SS 2006

# Einführung in die Informatik IV

Ernst W. Mayr

Fakultät für Informatik
TU München

http://www14.in.tum.de/lehre/2006SS/info4/

Sommersemester 2006

## 2.2 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. Eine Chomsky-Grammatik ist vom Typ 1 (auch: kontextsensitiv), falls sie monoton ist und für alle Regeln $\alpha \rightarrow \beta$ 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 $\alpha \rightarrow \beta \in P$ gilt:

$$\alpha \in V \; .$$

Bemerkung: Manchmal wird "kontextfrei" auch ohne die Monotonie-Bedingung definiert; streng monoton schließt dann die Monotonie mit ein.

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:

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

Auch hier gilt die entsprechende Bemerkung zur Monotonie-Bedingung.

## Beispiel 6

- 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

$$A \rightarrow abcDef$$

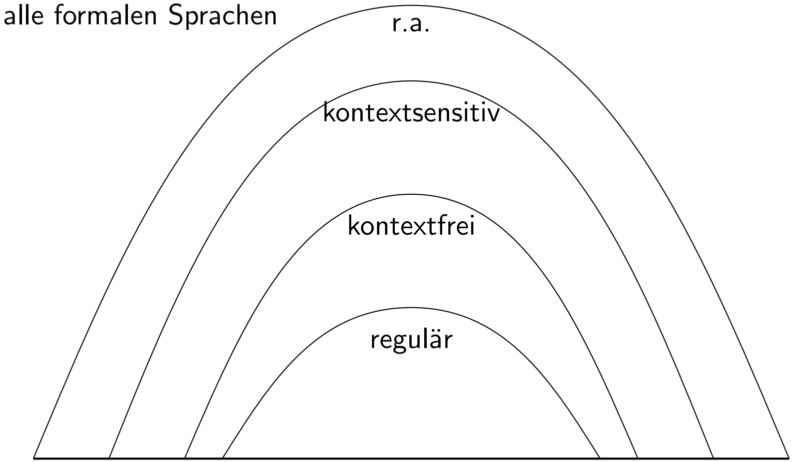
  heißt linear.

## Definition 7

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

## Lemma 8

*Sei $G = (V, \Sigma, P, S)$ eine Chomsky-Grammatik, so dass alle Produktionen $\alpha \to \beta$ die Bedingung $\alpha \in V$ erfüllen. Dann ist $L(G)$ kontextfrei.*

## Beweis:

## Definition 9

Ein $A \in V$ mit $A \to^* \epsilon$ heißt nullierbar.

## Bestimme alle nullierbaren $A \in V$:

$N := \{A \in V;\ (A \to \epsilon) \in P\}$
$N' := \emptyset$
**while** $N \neq N'$ **do**
  $N' := N$
  $N := N' \cup \{A \in V;$
    $(\exists (A \to \beta) \in P)[\beta \in N'^*]\}$
**od**

Wie man leicht durch Induktion sieht, enthält $N$ zum Schluss genau alle nullierbaren $A \in V$.

Sei nun $G$ eine Grammatik, so dass alle linken Seiten $\in V$, aber die Monotoniebedingung nicht unbedingt erfüllt ist.
Modifiziere $G$ zu $G'$ mit Regelmenge $P'$ wie folgt:

1. für jedes $(A \to x_1 x_2 \cdots x_n) \in P$, $n \geq 1$, füge zu $P'$ alle Regeln $A \to y_1 y_2 \cdots y_n$ hinzu, die dadurch entstehen, dass für nicht-nullierbare $x_i$ $y_i := x_i$ und für nullierbare $x_i$ die beiden Möglichkeiten $y_i := x_i$ und $y_i := \epsilon$ eingesetzt werden, ohne dass die ganze rechte Seite $= \epsilon$ wird.
2. falls $S$ nullierbar ist, sei $T$ ein neues Nichtterminal; füge zu $P'$ die Regeln $S \to \epsilon$ und $S \to T$ hinzu, ersetze $S$ in allen rechten Seiten durch $T$ und füge für jede Regel $(S \to x) \in P$, $|x| > 0$, die Regel $T \to x$ zu $P'$ hinzu.

## Lemma 10

$G' = (V \cup T, \Sigma, P', S)$ *ist kontextfrei, und es gilt*

$$L(G') = L(G)\ .$$

## Beweis:

Klar! $\square$

Auch für reguläre Grammatiken gilt ein entsprechender Satz über die “Entfernbarkeit” nullierbarer Nichtterminale:

## Lemma 11

*Sei $G = (V, \Sigma, P, S)$ eine Chomsky-Grammatik, so dass für alle Regeln $\alpha \to \beta \in P$ gilt:*

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

*Dann ist $L(G)$ regulär.*

## Beweis:

Übungsaufgabe! $\square$

## Beispiel 12

Typ 3: $L = \{a^n;\ n \in \mathbb{N}\}$, Grammatik:
$$\begin{aligned} S &\rightarrow a,\\ S &\rightarrow aS \end{aligned}$$

Typ 2: $L = \{a^n b^n;\ n \in \mathbb{N}_0\}$, Grammatik:
$$\begin{aligned} S &\rightarrow \epsilon,\\ S &\rightarrow T,\\ T &\rightarrow ab,\\ T &\rightarrow aTb \end{aligned}$$

Wir benötigen beim Scannen *einen* Zähler.

## Beispiel 12 (Forts.)

Typ 1: $L = \{a^n b^n c^n;\ n \in \mathbb{N}\}$, Grammatik:

$$\begin{aligned} S &\to aSXY, \\ S &\to abY, \\ YX &\to XY, \\ bX &\to bb, \\ bY &\to bc, \\ cY &\to cc \end{aligned}$$

Wir benötigen beim Scannen *mindestens zwei* Zähler.

**Bemerkung:** Diese Grammatik entspricht *nicht* unserer Definition von Typ 1. Wir zeigen als Hausaufgabe eine geeignete erweiterte Definition!

Die Backus-Naur-Form (BNF) ist ein Formalismus zur kompakten Darstellung von Typ-2-Grammatiken.

- Statt

$$\begin{aligned} A &\rightarrow \beta_1 \\ A &\rightarrow \beta_2 \\ &\vdots \\ A &\rightarrow \beta_n \end{aligned}$$

  schreibt man

$$A \rightarrow \beta_1 | \beta_2 | \ldots | \beta_n \ .$$

Die Backus-Naur-Form (BNF) ist ein Formalismus zur kompakten Darstellung von Typ-2-Grammatiken.

- Statt

$$\begin{aligned} A &\rightarrow \alpha\gamma \\ A &\rightarrow \alpha\beta\gamma \end{aligned}$$

schreibt man

$$A \rightarrow \alpha[\beta]\gamma \; .$$

(D.h., das Wort $\beta$ kann, muss aber nicht, zwischen $\alpha$ und $\gamma$ eingefügt werden.)

Die Backus-Naur-Form (BNF) ist ein Formalismus zur kompakten Darstellung von Typ-2-Grammatiken.

- Statt

$$\begin{array}{rcl} A & \rightarrow & \alpha\gamma \\ A & \rightarrow & \alpha B\gamma \\ B & \rightarrow & \beta \\ B & \rightarrow & \beta B \end{array}$$

  schreibt man

$$A \rightarrow \alpha\{\beta\}\gamma\ .$$

  (D.h., das Wort $\beta$ kann beliebig oft (auch Null mal) zwischen $\alpha$ und $\gamma$ eingefügt werden.)

## Beispiel 13

$$\begin{array}{rcl}
\langle\text{Satz}\rangle & \rightarrow & \langle\text{Subjekt}\rangle\langle\text{Prädikat}\rangle\langle\text{Objekt}\rangle \\
\langle\text{Subjekt}\rangle & \rightarrow & \langle\text{Artikel}\rangle\langle\text{Attribut}\rangle\langle\text{Substantiv}\rangle \\
\langle\text{Prädikat}\rangle & \rightarrow & \text{ist}|\text{hat}|\ldots \\
\langle\text{Artikel}\rangle & \rightarrow & \epsilon|\text{der}|\text{die}|\text{das}|\text{ein}|\ldots \\
\langle\text{Attribut}\rangle & \rightarrow & \{\langle\text{Adjektiv}\rangle\} \\
\langle\text{Adjektiv}\rangle & \rightarrow & \text{gross}|\text{klein}|\text{schön}|\ldots
\end{array}$$

## 2.3 Das Wortproblem

### Beispiel 14 (Arithmetische Ausdrücke)

$$
\begin{array}{lcl}
<\text{exp}> & \rightarrow & <\text{term}> \\
<\text{exp}> & \rightarrow & <\text{exp}> + <\text{term}> \\
<\text{term}> & \rightarrow & (<\text{exp}>) \\
<\text{term}> & \rightarrow & <\text{term}> \times <\text{term}> \\
<\text{term}> & \rightarrow & a \mid b \mid \ldots \mid z
\end{array}
$$

Aufgabe eines Parsers ist nun, zu prüfen, ob eine gegebene Zeichenreihe einen gültigen arithmetischen Ausdruck darstellt und, falls ja, ihn in seine Bestandteile zu zerlegen.

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

## Definition 15

1. **Wortproblem:** Gegeben ein Wort $w \in \Sigma^*$, stelle fest, ob

$$w \in L(G)\ ?$$

2. **Ableitungsproblem:** Gegeben ein Wort $w \in L(G)$, gib eine Ableitung $S \rightarrow_G^* w$ an, d.h., eine Folge

$$S = w^{(0)} \rightarrow_G w^{(1)} \rightarrow_G \cdots \rightarrow_G w^{(n)} = w$$

   mit $w^{(i)} \in (\Sigma \cup V)^*$ für $i = 1, \ldots, n$.

3. **uniformes Wortproblem:** Wortproblem, bei dem jede Probleminstanz sowohl die Grammatik $G$ wie auch die zu testende Zeichenreihe $w$ enthält. Ist $G$ dagegen global festgelegt, spricht man von einem nicht-uniformen Wortproblem.