mid A \overset{*}{\Longrightarrow} \lambda \,\} = \bigcup_{i \geq 1} M_i \\ M_1 &= \{\, A \in N \mid A \to \lambda \in P \,\} \\ M_{i+1} &= M_i \cup \{\, A \in N \mid A \to \beta \in P \text{ mit } \beta \in M_i^* \,\} \end{aligned}$$

Ersetze jede Regel $A \to u_0 B_1 u_1 B_2 u_2 \ldots B_k u_k$ mit $B_i \in M_\lambda$, $|u_i|_{M_\lambda} = 0$ durch $\{\, A \to u_0 X_1 u_1 X_2 u_2 \ldots X_k u_k \mid X_i \in \{B_i, \lambda\},\ 1 \leq i \leq k \,\}$.

1.3 Automaten

Endliche Automaten (NFA)

$A = (Q, \Sigma, q_0, F, \delta)$

- Q Alphabet der Zustände, Σ Alphabet der Eingabesymbole, $Q \cap \Sigma = \emptyset$
- $q_0 \in Q$ (Anfangszustand)
- $F \subseteq Q$ (akzeptierende Zustände)
- $\delta : Q \times \Sigma \to 2^Q$ (Überführungsfunktion)

$\hookrightarrow$ A **deterministisch (DFA)** gdw. $\delta : Q \times \Sigma \to Q$

Erweiterte Überführungsfunktion und akzeptierte Sprache

- $\delta^* : Q \times \Sigma^* \to 2^Q$ mit

$$\begin{aligned} \delta^*(q, \lambda) &= \{q\} \\ \delta^*(q, wa) &= \bigcup_{p \in \delta^*(q,w)} \delta(p, a) \end{aligned}$$

für alle $q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$

Erweiterte Überführungsfunktion und akzeptierte Sprache

- $\delta^* : Q \times \Sigma^* \to 2^Q$ mit

$$\begin{array}{rcl} \delta^*(q,\lambda) & = & \{q\} \\ \delta^*(q,wa) & = & \bigcup\limits_{p \in \delta^*(q,w)} \delta(p,a) \end{array}$$

  für alle $q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$

- $L(A) = \{\, w \in \Sigma^* \mid \delta^*(q_0, w) \cap F \neq \emptyset \,\}$

Sprachfamilien

$$\begin{aligned}
\mathcal{L}(\mathrm{NFA}_\lambda) &= \{\, L \mid L = L(A) \text{ für einen NFA mit } \lambda\text{-Übergängen} \,\} \\
\mathcal{L}(\mathrm{NFA}) &= \{\, L \mid L = L(A) \text{ für einen NFA ohne } \lambda\text{-Übergänge} \,\} \\
\mathcal{L}(\mathrm{DFA}) &= \{\, L \mid L = L(A) \text{ für einen DFA} \,\}
\end{aligned}$$

$$\mathcal{L}(\mathrm{NFA}_\lambda) = \mathcal{L}(\mathrm{NFA}) = \mathcal{L}(\mathrm{DFA}) = \mathcal{L}(\mathrm{REG})$$

Reguläre Ausdrücke

Diese Sprachfamilie $\mathcal{L}(\mathrm{REG})$ enthält genau die Sprachen,
die durch reguläre Ausdrücke beschrieben werden können, d.h., die sich aus

$\emptyset$, $\{\lambda\}$ und $\{a\}$ (für alle Buchstaben a des Alphabets)

durch (wiederholte) Anwendung der Operationen

Vereinigung ($\cup$), *Konkatenation* ($\cdot$) und *Kleene-Abschluss* ($*$)

gewinnen lassen.

Beispiel:

$$\begin{aligned} (ab)^*b + aa^* &= ((\{a\} \cdot \{b\})^* \cdot \{b\}) \cup \{a\} \cdot \{a\}^* \\ &= \{\, (ab)^n b \mid n \geq 0 \,\} \cup \{\, a^n \mid n \geq 1 \,\} \end{aligned}$$

Turing-Maschinen

- Endliche Automaten + zusätzliche Fähigkeiten:
    1. Kopfbewegungen nach rechts und links möglich
    2. Zellen auch jenseits des Eingabewortes können betreten werden
    3. gelesene Symbole (und leere Zellen) können überschrieben werden

- Stoppzustände (statt akzeptierender Zustände)

- $L(A) = \{\, w \in \Sigma^* \mid$ Anfangskonfiguration mit w kann in eine Konfiguration mit einem Stoppzustand überführt werden $\}$

- Berechnungsmodell (Bandinhalt bei Stopp ist der berechnete Funktionswert)

Mehrband-Turing-Maschinen und ihre Mächtigkeit

- „Schreibarbeit“ nicht auf dem *Eingabeband* (read-only), sondern auf seperaten *Arbeitsbändern* (read-write).
- Turing-Maschinen und Mehrband-Turing-Maschinen sind äquivalent.
- Nichtdeterministische Turing-Maschinen können durch deterministische simuliert werden.
- Sie akzeptieren genau die Typ-0-Sprachen ($\mathcal{L}(RE)$).

Platzbeschränkte Turing-Maschinen

- Sei M eine Turing-Maschine und $f : \mathbb{N} \to \mathbb{N}$.
  Werden bei jeder Eingabe w maximal $f(|w|)$ viele Zellen (auf jedem Band) benutzt/betreten, dann heißt M durch die Funktion f *platzbeschränkt*.

- Ist f eine lineare Funktion, so ist M ein linear beschränkter Automat (LBA).

- Die nichtdeterministischen LBAs akzeptieren genau die Typ-1-Sprachen.

Kellerautomaten/Pushdown-Automaten

- NFA mit λ-Übergängen + zusätzliches *Kellerband*:
  1. Die Übergänge hängen vom Zustand, dem gelesenen Eingabesymbol und dem obersten Kellersymbol ab.
  2. In einem Übergang wird das oberste Kellersymbol durch ein Wort (über dem Kelleralphabet) ersetzt.
- Nichtdeterministische Kellerautomaten akzeptieren genau die Typ-2-Sprachen.