Grammatiken: Abschlusseigenschaften und Entscheidungsprobleme

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik

Stand: 25. April 2023
Folien ursprünglich von PD Dr. David Sabel

Wiederholung: Definition einer Grammatik

Definition (Grammatik)

Eine Grammatik ist ein 4-Tupel $G = (V, \Sigma, P, S)$ mit

- V ist eine endliche Menge von Variablen
  (alternativ Nichtterminale, Nichtterminalsymbole)
- Σ (mit $V \cap \Sigma = \emptyset$) ist ein Alphabet von Zeichen
  (alternativ Terminale, Terminalsymbole)
- P ist eine endliche Menge von Produktionen von der Form $\ell \to r$ wobei $\ell \in (V \cup \Sigma)^+$ und $r \in (V \cup \Sigma)^*$
  (alternativ Regeln)
- $S \in V$ ist das Startsymbol
  (alternativ Startvariable)

Oft genügt es, P alleine zu notieren
(wenn klar ist, was Variablen, Zeichen und Startsymbol sind)

Wiederholung: Die Chomsky-Hierarchie

Sei $G = (V, \Sigma, P, S)$ eine Grammatik.

G ist vom Typ 0

G ist automatisch vom Typ 0

G ist vom Typ 1 (kontextsensitive Grammatik), wenn ...

für alle $(\ell \to r) \in P$: $|\ell| \leq |r|$

G ist vom Typ 2 (kontextfreie Grammatik), wenn ...

G ist vom Typ 1 und für alle $(\ell \to r) \in P$ gilt: $\ell \in V$

G ist vom Typ 3 (reguläre Grammatik), wenn ...

G ist vom Typ 2 und für alle $(A \to r) \in P$ gilt: $r = a$ oder $r = aA'$ für $a \in \Sigma, A' \in V$ (die rechten Seiten sind Wörter aus $\Sigma \cup \Sigma V$)

Kontextfrei (Typ 2) vs. kontextsensitiv (Typ 1)

- Kontextfreie Produktionen $A \to r$ sind immer auf ein Vorkommen von A anwendbar.

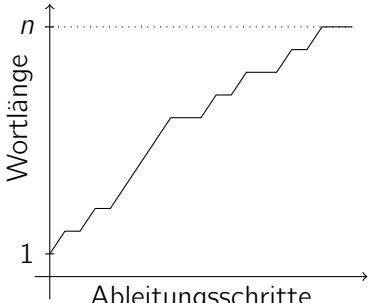
- Kontextsensitive Produktionen können solche Ersetzungen

  auf einen Kontext einschränken

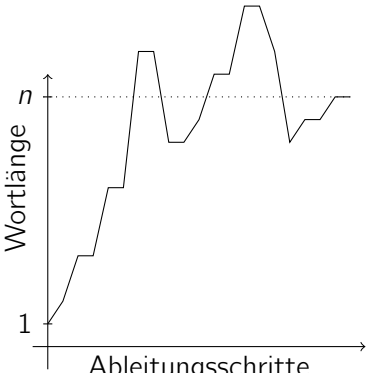
  und erlauben Regeln $uAv \to urv$, die die Ersetzung von A durch r nur erlauben, wenn A durch u und v umrahmt ist.

Typ 0 vs. kontextsensitiv (Typ 1)

Ableitung eines Wortes der Länge n



mit Typ 1-Grammatik



mit Typ 0-Grammatik

Syntaxbäume

Definition (Syntaxbaum)

Sei $G = (V, \Sigma, P, S)$ eine Typ 2- (oder Typ 3-)Grammatik und

$$S \Rightarrow w_0 \Rightarrow \cdots \Rightarrow w_n$$

eine Ableitung von $w_n \in \Sigma^*$.

Der **Syntaxbaum** zur Ableitung wird wie folgt erstellt:

- Die Wurzel des Baums ist mit S markiert.
- Wenn $w_i \Rightarrow w_{i+1}$, $w_i = uAv$ und $w_{i+1} = urv$ (Produktion $A \to r$ verwendet), dann erzeuge im Syntaxbaum $|r|$ viele Knoten als Kinder des mit A markierten Knotens. Markiere die Kinder mit den Symbolen aus r (in der Reihenfolge von links nach rechts).

Die Blätter sind daher genau mit dem Wort w_n markiert.

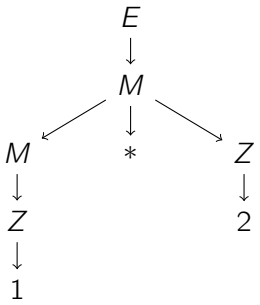
Beispiel

$$G = (\{E, M, Z\}, \{+, *, 1, 2, (,)\}, P, E) \text{ mit}$$
$$P = \{E \to M,\ \ E \to E + M,\ \ M \to Z,\ \ M \to M * Z,$$
$$Z \to 1,\ \ Z \to 2,\ \ Z \to (E)\}$$

Beide Ableitungen:

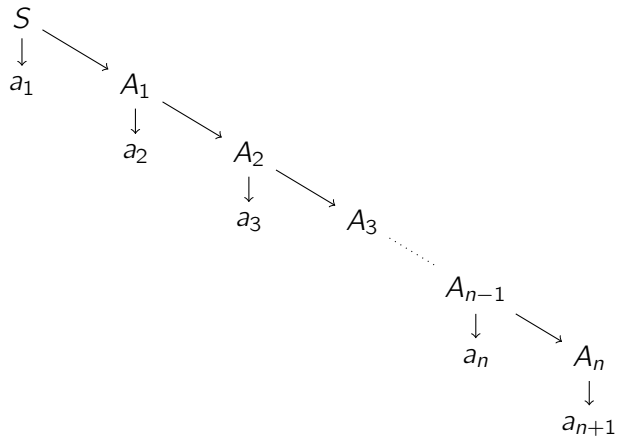
- $E \Rightarrow M \Rightarrow M * Z \Rightarrow Z * Z \Rightarrow 1 * Z \Rightarrow 1 * 2$ und
- $E \Rightarrow M \Rightarrow M * Z \Rightarrow M * 2 \Rightarrow Z * 2 \Rightarrow 1 * 2$

haben **denselben** Syntaxbaum.



Syntaxbäume bei Typ 3-Grammatiken

Syntaxbäume bei Typ 3-Grammatiken sind immer listenartig:



Links- und Rechtsableitungen (1)

- **Linksableitung**: Ersetze immer die linkeste Variable der Satzform.
- **Rechtsableitung**: Ersetze immer die rechteste Variable der Satzform.

Beispiele:

$$
\begin{aligned}
E &\Rightarrow E + M \\
&\Rightarrow M + M \\
&\Rightarrow M * Z + M \\
&\Rightarrow Z * Z + M \\
&\Rightarrow 1 * Z + M \\
&\Rightarrow 1 * 2 + M \\
&\Rightarrow 1 * 2 + Z \\
&\Rightarrow 1 * 2 + 3
\end{aligned}
\qquad
\begin{aligned}
E &\Rightarrow E + M \\
&\Rightarrow E + Z \\
&\Rightarrow E + 3 \\
&\Rightarrow M + 3 \\
&\Rightarrow M * Z + 3 \\
&\Rightarrow M * 2 + 3 \\
&\Rightarrow Z * 2 + 3 \\
&\Rightarrow 1 * 2 + 3
\end{aligned}
$$

Links- und Rechtsableitungen (2)

Satz

Sei G eine Typ 2-Grammatik und $w \in L(G)$. Dann gibt es eine Linksableitung (und eine Rechtsableitung) von w.

Links- und Rechtsableitungen (2)

Satz

Sei G eine Typ 2-Grammatik und $w \in L(G)$. Dann gibt es eine Linksableitung (und eine Rechtsableitung) von w.

Beweis:

- Da $w \in L(G)$, gibt es irgendeine Ableitung von w.
- Konstruiere Syntaxbaum zu dieser Ableitung.
- Lies Links- bzw. Rechtsableitung am Syntaxbaum ab. □

Backus-Naur-Form

Erweiterte Backus-Naur-Form (EBNF)

Für Typ 2-Grammatiken erlauben wir abkürzende Schreibweise für die Menge der Produktionen P:

1. Statt $A \to w_1, A \to w_2, \ldots, A \to w_n$ schreiben wir auch $A \to w_1 \mid w_2 \mid \cdots \mid w_n$
2. Die Schreibweise $A \to u[v]w$ steht für die beiden Produktionen $A \to uvw$ und $A \to uw$ (d.h. $[v]$ meint, dass v optional ist).
3. Die Schreibweise $A \to u\{v\}w$ steht für $A \to uw$ oder $A \to uBw$ mit $B \to v \mid vB$ (d.h. $\{v\}$ meint, dass v beliebig oft wiederholt werden kann).

Grammatiken, die diese Notation verwenden, nennen wir auch Grammatiken in erweiterter Backus-Naur-Form (EBNF)

Chomsky-Hierarchie: Teilmengenbeziehungen

Aus der Definition der Typ i-Sprachen folgt:

$$\text{Typ 3-Sprachen} \subseteq \text{Typ 2-Sprachen} \subseteq \text{Typ 1-Sprachen} \subseteq \text{Typ 0-Sprachen}$$

Chomsky-Hierarchie: Teilmengenbeziehungen

Aus der Definition der Typ i-Sprachen folgt:

$$\text{Typ 3-Sprachen} \subseteq \text{Typ 2-Sprachen} \subseteq \text{Typ 1-Sprachen} \subseteq \text{Typ 0-Sprachen}$$

Es gilt sogar:

$$\text{Typ 3-Sprachen} \subset \text{Typ 2-Sprachen} \subset \text{Typ 1-Sprachen} \subset \text{Typ 0-Sprachen}$$

Chomsky-Hierarchie: Teilmengenbeziehungen

Aus der Definition der Typ i-Sprachen folgt:

$$\text{Typ 3-Sprachen} \subseteq \text{Typ 2-Sprachen} \subseteq \text{Typ 1-Sprachen} \subseteq \text{Typ 0-Sprachen}$$

Es gilt sogar:

$$\text{Typ 3-Sprachen} \subset \text{Typ 2-Sprachen} \subset \text{Typ 1-Sprachen} \subset \text{Typ 0-Sprachen}$$

Trennende Beispiele sind (Beweise folgen im Laufe der Vorlesung):

- $L = \{a^n b^n \mid n \in \mathbb{N}_{>0}\}$ ist von Typ 2, aber nicht von Typ 3
- $L = \{a^n b^n c^n \mid n \in \mathbb{N}_{>0}\}$ ist von Typ 1, aber nicht von Typ 2
- $H = \{w\$x \mid \text{Turingmaschine } M_w \text{ hält für Eingabe } x\}$
  (das sogenannte Halteproblem) ist von Typ 0, aber nicht von Typ 1

Beachte: Es gibt auch Sprachen, die nicht Typ 0 sind:
Das Komplement von H ist eine solche Sprache.

Abgeschlossenheit von Sprachen

Eine Klasse $\mathcal{K}$ von Sprachen (d.h. eine Menge von Mengen)
heißt abgeschlossen bezüglich

- Vereinigung g.d.w. aus $L_1, L_2 \in \mathcal{K}$ folgt stets $L_1 \cup L_2 \in \mathcal{K}$,
- Schnittbildung g.d.w. aus $L_1, L_2 \in \mathcal{K}$ folgt stets $L_1 \cap L_2 \in \mathcal{K}$,
- Komplementbildung g.d.w. aus $L \in \mathcal{K}$ folgt stets $\overline{L} \in \mathcal{K}$ und
- Produktbildung g.d.w. aus $L_1, L_2 \in \mathcal{K}$ folgt stets $L_1 L_2 \in \mathcal{K}$.

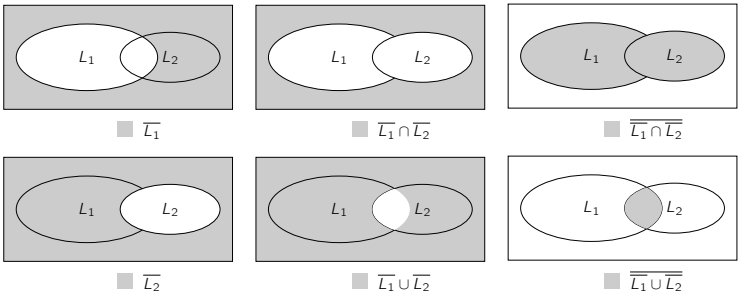
Wir werden im Laufe der Vorlesung untersuchen, ob die Typ i-Sprachen
abgeschlossen bezüglich obiger Operationen sind.

Abgeschlossenheit: Eigenschaften

Satz

Sei die Klasse von Sprachen $\mathcal{K}$ abgeschlossen bezüglich Komplementbildung. Dann ist $\mathcal{K}$ abgeschlossen bezüglich Schnittbildung g.d.w. $\mathcal{K}$ abgeschlossen bezüglich Vereinigung ist.

Das gilt, da: $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$ und $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$.



Entscheidbarkeit

Definition (Entscheidbarkeit)

Eine Sprache L heißt **entscheidbar**, wenn es einen Algorithmus gibt, der bei Eingabe eines Wortes w in endlicher Zeit feststellt, ob $w \in L$ gilt oder nicht.

Entscheidbarkeit

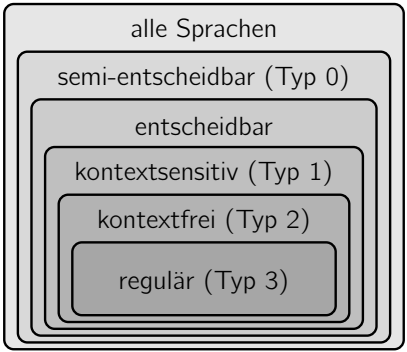
Definition (Entscheidbarkeit)

Eine Sprache L heißt **entscheidbar**, wenn es einen Algorithmus gibt, der bei Eingabe eines Wortes w in endlicher Zeit feststellt, ob $w \in L$ gilt oder nicht.

Eigenschaften der Typ i-Sprachen:

- Alle Typ 1, 2, 3-Sprachen sind **entscheidbar**.
- Es gibt Typ 0-Sprachen, die **nicht entscheidbar** sind.
- Alle Typ 0-Sprachen sind semi-entscheidbar (rekursiv aufzählbar):
  Für jede Typ 0-Sprache L gibt es einen Algorithmus, der bei Eingabe eines Wortes $w \in L$ in endlicher Zeit feststellt, dass $w \in L$ gilt, und bei einem Wort $w \notin L$ entweder feststellt, dass $w \notin L$ gilt, **oder nicht terminiert.**

Übersicht über die Sprachen



- Die Menge der Typ 0-Grammatiken ist abzählbar
  (jede Grammatik hat eine endliche Beschreibung,
  d.h. Grammatiken können der Größe nach aufgezählt werden).
- Menge aller Sprachen $= \mathcal{P}(\Sigma^*)$ ist überabzählbar.

Weitere Entscheidungsprobleme

Leerheitsproblem

Das Leerheitsproblem für Sprachen vom Typ i ist die Frage, ob für eine Typ i-Grammatik G die Gleichheit $L(G) = \emptyset$ gilt.

Endlichkeitsproblem

Das Endlichkeitsproblem für Sprachen vom Typ i ist die Frage, ob für eine Typ i-Grammatik G die Ungleichheit $|L(G)| < \infty$ gilt.

Schnittproblem

Das Schnittproblem für Sprachen vom Typ i ist die Frage, ob für Typ i-Grammatiken G_1, G_2 gilt: $L(G_1) \cap L(G_2) = \emptyset$.

Äquivalenzproblem

Das Äquivalenzproblem für Sprachen vom Typ i ist, die Frage, ob Typ i-Grammatiken G_1, G_2 gilt: $L(G_1) = L(G_2)$.

Typ i-Sprachen aus praktischer Sicht

Aus Informatik-Sicht:

- Typ 2- und Typ 3-Sprachen sind wichtig im Rahmen des Compilerbaus (syntaktische und lexikalische Analyse)
- Viele Fragestellungen sind jedoch kontextsensitiv oder Typ 0
- Praktisches Vorgehen: Nutze Typ 2-Sprache und Nebenbedingungen (z.B. Syntax als kontextfreie Grammatik und Nebenbedingungen, die prüfen, dass alle Variablen deklariert wurden o.ä.)