# Kapitel IV Formale Sprachen und Grammatiken

$\Sigma = \{0, 1\}$

## 1. Begriffe und Notationen

Sei $\Sigma$ ein (endliches) Alphabet. Dann

$\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 11, 1100, \ldots\}$

**Definition 42**

1. ist $\Sigma^*$ das Monoid über $\Sigma$, d.h. die Menge aller endlichen Wörter über $\Sigma$;
2. ist $\Sigma^+$ die Menge aller nichtleeren endlichen Wörter über $\Sigma$;
3. bezeichnet $|w|$ für $w \in \Sigma^*$ die Länge von $w$;
4. ist $\Sigma^n$ für $n \in \mathbb{N}_0$ die Menge aller Wörter der Länge $n$ in $\Sigma^*$;
5. eine Teilmenge $L \subseteq \Sigma^*$ eine formale Sprache.

# Kapitel IV Formale Sprachen und Grammatiken

## 1. Begriffe und Notationen

Sei $\Sigma$ ein (endliches) Alphabet. Dann

### Definition 42

1. ist $\Sigma^*$ das Monoid über $\Sigma$, d.h. die Menge aller endlichen Wörter über $\Sigma$;
2. ist $\Sigma^+$ die Menge aller nichtleeren endlichen Wörter über $\Sigma$;
3. bezeichnet $|w|$ für $w \in \Sigma^*$ die Länge von $w$;
4. ist $\Sigma^n$ für $n \in \mathbb{N}_0$ die Menge aller Wörter der Länge $n$ in $\Sigma^*$;
5. eine Teilmenge $L \subseteq \Sigma^*$ eine formale Sprache.

# Kapitel IV Formale Sprachen und Grammatiken

$0001 \in \{0,1\}^*$

$|2001| = 4$

## 1. Begriffe und Notationen

Sei $\Sigma$ ein (endliches) Alphabet. Dann

**Definition 42**

1. ist $\Sigma^*$ das Monoid über $\Sigma$, d.h. die Menge aller endlichen Wörter über $\Sigma$;
2. ist $\Sigma^+$ die Menge aller nichtleeren endlichen Wörter über $\Sigma$;
3. bezeichnet $|w|$ für $w \in \Sigma^*$ die Länge von $w$;
4. ist $\Sigma^n$ für $n \in \mathbb{N}_0$ die Menge aller Wörter der Länge $n$ in $\Sigma^*$;
5. eine Teilmenge $L \subseteq \Sigma^*$ eine formale Sprache.

# Kapitel IV Formale Sprachen und Grammatiken

$\{0,1\}^3 =$

$\{000, 001, 010, \ldots, 111\}$

## 1. Begriffe und Notationen

Sei $\Sigma$ ein (endliches) Alphabet. Dann

**Definition 42**

1. ist $\Sigma^*$ das Monoid über $\Sigma$, d.h. die Menge aller endlichen Wörter über $\Sigma$; $|\Sigma|^n$
2. ist $\Sigma^+$ die Menge aller nichtleeren endlichen Wörter über $\Sigma$;
3. bezeichnet $|w|$ für $w \in \Sigma^*$ die Länge von $w$;
4. ist $\Sigma^n$ für $n \in \mathbb{N}_0$ die Menge aller Wörter der Länge $n$ in $\Sigma^*$;
5. eine Teilmenge $L \subseteq \Sigma^*$ eine formale Sprache.

# Kapitel IV Formale Sprachen und Grammatiken

## 1. Begriffe und Notationen

Sei $\Sigma$ ein (endliches) Alphabet. Dann

### Definition 42

1. ist $\Sigma^*$ das Monoid über $\Sigma$, d.h. die Menge aller endlichen Wörter über $\Sigma$;
2. ist $\Sigma^+$ die Menge aller nichtleeren endlichen Wörter über $\Sigma$;
3. bezeichnet $|w|$ für $w \in \Sigma^*$ die Länge von $w$;
4. ist $\Sigma^n$ für $n \in \mathbb{N}_0$ die Menge aller Wörter der Länge $n$ in $\Sigma^*$;
5. eine Teilmenge $L \subseteq \Sigma^*$ eine formale Sprache.

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

## 2. Sprachkonzept

- Eine (formale) Sprache $L$ besteht aus Wörtern $w$ über einem Alphabet $\Sigma$
- Eine formale Sprache $L$ ist also i.a. einfach eine Teilmenge von $\Sigma^*$:

$$L \subseteq \Sigma^*$$

- Wir betrachten insbesondere formale Sprachen, deren Wörter entlang gewisser Regeln erzeugt werden
- Diese Regeln werden als Grammatik $G$ für die Sprache bezeichnet
- Die Menge der so bildbaren Wörter heißt Wortschatz oder Sprachschatz der Grammatik

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.

- Eine Grammatik wird durch
    1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
    2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
    3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
    4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
    1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
    2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
    3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
    4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (\underline{V \cup \Sigma})^*$

$V = \{A, B\}$
$\Sigma = \{a, b\}$

$\varepsilon, A, aAbbB$

- Eine Grammatik wird durch
  1. $\Sigma$, eine endliche Menge von Terminalzeichen (Alphabet)
  2. $V$, eine endliche Menge von Nichtterminalzeichen (grammatische Begriffe)
  3. $P$, eine endliche Menge von Produktionen (Ableitungs-/Ersetzungsregeln, grammatische Regeln)
  4. $S$, ein spezielles Element aus $V$, das Axiom oder Startsymbol

  vollständig beschrieben.
- Die Vereinigungsmenge $V \cup \Sigma$ der Nichtterminale und Terminale heißt das Vokabular der Grammatik.
- $V$ und $\Sigma$ werden o.B.d.A. als disjunkt vorausgesetzt, d.h.

$$V \cap \Sigma = \emptyset$$

- die Produktionen $p \in P$ haben ganz allgemein die Gestalt $A ::= B$, mit $A, B \in (V \cup \Sigma)^*$

## Beispiel 43

Wir betrachten folgende Grammatik:

gefolgt von

$$
\begin{aligned}
\langle \text{Satz} \rangle \quad &::= \quad \langle \text{Subjekt} \rangle \langle \text{Prädikat} \rangle \langle \text{Objekt} \rangle \\
\langle \text{Subjekt} \rangle \quad &::= \quad \langle \text{Artikel} \rangle \langle \text{Attribut} \rangle \langle \text{Substantiv} \rangle \\
\langle \text{Artikel} \rangle \quad &::= \quad \epsilon \\
\langle \text{Artikel} \rangle \quad &::= \quad \text{der}|\text{die}|\text{das}|\text{ein}| \ldots \\
\langle \text{Attribut} \rangle \quad &::= \quad \epsilon | \langle \text{Adjektiv} \rangle | \langle \text{Adjektiv} \rangle \langle \text{Attribut} \rangle \\
\langle \text{Adjektiv} \rangle \quad &::= \quad \text{gross}|\text{klein}|\text{schön}| \ldots
\end{aligned}
$$

oder

Die vorletzte Ersetzungsregel ist rekursiv, die durch diese Grammatik definierte Sprache deshalb unendlich.
Zur Darstellungsform dieser Grammatik später mehr!

## Beispiel 43

Wir betrachten folgende Grammatik:

$$
\begin{aligned}
\langle\text{Satz}\rangle &::= \langle\text{Subjekt}\rangle\langle\text{Prädikat}\rangle\langle\text{Objekt}\rangle\\
\langle\text{Subjekt}\rangle &::= \langle\text{Artikel}\rangle\langle\text{Attribut}\rangle\langle\text{Substantiv}\rangle\\
\langle\text{Artikel}\rangle &::= \epsilon\\
\langle\text{Artikel}\rangle &::= \text{der}|\text{die}|\text{das}|\text{ein}|\ldots\\
\langle\text{Attribut}\rangle &::= \epsilon|\langle\text{Adjektiv}\rangle|\langle\text{Adjektiv}\rangle\langle\text{Attribut}\rangle\\
\langle\text{Adjektiv}\rangle &::= \text{gross}|\text{klein}|\text{schön}|\ldots
\end{aligned}
$$

Die vorletzte Ersetzungsregel ist rekursiv, die durch diese Grammatik definierte Sprache deshalb unendlich.

Zur Darstellungsform dieser Grammatik später mehr!

## Beispiel 43

Wir betrachten folgende Grammatik:

$$
\begin{aligned}
\langle\text{Satz}\rangle &::= \langle\text{Subjekt}\rangle\langle\text{Prädikat}\rangle\langle\text{Objekt}\rangle\\
\langle\text{Subjekt}\rangle &::= \langle\text{Artikel}\rangle\langle\text{Attribut}\rangle\langle\text{Substantiv}\rangle\\
\langle\text{Artikel}\rangle &::= \epsilon\\
\langle\text{Artikel}\rangle &::= \text{der}|\text{die}|\text{das}|\text{ein}|\ldots\\
\langle\text{Attribut}\rangle &::= \epsilon|\langle\text{Adjektiv}\rangle|\langle\text{Adjektiv}\rangle\langle\text{Attribut}\rangle\\
\langle\text{Adjektiv}\rangle &::= \text{gross}|\text{klein}|\text{schön}|\ldots
\end{aligned}
$$

Die vorletzte Ersetzungsregel ist rekursiv, die durch diese Grammatik definierte Sprache deshalb unendlich.
Zur Darstellungsform dieser Grammatik später mehr!

## Beispiel 44

- $L_1 = \{aa, aaaa, aaaaaa, \ldots\} = \{(aa)^n, n \in \mathbb{N}\}$
  $(\Sigma_1 = \{a\})$
- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$
  $(\Sigma_2 = \{a, b\})$
- $L_3 = \{ab, aabb, aaabbb, \ldots\} = \{a^n b^n, n \in \mathbb{N}\}$
  $(\Sigma_3 = \{a, b\})$
- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, m + n > 0\}$ $(\Sigma_4 = \{a, b\})$

## Beispiel 44

- $L_1 = \{aa, aaaa, aaaaaa, \ldots\} = \{(aa)^n, n \in \mathbb{N}\}$
  $(\Sigma_1 = \{a\})$
- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$
  $(\Sigma_2 = \underline{\{a, b\}})$
- $L_3 = \{ab, aabb, aaabbb, \ldots\} = \{a^n b^n, n \in \mathbb{N}\}$
  $(\Sigma_3 = \{a, b\})$
- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, m + n > 0\} \qquad (\Sigma_4 = \{a, b\})$

## Beispiel 44

- $L_1 = \{aa, aaaa, aaaaaa, \ldots\} = \{(aa)^n, n \in \mathbb{N}\}$
  $(\Sigma_1 = \{a\})$
- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$
  $(\Sigma_2 = \{a, b\})$
- $L_3 = \{ab, aabb, aaabbb, \ldots\} = \{a^n b^n, n \in \mathbb{N}\}$
  $(\Sigma_3 = \{a, b\})$
- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, m + n > 0\}$ $\quad (\Sigma_4 = \{a, b\})$

$\Sigma^*$ $\Sigma^*\Sigma^*$

## Beispiel 44

- $L_1 = \{aa, aaaa, aaaaaa, \ldots\} = \{(aa)^n, n \in \mathbb{N}\}$
  $(\Sigma_1 = \{a\})$
- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$
  $(\Sigma_2 = \{a, b\})$
- $L_3 = \{ab, aabb, aaabbb, \ldots\} = \{a^n b^n, n \in \mathbb{N}\}$
  $(\Sigma_3 = \{a, b\})$
- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, \underbrace{m + n > 0}\}$ $(\Sigma_4 = \{a, b\})$

$L_5 = \{a^m b^n \cup b^m a^n\}$

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \;\in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \rightarrow_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \rightarrow_G B$ (oder nur $A \rightarrow B$)

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \to_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \to_G B$ (oder nur $A \to B$)

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \to_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \to_G B$ (oder nur $A \to B$)

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \rightarrow_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \rightarrow_G B$ (oder nur $A \rightarrow B$)

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \ \in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \rightarrow_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \rightarrow_G B$ (oder nur $A \rightarrow B$)

## 2.1 Erzeugung und Sprachschatz

- Sei $G$ eine Grammatik mit Terminalalphabet $\Sigma$, Nichtterminalalphabet $V$, Produktionen $P$, $p = A ::= B \in P$ und seien $x, y \in (\Sigma \cup V)^*$
- $y$ heißt mittels $G$ aus $x$ in einem Schritt ableitbar, falls es $v, w \in (\Sigma \cup V)^*$ gibt, so dass

$$x = vAw \text{ und } y = vBw$$

- $y$ wird also aus $x$ erzeugt, indem man in $x$ ein Vorkommen der linken Seite von $p$ durch deren rechte Seite ersetzt
- Wir schreiben dafür i.Z. auch

$$x \rightarrow_G y$$

- Überhaupt schreiben wir dementsprechend auch für Produktionen $A ::= B$ oft $A \rightarrow_G B$ (oder nur $A \rightarrow B$)

- Mit $\rightarrow^*$ bezeichnen wir die reflexive, transitive Hülle von $\rightarrow$.
- Es gilt also $x \rightarrow^* y$ gdw es $x^{(0)}, x^{(1)}, \ldots, x^{(k)} \in (\Sigma \cup V)^*$ gibt, so dass $k \geq 0$, $x = x^{(0)}$ und $y = x^{(k)}$ und

$$x^{(i)} \rightarrow_G x^{(i+1)} \text{ für alle } i = 0, \ldots, k-1$$

- Falls $S$ das Axiom der Grammatik $G$ ist und $S \rightarrow_G x$ gilt, so nennen wir $x$ auch eine (mittels $G$ erzeugbare) Sprach- oder Satzform

- Mit $\to^*$ bezeichnen wir die reflexive, transitive Hülle von $\to$.
- Es gilt also $x \to^* y$ gdw es $x^{(0)}, x^{(1)}, \ldots, x^{(k)} \in (\Sigma \cup V)^*$ gibt, so dass $k \geq 0$, $x = x^{(0)}$ und $y = x^{(k)}$ und

$$x^{(i)} \to_G x^{(i+1)} \text{ für alle } i = 0, \ldots, k-1$$

- Falls $S$ das Axiom der Grammatik $G$ ist und $S \to_G x$ gilt, so nennen wir $x$ auch eine (mittels $G$ erzeugbare) Sprach- oder Satzform

- Mit $\to^*$ bezeichnen wir die reflexive, transitive Hülle von $\to$.
- Es gilt also $x \to^* y$ gdw es $x^{(0)}, x^{(1)}, \ldots, x^{(k)} \in (\Sigma \cup V)^*$ gibt, so dass $k \geq 0$, $x = x^{(0)}$ und $y = x^{(k)}$ und

$$x^{(i)} \to_G x^{(i+1)} \text{ für alle } i = 0, \ldots, k-1$$

- Falls $S$ das Axiom der Grammatik $G$ ist und $S \to_G x$ gilt, so nennen wir $x$ auch eine (mittels $G$ erzeugbare) Sprach- oder Satzform

- Mit $\rightarrow^*$ bezeichnen wir die reflexive, transitive Hülle von $\rightarrow$.
- Es gilt also $x \rightarrow^* y$ gdw es $x^{(0)}, x^{(1)}, \ldots, x^{(k)} \in (\Sigma \cup V)^*$ gibt, so dass $k \geq 0$, $x = x^{(0)}$ und $y = x^{(k)}$ und

$$x^{(i)} \rightarrow_G x^{(i+1)} \text{ für alle } i = 0, \ldots, k-1$$

- Falls $S$ das Axiom der Grammatik $G$ ist und $S \rightarrow_G x$ gilt, so nennen wir $x$ auch eine (mittels $G$ erzeugbare) Sprach- oder Satzform

$S \rightarrow aA$
$A \rightarrow bbB$
$B \rightarrow \varepsilon \mid A$

$S \rightarrow aA$
$abbB \rightarrow abb$
$abbA$
$abbbbB \rightarrow abbbb$

## 2.2 Darstellungsformen

### 2.2.1 Backus-Naur-Form

- Zur Beschreibung der Programmiersprache ALGOL 60 erstmals benutzt
- Nichtterminale werden durch Winkelklammern kenntlich gemacht, z.B. ¡Ausdruck¿, diese können entfallen, wenn weiterhin die Eindeutigkeit der Regeln gegeben ist
- Produktionsregeln mit identischer linker Seite können mit dem „—“ Symbol zusammengefasst werden:
  $A ::= W_1$ und $A ::= W_2$ wird so zu $A ::= W_1|W_2$
- Im Rahmen der Vorlesung werden häufig Kleinbuchstaben für Terminale und Großbuchstaben für Nichtterminale gewählt

## 2.2 Darstellungsformen

### 2.2.1 Backus-Naur-Form

- Zur Beschreibung der Programmiersprache ALGOL 60 erstmals benutzt
- Nichtterminale werden durch Winkelklammern kenntlich gemacht, z.B. ¡Ausdruck¿, diese können entfallen, wenn weiterhin die Eindeutigkeit der Regeln gegeben ist
- Produktionsregeln mit identischer linker Seite können mit dem „—“ Symbol zusammengefasst werden:
  $A ::= W_1$ und $A ::= W_2$ wird so zu $A ::= W_1|W_2$
- Im Rahmen der Vorlesung werden häufig Kleinbuchstaben für Terminale und Großbuchstaben für Nichtterminale gewählt

## 2.2 Darstellungsformen

### 2.2.1 Backus-Naur-Form

- Zur Beschreibung der Programmiersprache ALGOL 60 erstmals benutzt
- Nichtterminale werden durch Winkelklammern kenntlich gemacht, z.B. ¡Ausdruck¿, diese können entfallen, wenn weiterhin die Eindeutigkeit der Regeln gegeben ist
- Produktionsregeln mit identischer linker Seite können mit dem „—“ Symbol zusammengefasst werden:
  $A ::= W_1$ und $A ::= W_2$ wird so zu $A ::= W_1|W_2$
- Im Rahmen der Vorlesung werden häufig Kleinbuchstaben für Terminale und Großbuchstaben für Nichtterminale gewählt

## 2.2 Darstellungsformen

### 2.2.1 Backus-Naur-Form

- Zur Beschreibung der Programmiersprache ALGOL 60 erstmals benutzt
- Nichtterminale werden durch Winkelklammern kenntlich gemacht, z.B. ¡Ausdruck¿, diese können entfallen, wenn weiterhin die Eindeutigkeit der Regeln gegeben ist
- Produktionsregeln mit identischer linker Seite können mit dem „—“ Symbol zusammengefasst werden:
  $A ::= W_1$ und $A ::= W_2$ wird so zu $A ::= W_1|W_2$
- Im Rahmen der Vorlesung werden häufig Kleinbuchstaben für Terminale und Großbuchstaben für Nichtterminale gewählt

## 2.2 Darstellungsformen

### 2.2.1 Backus-Naur-Form

- Zur Beschreibung der Programmiersprache ALGOL 60 erstmals benutzt
- Nichtterminale werden durch Winkelklammern kenntlich gemacht, z.B. ¡Ausdruck¿, diese können entfallen, wenn weiterhin die Eindeutigkeit der Regeln gegeben ist
- Produktionsregeln mit identischer linker Seite können mit dem „—“ Symbol zusammengefasst werden:
  $A ::= W_1$ und $A ::= W_2$ wird so zu $A ::= W_1|W_2$
- Im Rahmen der Vorlesung werden häufig Kleinbuchstaben für Terminale und Großbuchstaben für Nichtterminale gewählt

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - — ( ) Klammer für Zusammenfassung
  - — [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - — { } Inhalt der Klammer beliebig oft mal (0,1,2,. . . )
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - ( ) Klammer für Zusammenfassung
  - [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - { } Inhalt der Klammer beliebig oft mal (0,1,2,. . . )
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - — ( ) Klammer für Zusammenfassung
  - — [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - — { } Inhalt der Klammer beliebig oft mal (0,1,2,. . . )
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - ( ) Klammer für Zusammenfassung
  - [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - { } Inhalt der Klammer beliebig oft mal (0,1,2,. . . )
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - ( ) Klammer für Zusammenfassung
  - [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - { } Inhalt der Klammer beliebig oft mal (0,1,2,. . . )
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

- Zur Vereinfachung von rekursiven Produktionen (das zu definierende Nichtterminal steht auch auf der rechten Seite), werden Metasymbole verwendet:
  - ( ) Klammer für Zusammenfassung
  - [ ] Inhalt der Klammer optional (0-mal oder 1-mal)
  - { } Inhalt der Klammer beliebig oft mal (0,1,2,...)
- Diese erweiterte Notation wird als ExtendedBNF oder EBNF bezeichnet

## Beispiel 45 (Grammatik für die Menge aller arithmetischen Ausdrücke in BNF)

$$
\begin{aligned}
<\text{Terminal}> \quad &::= \quad + \mid - \mid * \mid / \mid \uparrow \mid ( \mid ) \mid v \mid z \\
<\text{Ausdruck}> \quad &::= \quad <\text{Term}> \mid + <\text{Term}> \mid - <\text{Term}> \mid \\
& \qquad <\text{Ausdruck}> + <\text{Term}> \mid \\
& \qquad <\text{Ausdruck}> - <\text{Term}> \\
<\text{Term}> \quad &::= \quad <\text{Faktor}> \mid <\text{Term}> * <\text{Faktor}> \mid \\
& \qquad <\text{Term}> / <\text{Faktor}> \\
<\text{Faktor}> \quad &::= \quad <\text{Elementarausdruck}> \mid \\
& \qquad <\text{Faktor}> \uparrow <\text{Elementarausdruck}> \\
<\text{Elementarausdruck}> \quad &::= \quad z \mid v \mid ( <\text{Ausdruck}> )
\end{aligned}
$$

Einsatz der EBNF-Metasymbole vereinfacht die Produktionsregel für $<\text{Faktor}>$ zu

$$<\text{Faktor}> \quad ::= \quad \{<\text{Faktor}> \uparrow\} <\text{Elementarausdruck}>$$

## Beispiel 45 (Grammatik für die Menge aller arithmetischen Ausdrücke in BNF)

$$
\begin{array}{rcl}
< \text{Terminal} > & ::= & + \mid - \mid * \mid / \mid \uparrow \mid ( \mid ) \mid v \mid z \\
< \text{Ausdruck} > & ::= & < \text{Term} > \mid + < \text{Term} > \mid - < \text{Term} > \mid \\
 & & < \text{Ausdruck} > + < \text{Term} > \mid \\
 & & < \text{Ausdruck} > - < \text{Term} > \\
< \text{Term} > & ::= & < \text{Faktor} > \mid < \text{Term} > * < \text{Faktor} > \mid \\
 & & < \text{Term} > / < \text{Faktor} > \\
< \text{Faktor} > & ::= & < \text{Elementarausdruck} > \mid \\
 & & < \text{Faktor} > \uparrow < \text{Elementarausdruck} > \\
< \text{Elementarausdruck} > & ::= & z \mid v \mid ( < \text{Ausdruck} > )
\end{array}
$$

Einsatz der EBNF-Metasymbole vereinfacht die Produktionsregel für $< \text{Faktor} >$ zu

Faktor ↑ Faktor ↑ Faktor

$$
< \text{Faktor} > \; ::= \; \{< \text{Faktor} > \uparrow\} < \text{Elementarausdruck} >
$$

## 2.2.2 Syntaxdiagramme

- Produktionsregeln der BNF können direkt in ein Syntaxdiagramm überführt werden
- Terminalzeichen sind durch Kreise bzw. abgerundete Rechtecke, syntaktischen Variablen durch Rechtecke dargestellt
- Produktionsregeln mit Wiederholungen und Alternativen werden durch Verbindungen zwischen den Elementen dargestellt

## 2.2.2 Syntaxdiagramme

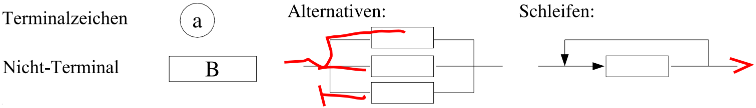
- Produktionsregeln der BNF können direkt in ein Syntaxdiagramm überführt werden
- Terminalzeichen sind durch Kreise bzw. abgerundete Rechtecke, syntaktischen Variablen durch Rechtecke dargestellt
- Produktionsregeln mit Wiederholungen und Alternativen werden durch Verbindungen zwischen den Elementen dargestellt

### 2.2.2 Syntaxdiagramme

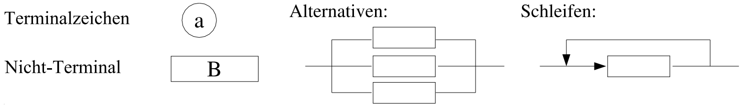
- Produktionsregeln der BNF können direkt in ein Syntaxdiagramm überführt werden
- Terminalzeichen sind durch Kreise bzw. abgerundete Rechtecke, syntaktischen Variablen durch Rechtecke dargestellt
- Produktionsregeln mit Wiederholungen und Alternativen werden durch Verbindungen zwischen den Elementen dargestellt

## 2.2.2 Syntaxdiagramme

- Produktionsregeln der BNF können direkt in ein Syntaxdiagramm überführt werden
- Terminalzeichen sind durch Kreise bzw. abgerundete Rechtecke, syntaktischen Variablen durch Rechtecke dargestellt
- Produktionsregeln mit Wiederholungen und Alternativen werden durch Verbindungen zwischen den Elementen dargestellt

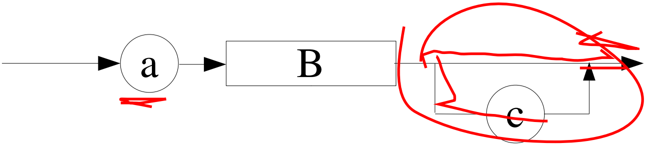



# Graphische Darstellung der BNF




Graphische Darstellung der BNF

## Beispiel 46




Syntaxdiagramm der Produktion $S ::= aB[c]$

## 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

### 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

# 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

## 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

1. einem Terminalalphabet $\Sigma$ (manchmal auch $T$), $|\Sigma| < \infty$
2. einem endlichen Vorrat von Nichtterminalzeichen (Variablen) $V$, $V \cap \Sigma = \emptyset$
3. einem Startsymbol (Axiom) $S \in V$
4. einer endliche Menge $P$ von Produktionen (Ableitungsregeln) der Form $l \to r$ (oder $l ::= r$), mit $l \in (V \cup \Sigma)^*$, $r \in (V \cup \Sigma)^*$

# 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

## 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

1. einem Terminalalphabet $\Sigma$ (manchmal auch $T$), $|\Sigma| < \infty$
2. einem endlichen Vorrat von Nichtterminalzeichen (Variablen) $V$, $V \cap \Sigma = \emptyset$
3. einem Startsymbol (Axiom) $S \in V$
4. einer endliche Menge $P$ von Produktionen (Ableitungsregeln) der Form $l \to r$ (oder $l ::= r$), mit $l \in (V \cup \Sigma)^*$, $r \in (V \cup \Sigma)^*$

# 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

## 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

1. einem Terminalalphabet $\Sigma$ (manchmal auch $T$), $|\Sigma| < \infty$
2. einem endlichen Vorrat von Nichtterminalzeichen (Variablen) $V$, $V \cap \Sigma = \emptyset$
3. einem Startsymbol (Axiom) $S \in V$
4. einer endliche Menge $P$ von Produktionen (Ableitungsregeln) der Form $l \to r$ (oder $l ::= r$), mit $l \in (V \cup \Sigma)^*$, $r \in (V \cup \Sigma)^*$

# 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

## 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

1. einem Terminalalphabet $\Sigma$ (manchmal auch $T$), $|\Sigma| < \infty$
2. einem endlichen Vorrat von Nichtterminalzeichen (Variablen) $V$, $V \cap \Sigma = \emptyset$
3. einem Startsymbol (Axiom) $S \in V$
4. einer endliche Menge $P$ von Produktionen (Ableitungsregeln) der Form $l \to r$ (oder $l ::= r$), mit $l \in (V \cup \Sigma)^*$, $r \in (V \cup \Sigma)^*$

## 3. Die Chomsky-Hierarchie

Diese Sprachenhierarchie ist nach Noam Chomsky [MIT, 1976] benannt.

### 3.1 Phrasenstrukturgrammatik, Chomsky-Grammatik

Grammatiken bestehen aus

1. einem Terminalalphabet $\Sigma$ (manchmal auch $T$), $|\Sigma| < \infty$
2. einem endlichen Vorrat von Nichtterminalzeichen (Variablen) $V$, $V \cap \Sigma = \emptyset$
3. einem Startsymbol (Axiom) $S \in V$
4. einer endliche Menge $P$ von Produktionen (Ableitungsregeln) der Form $l \to r$ (oder $l ::= r$), mit $l \in (V \cup \Sigma)^*$, $r \in (V \cup \Sigma)^*$

Eine Phrasenstrukturgrammatik (Grammatik) ist ein Quadrupel $G = (V, \Sigma, P, S)$.

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

**Definition 47**

Wir schreiben

genau dann wenn

"$\Longleftrightarrow$"

1. $z \rightarrow_G z'$ gdw
   $(\exists x, y \in (V \cup \Sigma)^*, l \rightarrow r \in P)[z = xly, z' = xry]$
2. $z \rightarrow_G^* z'$ gdw $z = z'$ oder
   $z \rightarrow_G z^{(1)} \rightarrow_G z^{(2)} \rightarrow_G \ldots \rightarrow_G z^{(k)} = z'$. Eine solche Folge von Ableitungsschritten heißt eine Ableitung für $z'$ von $z$ in $G$ (der Länge $k$).
3. Die von $G$ erzeugte Sprache ist

$$L(G) := \{z \in \Sigma^*;\ S \rightarrow_G^* z\}$$

Zur Vereinfachung der Notation schreiben wir gewöhnlich $\rightarrow$ und $\rightarrow^*$ statt $\rightarrow_G$ und $\rightarrow_G^*$

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

## Definition 47

Wir schreiben

1. $z \rightarrow_G z'$ gdw
   $(\exists x, y \in (V \cup \Sigma)^*, l \rightarrow r \in P)[z = xly, z' = xry]$
2. $z \rightarrow_G^* z'$ gdw $z = z'$ oder
   $z \rightarrow_G z^{(1)} \rightarrow_G z^{(2)} \rightarrow_G \ldots \rightarrow_G z^{(k)} = z'$. Eine solche Folge von Ableitungsschritten heißt eine Ableitung für $z'$ von $z$ in $G$ (der Länge $k$).
3. Die von $G$ erzeugte Sprache ist

$$L(G) := \{z \in \Sigma^*;\ S \rightarrow_G^* z\}$$

Zur Vereinfachung der Notation schreiben wir gewöhnlich $\rightarrow$ und $\rightarrow^*$ statt $\rightarrow_G$ und $\rightarrow_G^*$

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

## Definition 47

Wir schreiben

1. $z \rightarrow_G z'$ gdw
   $(\exists x, y \in (V \cup \Sigma)^*, l \rightarrow r \in P)[z = xly, z' = xry]$
2. $z \rightarrow_G^* z'$ gdw $z = z'$ oder
   $z \rightarrow_G z^{(1)} \rightarrow_G z^{(2)} \rightarrow_G \ldots \rightarrow_G z^{(k)} = z'$. Eine solche Folge von Ableitungsschritten heißt eine Ableitung für $z'$ von $z$ in $G$ (der Länge $k$).
3. Die von $G$ erzeugte Sprache ist

$$L(G) := \{z \in \Sigma^*;\ S \rightarrow_G^* z\}$$

Zur Vereinfachung der Notation schreiben wir gewöhnlich $\rightarrow$ und $\rightarrow^*$ statt $\rightarrow_G$ und $\rightarrow_G^*$

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

## Definition 47

Wir schreiben

1. $z \rightarrow_G z'$ gdw
$(\exists x, y \in (V \cup \Sigma)^*, l \rightarrow r \in P)[z = xly, z' = xry]$
2. $z \rightarrow_G^* z'$ gdw $z = z'$ oder
$z \rightarrow_G z^{(1)} \rightarrow_G z^{(2)} \rightarrow_G \ldots \rightarrow_G z^{(k)} = z'$. Eine solche Folge von Ableitungsschritten heißt eine Ableitung für $z'$ von $z$ in $G$ (der Länge $k$).
3. Die von $G$ erzeugte Sprache ist

$$L(G) := \{z \in \Sigma^*;\ S \rightarrow_G^* z\}$$

Zur Vereinfachung der Notation schreiben wir gewöhnlich $\rightarrow$ und $\rightarrow^*$ statt $\rightarrow_G$ und $\rightarrow_G^*$

**Vereinbarung:**

Wir bezeichnen Nichtterminale mit großen und Terminale mit kleinen Buchstaben!

### Beispiel 48

Wir erinnern uns:

$L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n; n \in \mathbb{N}\}$
$(\Sigma_2 = \{a, b\})$

Grammatik für $L_2$ mit folgenden Produktionen:

$$S \to ab, S \to abS$$

**Vereinbarung:**
Wir bezeichnen Nichtterminale mit großen und Terminale mit kleinen Buchstaben!

## Beispiel 48

Wir erinnern uns:

- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$
  $(\Sigma_2 = \{a, b\})$
- Grammatik für $L_2$ mit folgenden Produktionen:

$$S \to ab, S \to abS$$

**Vereinbarung:**
Wir bezeichnen Nichtterminale mit großen und Terminale mit kleinen Buchstaben!

## Beispiel 48

Wir erinnern uns:

- $L_2 = \{ab, abab, ababab, \ldots\} = \{(ab)^n, n \in \mathbb{N}\}$ $(\Sigma_2 = \{a, b\})$
- Grammatik für $L_2$ mit folgenden Produktionen:

$$S \to ab, S \to abS$$

$S \to ab \mid abS$

$abS$

## Beispiel 48 (Forts.)

- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, m + n > 0\} \qquad (\Sigma_4 = \{a, b\})$
- Grammatik für $L_4$ mit folgenden Produktionen:

$$S \to A,\ S \to B,\ S \to AB,$$
$$A \to a,\ A \to aA,$$
$$B \to b,\ B \to bB$$

$S \to a \mid b \mid aS \mid Sb$

## Beispiel 48 (Forts.)

- $L_4 = \{a, b, aa, ab, bb, aaa, aab, abb, bbb \ldots\}$
  $= \{a^m b^n, m, n \in \mathbb{N}_0, m + n > 0\} \qquad (\Sigma_4 = \{a, b\})$
- Grammatik für $L_4$ mit folgenden Produktionen:

$$S \to A,\ S \to B,\ S \to AB,$$
$$A \to a,\ A \to aA,$$
$$B \to b,\ B \to bB$$

$\to a, aa, aaa, \ldots$

bbb $\quad S \to B$
$\quad bB$
$\quad bbB$
$\quad bbb$

$\boxed{\begin{array}{l} A \to B \\ A \to C \\ A \to B \mid C \end{array}}$

## 3.2 Die Chomsky-Hierarchie

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

1. Jede Phrasenstrukturgrammatik (Chomsky-Grammatik) ist (zunächst) automatisch vom Typ 0.
2. Eine Chomsky-Grammatik heißt (längen-)monoton, falls für alle Regeln

$$\alpha \to \beta \in P \text{ mit } \alpha \neq S$$

gilt:

$$|\alpha| \leq |\beta| ,$$

und, falls $S \to \epsilon \in P$, dann das Axiom $S$ auf keiner rechten Seite vorkommt.

## 3.2 Die Chomsky-Hierarchie

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

1. Jede Phrasenstrukturgrammatik (Chomsky-Grammatik) ist (zunächst) automatisch vom Typ 0.
2. Eine Chomsky-Grammatik heißt (längen-)monoton, falls für alle Regeln

$$\alpha \to \beta \in P \text{ mit } \alpha \neq S \qquad aA$$

gilt:

$$|\alpha| \leq |\beta| ,$$

und, falls $S \to \epsilon \in P$, dann das Axiom $S$ auf keiner rechten Seite vorkommt.

$aAB \Rightarrow bB$

3. Eine Chomsky-Grammatik ist vom Typ 1 (auch: kontextsensitiv), falls sie monoton ist und für alle Regeln $\alpha \to \beta$ in $P$ mit $\alpha \neq S$ gilt:

$$\alpha = \alpha' A \alpha'' \text{ und } \beta = \alpha' \beta' \alpha''$$

für geeignete $A \in V$, $\alpha', \alpha'' \in (V \cup \Sigma)^*$ und $\beta' \in (V \cup \Sigma)^+$.

4. Eine Chomsky-Grammatik ist vom Typ 2 (auch: kontextfrei), falls sie monoton ist und für alle Regeln $\alpha \to \beta \in P$ gilt:

$$\alpha \in V\ .$$

3. Eine Chomsky-Grammatik ist vom Typ 1 (auch: kontextsensitiv), falls sie monoton ist und für alle Regeln $\alpha \to \beta$ in $P$ mit $\alpha \neq S$ gilt:

$$\alpha = \alpha' A \alpha'' \text{ und } \beta = \alpha' \beta' \alpha''$$

für geeignete $A \in V$, $\alpha', \alpha'' \in (V \cup \Sigma)^*$ und $\beta' \in (V \cup \Sigma)^+$.

4. Eine Chomsky-Grammatik ist vom Typ 2 (auch: kontextfrei), falls sie monoton ist und für alle Regeln $\alpha \to \beta \in P$ gilt:

$$\alpha \in V \ .$$

3. Eine Chomsky-Grammatik ist vom Typ 1 (auch: kontextsensitiv), falls sie monoton ist und für alle Regeln $\alpha \to \beta$ in $P$ mit $\alpha \neq S$ gilt:

$$\alpha = \alpha' A \alpha'' \text{ und } \beta = \alpha' \beta' \alpha''$$

für geeignete $A \in V$, $\alpha', \alpha'' \in (V \cup \Sigma)^*$ und $\beta' \in (V \cup \Sigma)^+$.

4. Eine Chomsky-Grammatik ist vom Typ 2 (auch: kontextfrei), falls sie monoton ist und für alle Regeln $\alpha \to \beta \in P$ gilt:

$$\alpha \in V\ .$$

Bemerkung: Manchmal wird "kontextfrei" auch ohne die Monotonie-Bedingung definiert; streng monoton schließt dann die Monotonie mit ein, so dass $\epsilon$ nicht als rechte Seite vorkommen kann.

5. Eine Chomsky-Grammatik ist vom Typ 3 (auch: regulär, rechtslinear), falls sie monoton ist und für alle Regeln $\alpha \to \beta$ in $P$ mit $\beta \neq \epsilon$ gilt:

$$\alpha \in V \text{ und } \beta \in \Sigma^+ \cup \Sigma^* V \ .$$

5. Eine Chomsky-Grammatik ist vom Typ 3 (auch: regulär, rechtslinear), falls sie monoton ist und für alle Regeln $\alpha \rightarrow \beta$ in $P$ mit $\beta \neq \epsilon$ gilt:

$$\alpha \in V \text{ und } \beta \in \Sigma^{+} \cup \Sigma^{*}V\,.$$

Auch hier gilt die entsprechende Bemerkung zur Monotonie-Bedingung.

## Beispiel 49

- Die folgende Grammatik ist regulär:

$$S \to \epsilon,\ S \to A,$$
$$A \to aa,\ A \to aaA$$

- Eine Produktion

$$A \to Bcde$$

heißt linkslinear.

- Eine Produktion

$$A \to abcDef$$

heißt linear.

## Beispiel 49

- Die folgende Grammatik ist regulär:

$$S \to \epsilon,\ S \to A,$$
$$A \to aa,\ A \to aaA$$

- Eine Produktion

$$A \to Bcde$$

heißt linkslinear.

- Eine Produktion

$$A \to abcDef$$

heißt linear.

## Beispiel 49

- Die folgende Grammatik ist regulär:
$$S \rightarrow \epsilon,\ S \rightarrow A,$$
$$A \rightarrow aa,\ A \rightarrow aaA$$
- Eine Produktion
$$A \rightarrow Bcde$$
heißt linkslinear.
- Eine Produktion
$$A \rightarrow abcDef$$
heißt linear.

## Definition 50

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

In der Chomsky-Hierarchie bilden also die Typ-3- oder regulären Sprachen die kleinste, unterste Stufe, darüber kommen die kontextfreien, dann die kontextsensitiven Sprachen. Oberhalb der Typ-1-Sprachen kommen die Typ-0-Sprachen, die auch rekursiv aufzählbar oder semientscheidbar genannt werden. Darüber (und nicht mehr Teil der Chomsky-Hierarchie) findet sich die Klasse aller formalen Sprachen.

In Typ-3-Grammatiken müssen entweder alle Produktionen rechtslinear oder alle linkslinear sein.
Überlegen Sie sich eine lineare Grammatik, deren Sprache nicht regulär ist! (Beweismethode später!)

## Definition 50

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

In der Chomsky-Hierarchie bilden also die Typ-3- oder regulären Sprachen die kleinste, unterste Stufe, darüber kommen die kontextfreien, dann die kontextsensitiven Sprachen. Oberhalb der Typ-1-Sprachen kommen die Typ-0-Sprachen, die auch rekursiv aufzählbar oder semientscheidbar genannt werden. Darüber (und nicht mehr Teil der Chomsky-Hierarchie) findet sich die Klasse aller formalen Sprachen.

In Typ-3-Grammatiken müssen entweder alle Produktionen rechtslinear oder alle linkslinear sein.

Überlegen Sie sich eine lineare Grammatik, deren Sprache nicht regulär ist! (Beweismethode später!)

## Definition 50

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

In der Chomsky-Hierarchie bilden also die Typ-3- oder regulären Sprachen die kleinste, unterste Stufe, darüber kommen die kontextfreien, dann die kontextsensitiven Sprachen. Oberhalb der Typ-1-Sprachen kommen die Typ-0-Sprachen, die auch rekursiv aufzählbar oder semientscheidbar genannt werden. Darüber (und nicht mehr Teil der Chomsky-Hierarchie) findet sich die Klasse aller formalen Sprachen.

In Typ-3-Grammatiken müssen entweder alle Produktionen rechtslinear oder alle linkslinear sein.

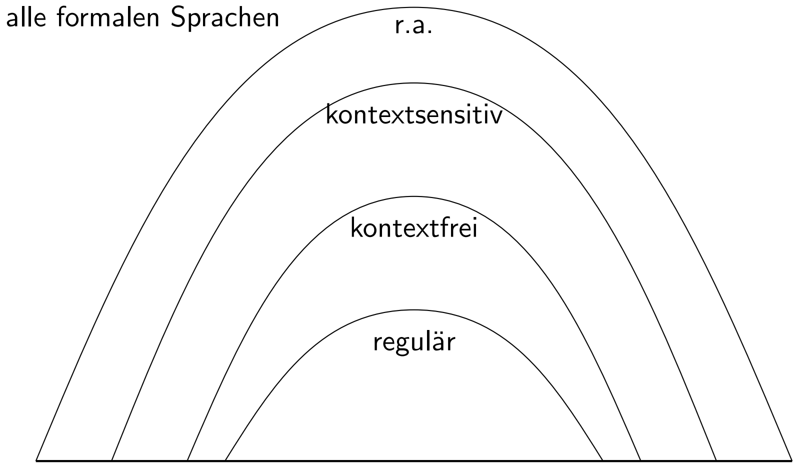
Überlegen Sie sich eine lineare Grammatik, deren Sprache nicht regulär ist! (Beweismethode später!)

## Definition 50

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

In der Chomsky-Hierarchie bilden also die Typ-3- oder regulären Sprachen die kleinste, unterste Stufe, darüber kommen die kontextfreien, dann die kontextsensitiven Sprachen. Oberhalb der Typ-1-Sprachen kommen die Typ-0-Sprachen, die auch rekursiv aufzählbar oder semientscheidbar genannt werden. Darüber (und nicht mehr Teil der Chomsky-Hierarchie) findet sich die Klasse aller formalen Sprachen.

In Typ-3-Grammatiken müssen entweder alle Produktionen rechtslinear oder alle linkslinear sein.
Überlegen Sie sich eine lineare Grammatik, deren Sprache nicht regulär ist! (Beweismethode später!)


alle formalen Sprachen
r.a.
kontextsensitiv
kontextfrei
regulär