Grammatiken

Eine Grammatik G mit Alphabet Σ besteht aus:

- Variablen V
- Startsymbol $S \in V$
- Produktionen $P \subseteq \left((V \cup \Sigma)^* \setminus \Sigma^*\right) \times (V \cup \Sigma)^*$

Kurzschreibweise $G = (V, \Sigma, P, S)$

Schreibweise für Produktion $(\alpha, \beta) \in P$: $\alpha \to \beta$

Grammatiken wie hier definiert sind auch unter dem Namen *Phrasenstrukturgrammatiken* bekannt. Sie wurden in den 1960er Jahren von Noam Chomsky in der Sprachwissenschaft zur Beschreibung der Struktur von natürlichsprachlichen Sätzen eingeführt. Unabhängig davon wurden sehr ähnliche Formalismen in Informatik zur Definition der Syntax von Programmiersprachen eingeführt, zuerst durch J.W. Backus für FORTRAN und P. Naur für ALGOL 60.

Jede Produktion ist ein Paar $(\alpha, \beta) \in P$ von Strings aus $(V \cup \Sigma)^*$, wobei die linke Seite α nichtleer ist und mindestens eine Variable enthält. Sie ist zu lesen als: ersetze α durch β.

Ableitung in einem Schritt

Für eine Grammatik $G = (V, \Sigma, P, S)$ wird definiert:

Für $\gamma_1, \gamma_2 \in (V \cup \Sigma)^*$ gilt $\gamma_1 \Rightarrow_G \gamma_2$

(γ_2 ist in einem Schritt aus γ_1 ableitbar)

falls:

- es gibt eine Produktion $\alpha \to \beta$ in P,
- und Wörter $\delta_1, \delta_2 \in (V \cup \Sigma)^*$ mit

  $\gamma_1 = \delta_1 \alpha \delta_2$ und $\gamma_2 = \delta_1 \beta \delta_2$.

Ein String γ_2 ist demnach aus γ_1 in einem Schritt ableitbar, wenn γ_1 die linke Seite α einer Produktion $\alpha \to \beta$ als Teilstring enthält, und γ_2 dann aus γ_1 entsteht, indem dieser Teilstring im Kontext durch die rechte Seite β ersetzt wird.

Ableitung

$\Rightarrow_G^*$ ist die reflexive, transitive Hülle von $\Rightarrow_G$

und ist induktiv definiert:

- $\alpha \Rightarrow_G^* \alpha$ für jedes α
- ist $\alpha \Rightarrow_G^* \beta$ und $\beta \Rightarrow_G \gamma$,
  dann ist $\alpha \Rightarrow_G^* \gamma$.

$\alpha \Rightarrow_G^* \beta$ gilt, wenn es eine Folge $\gamma_1, \ldots, \gamma_n$ gibt mit

- $\alpha = \gamma_1$ und $\beta = \gamma_n$
- $\gamma_i \Rightarrow_G \gamma_{i+1}$ für $1 \leq i < n$

Die reflexive und transitive Hülle kann man ganz algemein genauso für jede Relation definieren, was wir auch im Folgenden stillschweigend tun werden.

Erzeugte Sprache

Definition

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

$$L(G) := \left\{ w \in \Sigma^* ; S \Rightarrow_G^* w \right\}$$

Beachten Sie, dass die von G erzeugte Sprache $L(G)$ nur Wörter aus Σ^* enthält. Die Variablen sind also dementsprechend nur Hilfssymbole, die bei der Ableitung verwendet werden.

Beispiel

Sei eine Grammatik G gegeben durch:

- $\Sigma = \{a, b, c\}$ und $V = \{S, B, C\}$
- Startsymbol ist S
- Produktionen sind:

$$S \to aSBC \quad , \quad S \to aBC \quad , \quad CB \to BC$$
$$aB \to ab \quad , \quad bB \to bb \quad , \quad bC \to bc \quad , \quad cC \to cc$$

Eine Ableitung in G:

$$S \Rightarrow aSBC \Rightarrow aaSBCBC \Rightarrow aaaBCBCBC \Rightarrow aaaBBCCBC$$
$$\Rightarrow^* aaaBBBCCC \Rightarrow aaabBBCCC \Rightarrow aaabbBCCC$$
$$\Rightarrow aaabbbCCC \Rightarrow aaabbbcCC \Rightarrow^* aaabbbccc$$

Es gilt: $L(G) = \{ a^n b^n c^n ; n \in \mathbb{N} \setminus \{0\} \}$

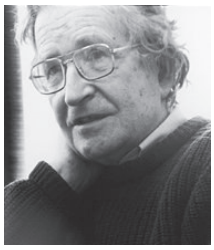
Es soll gezeigt werden, dass die Beispielgrammatik tatsächlich die genannte Sprache erzeugt.

Zunächst kann man jedes Wort der Form $a^n b^n c^n$ für ein $n \geq 1$ nach dem Muster der Beispielableitung erzeugen: zuerst wendet man $n-1$ mal die erste und dann die zweite Produktion an, um $a^n(BC)^n$ zu erzeugen. Danach verwendet man die Produktion $CB \to BC$, um die B und C zu sortieren, und leitet so $a^n B^n C^n$ ab. Schließlich verwendet man die Produktionen in der zweiten Zeile, um der Reihe nach die B durch b und die C durch c zu ersetzen.

Umgekehrt ist jedes Wort in $L(G)$ von der Form $a^n b^n c^n$. Zuerst beobachtet man, dass für jeden abeitbaren String γ gilt $|\gamma|_a = |\gamma|_B + |\gamma|_b = |\gamma|_C + |\gamma|_c$. Am Ende muss also $|w|_a = |w|_b = |w|_c$ sein.
Außerdem stehen alle a links von den anderen Symbolen. Dann beobachtet man, dass die B und C nur dann alle zu b und c werden können, wenn sie in der richtigen Reihenfolge da stehen: ist ein c am Ende vor einem B, kann dieses nicht mehr ersetzt werden, es ensteht also kein Wort aus Σ^*.

Die Chomsky-Hierarchie

Nach *Noam Chomsky* werden Grammatiken und die davon erzeugten Sprachen in 4 Typen unterteilt, nach Art der erlaubten Regeln.



Typ	Name	Bedingung an Regeln $\alpha \to \beta$
0	allgemein	-
1	monoton	$\lvert\alpha\rvert \leq \lvert\beta\rvert$
2	kontextfrei	Typ 1 und $\alpha \in V$
3	regulär	Typ 2 und $\beta \in \Sigma^* V \cup \Sigma^*$

Grammatiken und Sprachen vom Typ 1 werden auch *kontextsensitiv* genannt, diese Bezeichnung kommt von einer anderen, äquivalenten Beschreibung dieser Klasse.

Die Produktionen einer kontextfreien Grammatik sind alle von der Form $A \to \beta$, mit $A \in V$ und $\beta \in (V \cup \Sigma)^+$. Es kann also eine Variable durch einen String ersetzt werden, unabhängig davon, in welchem Kontext sie steht, daher die Bezeichnung *kontextfrei*.

Sonderregelung für das leere Wort

Nach Definition können Typ 1-Sprachen nicht das leere Wort enthalten.

Um auch Sprachen L mit $\epsilon \in L$ erzeugen zu können, wird die Definition erweitert:

Eine Grammatik ist auch vom Typ 1, wenn gilt:

- es gibt eine Produktion $S \rightarrow \epsilon$
- für alle anderen Produktionen $\alpha \rightarrow \beta$ gilt $|\alpha| \leq |\beta|$
- für alle Produktionen $\alpha \rightarrow \beta$ kommt S in β nicht vor.

Typ 3-Grammatiken und reguläre Sprachen

Satz

Die von Typ 3-Grammatiken erzeugten Sprachen sind genau die regulären Sprachen.

Dazu zeigen wir zwei Teile:

Lemma

Für jede Typ 3-Grammatik G gibt es einen NEA A_G mit $L(A_G) = L(G)$.

Lemma

Für jeden DEA A gibt es eine Typ 3-Grammatik G_A mit $L(G_A) = L(A)$.

NEA aus Typ 3-Grammatik

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

ObdA sind alle Produktionen von der Form:

$$A \to aB \quad , \quad A \to a \quad \text{oder} \quad S \to \epsilon$$

Definiere eine NEA A_G mit $L(A_G) = L(G)$ wie folgt:

- Zustandsmenge ist $V \cup \{X\}$
- Anfangszustand ist S
- Endzustände sind $\{X\}$, falls $\epsilon \notin L(G)$, oder $\{S, X\}$ falls $\epsilon \in L(G)$.
- für jede Produktion $A \to aB$ ist $B \in \delta(A, a)$
- für jede Produktion $A \to a$ ist $X \in \delta(A, a)$

Die oben angegebene Form gilt ohne Einschränkung der Allgemeinheit, da jede Typ 3-Grammatik leicht in eine transformiert werden kann, die der Bedingung genügt. Enthält sie nämich eine Produktion $A \to a_1 \dots a_k B$ mit $k > 1$, so führt man $k-1$ neue Variablen $H_1, \dots H_{k-1}$ ein, und ersetzt die Produktion durch

$$A \to a_1 H_1 \quad , \quad H_1 \to a_2 H_2 \quad , \quad \dots \quad , \quad H_{k-1} \to a_k B$$

Analog verfährt man für Produktionen $A \to w$ mit $|w| > 1$.

Zum Beweis, dass $L(A_G) = L(A)$ ist, sind die folgenden zwei Aussagen simultan durch Induktion nach der Länge $|w|$ zu beweisen: für alle $w \in \Sigma^*$ gilt

- $S \Rightarrow_G^* wB$ genau dann, wenn $B \in \hat{\delta}(S, w)$
- $S \Rightarrow_G^* w$ genau dann, wenn $X \in \hat{\delta}(S, w)$.

Kontextfreie Sprachen

Jede reguläre Sprache ist kontextfrei (Typ 2), aber es gibt kontextfreie Sprachen, die nicht regulär sind.

Z.B. ist die Sprache L_{kl} der korrekt geklammerten Klammerausdrücke kontextfrei.

Typ 2-Grammatik für L_{kl}:

- Variablen sind $\{S, K\}$, Startsymbol ist S.
- Produktionen sind:
  - $S \rightarrow \epsilon$
  - $S \rightarrow K$
  - $K \rightarrow <>$
  - $K \rightarrow <K>$
  - $K \rightarrow KK$

Offensichtlich ist jedes Wort, dass von der angegebenen Grammatik erzeugt wird, ein korrekt geklammerter Ausdruck.

Wir beweisen die Umkehrung wie folgt: Sei w ein korrekt geklammerter Ausdruck. Ist $w = \epsilon$, wird er mit der ersten Produktion erzeugt. Ist w nicht leer, so zeigen wir durch Induktion nach der Wortlänge, dass sich w aus K erzeugen lässt.

w muss mit einer öffnenden Klammer beginnen, und zu dieser gibt es eine matchende schließende Kammer. Also ist $w = <w_1>w_2$, wobei w_1 und w_2 korrekt geklammerte Ausdrücke sind, die kürzer sind als w.

Ist w_2 nicht leer, so wird zunächst mit der letzten Produktion KK abgeleitet, und nach Induktionshypothese ist w_2 aus dem zweiten K ableitbar. Bleibt also $<w_1>$ aus K abzuleiten. Ist $w_1 = \epsilon$, so wird dies mit der dritten Produktion erreicht. Sonst wird mit der vierten Produktion $<K>$ abgeleitet, und nach Induktionshypothese ist dannn aus dem K dann w_1 ableitbar.

Kontextfreie Grammatiken in der Praxis

In der Praxis werden kontextfreie Grammatiken in der sog. Backus-Naur-Form präsentiert.

Notation:	steht für:
$\mathtt{A} ::= \gamma_1 \mid \ldots \mid \gamma_k.$	$A \to \gamma_1 \,, \ldots \,, A \to \gamma_k$
$\mathtt{A} ::= \alpha[\beta]\gamma$	$A \to \alpha\gamma \,, A \to \alpha\beta\gamma$
$\mathtt{A} ::= \alpha\{\beta\}\gamma$	$A \to \alpha\gamma \,, A \to \alpha B\gamma,$ $B \to \beta B \,, B \to \beta$

Wörter aus Σ^* werden in Anführungszeichen gesetzt.

Grammatik für L_{kl} in Backus-Naur-Form:

```
S ::= "" | K.
K ::= {K} | "<>" | "<"K">".
```

Die Syntax von Sprachen, die in der Informatik verwendet werden, wie Programmier-, Anfrage-, Skript- und Markupsprachen, ist meist durch kontextfreie Grammatiken definiert. Dabei wird oft die sog. Backus-Naur-Form verwendet, die die oben angegeben allgemeineren Regeln erlaubt. Diese können jedoch durch kontextfreie Regeln wie definiert ersetzt werden.