Inhaltsangabe

Dresden

§ 6. Die Chomsky-Hierarchie

Grammatiken dienen dazu, Wörter zu erzeugen.

Man hat dazu Regeln, die es erlauben, ein Wort durch ein anderes Wort zu ersetzen (aus ihm abzuleiten).

Die erzeugte Sprache ist die Menge der Wörter, die ausgehend von einem Startsymbol erzeugt werden können durch wiederholtes Ersetzen.

Beispiel 6.1

Regeln:	$S \longrightarrow aSb$	(1)
	$S \longrightarrow \varepsilon$	(2)
Startsymbol:	S	

$$S \xrightarrow{1} aSb \xrightarrow{1} aaSbb \xrightarrow{1} aaaSbbb \xrightarrow{2} aaabbb$$

Das Symbol S ist hier ein Hilfssymbol (nichtterminales Symbol)

und man ist nur an den erzeugten Wörtern interessiert, die das Hilfssymbol nicht enthalten (Terminalwörter).

Man sieht leicht, daß dies hier alle Wörter $a^n b^n$ mit $n \geq 0$ sind.

Definition 6.2 (Grammatik)

Eine Grammatik ist von der Form $G = (N, \Sigma, P, S)$, wobei

- N und Σ endliche, disjunkte Alphabete sind (N: Nichtterminalsymbole, Σ: Terminalsymbole),
- $S \in N$ das Startsymbol ist,
- $P \subseteq (N \cup \Sigma)^+ \times (N \cup \Sigma)^*$ eine endliche Menge von Ersetzungsregeln (Produktionen) ist.

Produktionen $(u, v) \in P$ schreibt man gewöhnlich als $u \longrightarrow v$.

Beispiel 6.3

$G = (N, \Sigma, P, S)$ mit $N = \{S, B\}$, $\Sigma = \{a, b, c\}$ und

$$P = \{S \longrightarrow aSBc,\ S \longrightarrow abc,\ cB \longrightarrow Bc,\ bB \longrightarrow bb\}$$

Wir schreiben Elemente von N mit Großbuchstaben und Elemente von Σ mit Kleinbuchstaben.

Definition 6.4 (Ableitbarkeit, erzeugte Sprache)

Es sei $G = (N, \Sigma, P, S)$ eine Grammatik und $x, y \in (N \cup \Sigma)^*$.

1. y aus x direkt ableitbar:

   $x \vdash_G y$ gdw $\exists x_1, x_2 \in (N \cup \Sigma)^* : \exists u \longrightarrow v \in P : x = x_1 u x_2 \wedge y = x_1 v x_2$

2. y aus x in n Schritten ableitbar:

   $x \vdash_G^n y$ gdw $\exists x_0, x_1, \ldots, x_n \in (N \cup \Sigma)^* : x_0 = x \wedge x_n = y \wedge x_i \vdash_G x_{i+1}$ für $0 \leq i < n$

3. y aus x ableitbar:

   $x \vdash_G^* y$ gdw $\exists n \geq 0 : x \vdash_G^n y$

4. Die durch G erzeugte Sprache:

   $L(G) := \{w \in \Sigma^* \mid S \vdash_G^* w\}$.

Dresden

Beispiel 6.3 (Fortsetzung)

$G = (N, \Sigma, P, S)$ mit $N = \{S, B\}, \Sigma = \{a, b, c\}$ und

$$P = \{S \longrightarrow aSBc,\ S \longrightarrow abc,\ cB \longrightarrow Bc,\ bB \longrightarrow bb\}$$

$S \quad \vdash_G \quad abc,$ d.h. $abc \in L(G)$

$$S \quad \vdash_G \quad aSBc \vdash_G aaSBcBc \vdash_G aaabcBcBc \vdash_G aaabBccBc$$

$$\vdash_G^2 \quad aaabBBccc \vdash_G^2 aaabbbccc$$

Es gilt: $L(G) = \{a^n b^n c^n \mid n \geq 1\}$

Beispiel 6.5

$G = (N, \Sigma, P, S)$ mit $N = \{S, B\}, \Sigma = \{a, b\}$ und

$$
\begin{array}{rcrcl}
P & = & \{S & \longrightarrow & aS, \\
 & & S & \longrightarrow & bS, \\
 & & S & \longrightarrow & abB, \\
 & & B & \longrightarrow & aB, \\
 & & B & \longrightarrow & bB, \\
 & & B & \longrightarrow & \varepsilon \ \}
\end{array}
$$

$$L(G) = \Sigma^* \cdot \{a\} \cdot \{b\} \cdot \Sigma^*$$

Dresden

Definition 6.6 (Chomsky-Hierarchie)

Es sei $G = (N, \Sigma, P, S)$ eine Grammatik.

- Jede Grammatik G heißt Grammatik vom Typ 0.
- G heißt Grammatik vom Typ 1 (kontextsensitiv), falls jede Produktion von G die Form
  - $u_1 A u_2 \longrightarrow u_1 w u_2$ mit $A \in N$, $u_1, u_2, w \in (\Sigma \cup N)^*$ und $|w| \geq 1$

    oder
  - $S \longrightarrow \varepsilon$ hat.

  Ist $S \longrightarrow \varepsilon \in P$, so kommt S nicht auf der rechten Seite einer Produktion vor.
- G heißt Grammatik vom Typ 2 (kontextfrei), falls jede Regel von G die Form $A \longrightarrow w$ hat mit $A \in N$, $w \in (\Sigma \cup N)^*$.
- G heißt Grammatik vom Typ 3 (rechtslinear), falls jede Regel von G die Form $A \longrightarrow uB$ oder $A \longrightarrow u$ hat mit $A \in N$, $u \in \Sigma^*$.

Warum heißt Typ 2 kontextfrei und Typ 1 kontextsensitiv?

kontextfrei:

linke Seite nur Nichtterminalsymbol A, das daher stets unabhängig vom Kontext im Wort ersetzt werden kann.

kontextsensitiv:

Produktionen der Form $u_1 A u_2 \longrightarrow u_1 w u_2$. Hier ist die Ersetzung von A durch w abhängig davon, daß der richtige Kontext (u_1 links und u_2 rechts) vorhanden ist.

Die Einschränkung $|w| \geq 1$ sorgt dafür, daß die Produktionen die Wörter nicht verkürzen.

Ausnahme $S \longrightarrow \varepsilon$, die aber nur zur Erzeugung von ε dient.

Beispiel 6.1

$$\begin{array}{lcl} S & \longrightarrow & aSb \\ S & \longrightarrow & \varepsilon \end{array}$$

Typ 2 (kontextfrei)

Beispiel 6.3

$$\begin{array}{rcl} S & \longrightarrow & aSBc, \\ S & \longrightarrow & abc, \\ cB & \longrightarrow & Bc, \\ bB & \longrightarrow & bb \end{array}$$

Typ 0

„fast“ Typ 1 (kontextsensitiv)

Beispiel 6.5

$$\begin{array}{lcl} S & \longrightarrow & aS, \\ S & \longrightarrow & bS, \\ S & \longrightarrow & abB, \\ B & \longrightarrow & aB, \\ B & \longrightarrow & bB, \\ B & \longrightarrow & \varepsilon \end{array}$$

Typ 3 (rechtslinear)

Definition 6.7 (Sprachklassen)

Für $i = 0, 1, 2, 3$ ist die Klasse der Typ-i-Sprachen definiert als

$$\mathcal{L}_i := \{L(G) \mid G \text{ ist Grammatik vom Typ i}\}$$

Wir werden sehen:

$$\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$$

Nach Definition der Grammatiktypen gilt offenbar:

$$\mathcal{L}_3 \subseteq \mathcal{L}_2 \;\text{ und }\; \mathcal{L}_1 \subseteq \mathcal{L}_0$$

Inhaltsangabe

§ 7. Rechtslineare Grammatiken und reguläre Sprachen

Satz 7.1 (Typ 3 = regulär)

Die Typ-3-Sprachen sind genau die regulären/erkennbaren Sprachen, d.h.

$$\mathcal{L}_3 = \{L \mid L \text{ ist regulär}\}.$$

Beweis:

1. Jede Typ-3-Sprache ist erkennbar.

2. Jede erkennbare Sprache ist eine Typ-3-Sprache.

1. Jede Typ-3-Sprache ist erkennbar.

Es sei $L \in \mathcal{L}_3$, d.h. $L = L(G)$ für eine Typ-3-Grammatik $G = (N, \Sigma, P, S)$.

Wir konstruieren einen NEA mit Worttransitionen für L:

$$\mathcal{A} = (N \cup \{\Omega\}, \Sigma, S, \Delta, \{\Omega\}),$$

wobei $\Omega \notin N$ Endzustand ist und

$$\begin{array}{rcl} \Delta & = & \{(A, w, B) \mid A \longrightarrow wB \in P\} \cup \\ & & \{(A, w, \Omega) \mid A \longrightarrow w \in P\}. \end{array}$$

Es gilt $w \in L(G)$

gdw es gibt eine Ableitung

$S = B_0 \vdash_G w_1 B_1 \vdash_G \ldots \vdash_G w_1 \ldots w_{n-1} B_{n-1} \vdash_G w_1 \ldots w_{n-1} w_n = w$

für Produktionen $B_{i-1} \longrightarrow w_i B_i$ $(i \leq n-1)$ und $B_{n-1} \longrightarrow w_n$ von G

gdw es gibt einen Pfad

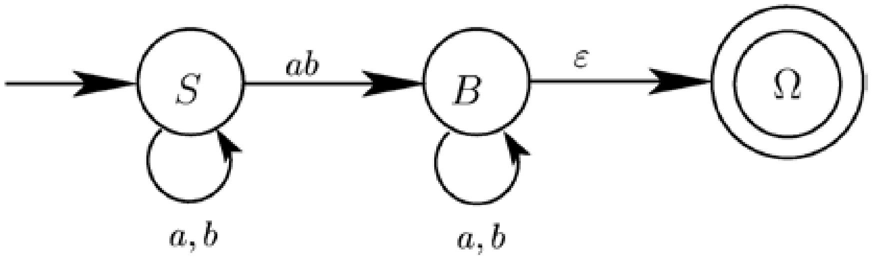
$(S, w_1, B_1) \ldots (B_{n-2}, w_{n-1}, B_{n-1})(B_{n-1}, w_n, \Omega)$

im Automaten $\mathcal{A}$.

Dies zeigt $L(\mathcal{A}) = L(G)$.

Beispiel 6.5

$$\begin{array}{lcl} S & \longrightarrow & aS, \\ S & \longrightarrow & bS, \\ S & \longrightarrow & abB, \\ B & \longrightarrow & aB, \\ B & \longrightarrow & bB, \\ B & \longrightarrow & \varepsilon \end{array}$$



2. Jede erkennbare Sprache ist Typ-3-Sprache.

Es sei $L = L(\mathcal{A})$ für einen NEA $\mathcal{A} = (Q, \Sigma, q_0, \Delta, F)$.

Wir definieren daraus eine Typ-3-Grammatik $G = (N, \Sigma, P, S)$ wie folgt:

$$\begin{array}{rcl} N & := & Q \\ S & := & q_0 \\ P & := & \{p \longrightarrow aq \mid (p, a, q) \in \Delta\} \;\cup \\ & & \{p \longrightarrow \varepsilon \mid p \in F\} \end{array}$$

Ein Pfad in $\mathcal{A}$ der Form

$$(q_0, a_1, q_1)(q_1, a_2, q_2) \ldots (q_{n-1}, a_n, q_n) \ \text{ mit } q_n \in F$$

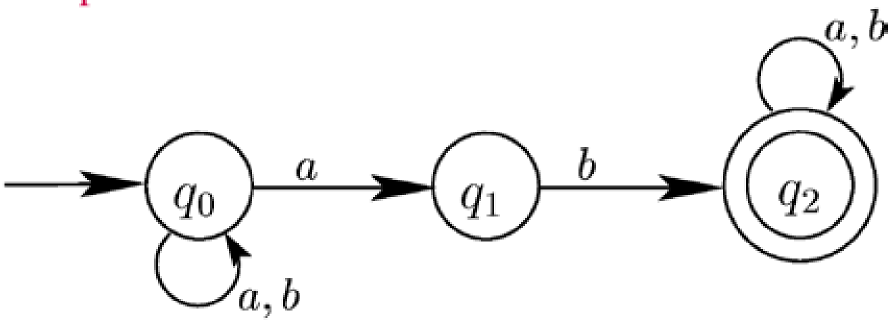
entspricht nun genau einer Ableitung in G:

$$q_0 \vdash_G a_1 q_1 \vdash_G a_1 a_2 q_2 \vdash_G \ldots \vdash_G a_1 \ldots a_n q_n \vdash_G a_1 \ldots a_n$$

Dies zeigt $L(\mathcal{A}) = L(G)$.

Dresden

Beispiel 7.2



liefert die folgenden rechtslinearen Produktionen:

$$
\begin{array}{lcl}
q_0 & \longrightarrow & aq_0, \\
q_0 & \longrightarrow & bq_0, \\
q_0 & \longrightarrow & aq_1, \\
q_1 & \longrightarrow & bq_2, \\
q_2 & \longrightarrow & aq_2, \\
q_2 & \longrightarrow & bq_2, \\
q_2 & \longrightarrow & \varepsilon.
\end{array}
$$

Dresden

Korollar 7.3

$$\boxed{\mathcal{L}_3 \subset \mathcal{L}_2}$$

Beweis:

Wir wissen bereits $\mathcal{L}_3 \subseteq \mathcal{L}_2$.

Wieso ist die Inklusion echt?

Mit Beispiel 6.1 ist

$$L := \{a^n b^n \mid n \geq 0\} \in \mathcal{L}_2.$$

Wir haben gezeigt (Beispiel 2.19 und 3.2), daß L nicht erkennbar/regulär ist,

d.h. mit Satz 7.1 folgt $L \notin \mathcal{L}_3$.

Beispiel 7.4

Ein weiteres Beispiel für eine kontextfreie Sprache, die nicht regulär ist:

$$\{a^n b^m \mid n \neq m\}.$$

Die folgende kontextfreier Grammatik erzeugt diese Sprache:

$G = (N, \Sigma, P, S)$ mit $N = \{S, S_\geq, S_\leq\}$, $\Sigma = \{a, b\}$ und

$$\begin{array}{rcllllll} P & = & \{S & \longrightarrow & aS_\geq, & S & \longrightarrow & S_\leq b, \\ & & S_\geq & \longrightarrow & aS_\geq b, & S_\leq & \longrightarrow & aS_\leq b, \\ & & S_\geq & \longrightarrow & aS_\geq, & S_\leq & \longrightarrow & S_\leq b, \\ & & S_\geq & \longrightarrow & \varepsilon, & S_\leq & \longrightarrow & \varepsilon \;\; \} \end{array}$$

Es gilt nun:

- $S_\geq \vdash_G^* w \in \{a, b\}^* \Rightarrow w = a^n b^m$ mit $n \geq m$,
- $S_\leq \vdash_G^* w \in \{a, b\}^* \Rightarrow w = a^m b^n$ mit $n \geq m$,
- $S \vdash_G^* w \in \{a, b\}^* \Rightarrow w = aa^n b^m$ oder $w = a^m b^n b$ mit $n \geq m$.

d.h. $L(G) = \{a^n b^m \mid n \neq m\}$.

Inhaltsangabe

§ 8. Normalformen kontextfreier Grammatiken

Wir werden zeigen, daß man die Klasse aller kontextfreien Sprachen bereits mit kontextfreien Grammatiken einer eingeschränkten Form erzeugen kann.

Zwei Grammatiken heißen äquivalent, falls sie dieselbe Sprache erzeugen.

Definition 8.1 (Grammatiken ohne überflüssige nichtterminale Symbole)

Es sei $G = (N, \Sigma, P, S)$ eine kontextfreie Grammatik.

1. $A \in N$ heißt terminierend, falls es ein $w \in \Sigma^*$ gibt mit $A \vdash_G^* w$.
2. $A \in N$ heißt erreichbar, falls es $u, v \in (\Sigma \cup N)^*$ gibt mit $S \vdash_G^* uAv$.
3. G heißt reduziert, falls alle Elemente von N erreichbar und terminierend sind.

Lemma 8.2 (Berechnung der erreichbaren Symbole)

Zu einer kontextfreien Grammatik $G = (N, \Sigma, P, S)$ kann man effektiv die Menge der erreichbaren Nichtterminalsymbole bestimmen.

Wir definieren dazu

$$\begin{aligned} E_0 &:= \{S\} \\ E_{i+1} &:= E_i \cup \{A \mid \exists B \in E_i \text{ mit Regel } B \longrightarrow u_1 A u_2 \in P\} \end{aligned}$$

Es gilt $E_0 \subseteq E_1 \subseteq E_2 \subseteq \ldots \subseteq N$.

Da N endlich ist, gibt es ein k mit $E_k = E_{k+1}$ und damit

$$E_k = \bigcup_{i \geq 0} E_i$$

Behauptung:

$$E_k = \{A \in N \mid A \text{ ist erreichbar}\}.$$

Beispiel 8.3

$$
\begin{aligned}
P = \{S &\longrightarrow aS, & S &\longrightarrow SB, & S &\longrightarrow SS, & S &\longrightarrow \varepsilon, \\
A &\longrightarrow ASB, & A &\longrightarrow C, & B &\longrightarrow Cb\}
\end{aligned}
$$

$$E_0 = \{S\} \subset E_1 = \{S, B\} \subset E_2 = \{S, B, C\} = E_3$$

Es ist also A das einzige nicht erreichbare Symbol.

Dresden

Lemma 8.4 (Berechnung der terminierenden Symbole)

Zu einer kontextfreien Grammatik $G = (N, \Sigma, P, S)$ kann man effektiv die Menge der terminierenden Nichtterminalsymbole bestimmen.

Wir definieren dazu

$$\begin{aligned} T_1 &:= \{A \in N \mid \exists w \in \Sigma^* : A \longrightarrow w \in P\} \\ T_{i+1} &:= T_i \cup \{A \in N \mid \exists w \in (\Sigma \cup T_i)^* : A \longrightarrow w \in P\} \end{aligned}$$

Es gilt $T_1 \subseteq T_2 \subseteq T_3 \subseteq \ldots \subseteq N$.

Da N endlich ist, gibt es ein k mit $T_k = T_{k+1}$ und damit

$$T_k = \bigcup_{i \geq 1} T_i$$

Behauptung:

$$T_k = \{A \in N \mid A \text{ ist terminierend}\}$$

Dresden

Satz 8.5 (Leerheitsproblem für kf Sprachen)

Das Leerheitsproblem ist für kontextfreie Sprachen entscheidbar.

Beweis:

Offenbar gilt

$$L(G) \neq \emptyset \ \text{ gdw } \ \exists w \in \Sigma^* : S \vdash_G^* w \ \text{ gdw } \ S \text{ ist terminierend}$$

Dresden

Lemma 8.6 (Erzeugen einer reduzierten Grammatik)

Zu jeder kontextfreien Grammatik G mit $L(G) \neq \emptyset$ kann man effektiv eine äquivalente reduzierte kontextfreie Grammatik erzeugen.

Beweis:

Erster Schritt: Eliminieren nicht terminierender Symbole.

Zu $G = (N, \Sigma, P, S)$ definieren wir $G' := (N', \Sigma, P', S)$, wobei

$$N' := \{A \in N \mid A \text{ ist terminierend in } G\},$$

$$P' := \{A \longrightarrow w \in P \mid A \in N', w \in (N' \cup \Sigma)^*\}.$$

Beachte:

- Aus $L(G) \neq \emptyset$ folgt $S \in N'$!
- Die Elemente von N' sind auch terminierend in G'.

Dresden

Zweiter Schritt: Eliminieren unerreichbare Symbole.

Wir definieren $G'' := (N'', \Sigma, P'', S)$, wobei

$$N'' := \{A \in N' \mid A \text{ ist erreichbar in } G'\},$$

$$P'' := \{A \longrightarrow w \in P' \mid A \in N''\}.$$

Beachte: Ist $A \in N''$ und $A \longrightarrow u_1Bu_2 \in P'$, so ist $B \in N''$.

Man sieht leicht, daß $L(G) = L(G') = L(G'')$ und G'' reduziert ist.

Vorsicht: Die Reihenfolge der beiden Schritte ist wichtig!

Symbole, die in G noch erreichbar waren, müssen es in G' nicht sein:

z.B.: $S \longrightarrow AB$, A terminierend, B nicht.

Lemma 8.7 (Eliminieren von ε-Produktionen)

Es sei G eine kontextfreie Grammatik. Dann läßt sich eine Grammatik G' ohne Regeln der Form $A \longrightarrow \varepsilon$ konstruieren mit $L(G') = L(G) \setminus \{\varepsilon\}$.

Beweis:

(1) Finde alle $A \in N$ mit $A \vdash_G^* \varepsilon$:

$$\begin{aligned} N_1 &:= \{A \in N \mid A \longrightarrow \varepsilon \in P\} \\ N_{i+1} &:= N_i \cup \{A \in N \mid A \longrightarrow B_1 \ldots B_n \in P \text{ für } B_j \in N_i\} \end{aligned}$$

Es gibt ein k mit

$$N_k = N_{k+1} = \bigcup_{i \geq 1} N_i.$$

Für dieses k gilt: $A \in N_k$ gdw $A \vdash_G^* \varepsilon$.

Dresden

(2) Eliminiere in G alle Regeln $A \longrightarrow \varepsilon$.

Um dies auszugleichen, nimmt man für alle Regeln

$$A \longrightarrow u_1 B_1 \dots u_n B_n u_{n+1}$$

mit $B_i \in N_k$ und $u_i \in (\Sigma \cup (N \setminus N_k)^*)$ die Regeln

$$A \longrightarrow u_1 \beta_1 u_2 \dots u_n \beta_n u_{n+1}$$

hinzu mit

- $\beta_i \in \{B_i, \varepsilon\}$ und
- $u_1 \beta_1 u_2 \dots u_n \beta_n u_{n+1} \neq \varepsilon$.

Dresden

Beispiel:

$$
\begin{aligned}
P = \quad & \{S \longrightarrow aS,\ \ S \longrightarrow SS,\ \ S \longrightarrow bA,\\
& A \longrightarrow BB,\\
& B \longrightarrow CC, B \longrightarrow aAbC,\\
& C \longrightarrow \varepsilon\}
\end{aligned}
$$

$N_1 = \{C\}, \quad N_2 = \{C, B\}, \quad N_3 = \{C, B, A\} = N_4$

$$
\begin{aligned}
P' = \quad & \{S \longrightarrow aS,\ \ S \longrightarrow SS,\\
& S \longrightarrow bA,\ \ S \longrightarrow b,\\
& A \longrightarrow BB,\ \ A \longrightarrow B,\\
& B \longrightarrow CC,\ \ B \longrightarrow C,\\
& B \longrightarrow aAbC,\ \ B \longrightarrow aAb,\ \ B \longrightarrow abC,\ \ B \longrightarrow ab\}
\end{aligned}
$$

Die Ableitung

$$S \vdash bA \vdash bBB \vdash bCCB \vdash bCCCC \vdash^* b$$

kann in G' direkt durch $S \vdash b$ erreicht werden.

Dresden

Definition 8.8 (ε-freie kontextfreie Grammatik)

Eine kontextfreie Grammatik heißt ε-frei, falls gilt:

1. Sie enthält keine Regeln $A \longrightarrow \varepsilon$ für $A \neq S$.
2. Ist $S \longrightarrow \varepsilon$ enthalten, so kommt S nicht auf der rechten Seite einer Regel vor.

Dresden

Satz 8.9

Zu jeder kontextfreien Grammatik G kann effektiv eine äquivalente ε-freie Grammatik konstruiert werden.

Beweis:

Konstruiere G' wie im Beweis von Lemma 8.7 beschrieben.

Ist $\varepsilon \notin L(G)$ (d.h. $S \not\vdash_G^* \varepsilon$, also $S \notin N_k$), so ist G' die gesuchte ε-freie Grammatik.

Sonst erweitere G' um ein neues Startsymbol S_0 und die Produktionen

$$S_0 \longrightarrow S \;\text{ und }\; S_0 \longrightarrow \varepsilon.$$

Dresden

Korollar 8.9

$$\mathcal{L}_2 \subseteq \mathcal{L}_1.$$

Beweis:

Offenbar ist jede ε-freie kontextfreie Grammatik eine Typ-1-Grammatik:

Typ-1-Grammatik:

Produktionen der Form

$$u_1 A u_2 \longrightarrow u_1 w u_2$$

mit $|w| \geq 1$, oder

$$S \longrightarrow \varepsilon$$

und S nicht auf rechter Seite einer Produktion.

ε-frei kontextfrei:

Produktionen der Form

$$A \longrightarrow w$$

mit $|w| \geq 1$, oder

$$S \longrightarrow \varepsilon$$

und S nicht auf rechter Seite einer Produktion.

Satz 8.11 (Entferne Produktionen $A \longrightarrow B$)

Zu jeder kontextfreien Grammatik kann man effektiv eine äquivalente kontextfreie Grammatik konstruieren, die keine Regeln der Form $A \longrightarrow B$ $(A, B \in N)$ enthält.

Beweis: O.B.d.A. sei G ε-frei.

(1) Bestimme zu jedem $A \in N$ die Menge $N(A) := \{B \in N \mid A \vdash_G^* B\}$.

$$\begin{aligned} N_0(A) &:= \{A\}, \\ N_{i+1}(A) &:= N_i(A) \cup \{B \in N \mid \exists B' \in N_i(A) : B' \longrightarrow B \in P\} \end{aligned}$$

Es gibt ein k mit $N_k(A) = N_{k+1}(A)$ und für dieses k ist $N(A) = N_k(A)$.

(2) Definiere

$$P' = \{A \longrightarrow w \mid B \longrightarrow w \in P, B \in N(A) \text{ und } w \notin N\}$$

Beispiel

$$P = \{S \longrightarrow A,\ \ A \longrightarrow B,\ \ \underbrace{B \longrightarrow aA,\ \ B \longrightarrow b}_{w \notin N}\}$$

$$N(S) = \{S, A, B\}, \qquad N(A) = \{A, B\}, \qquad N(B) = \{B\}$$

Definition von P':

$$P' = \{A \longrightarrow w \mid B \longrightarrow w \in P, B \in N(A) \text{ und } w \notin N\}$$

$$\begin{aligned} P' = \{ & B \longrightarrow aA,\ \ A \longrightarrow aA,\ \ S \longrightarrow aA, \\ & \underbrace{B \longrightarrow b,}_{B \in N(B)}\ \ \underbrace{A \longrightarrow b,}_{B \in N(A)}\ \ \underbrace{S \longrightarrow b}_{B \in N(S)}\ \} \end{aligned}$$

Dresden

Satz 8.12 (Chomsky-Normalform)

Jede kontextfreie Grammatik läßt sich umformen in eine äquivalente Grammatik in Chomsky-Normalform, d.h. eine Grammatik, die nur Regeln der folgenden Form hat:

$$A \longrightarrow a \quad \text{und} \quad A \longrightarrow BC$$

mit $A, B, C \in N$, $a \in \Sigma$ und eventuell

$$S \longrightarrow \varepsilon,$$

wobei S nicht rechts vorkommt.

Beweis:

1. Konstruiere zu der gegebenen Grammatik zunächst eine äquivalente ε-freie Grammatik und dann eine ohne Regeln der Form $A \longrightarrow B$ (Reihenfolge!).

2. Führe für jedes $a \in \Sigma$ ein neues Nichtterminalsymbol X_a und die Produktion $X_a \longrightarrow a$ ein.

Satz 8.12 (Chomsky-Normalform)

Jede kontextfreie Grammatik läßt sich umformen in eine äquivalente Grammatik in Chomsky-Normalform, d.h. eine Grammatik, die nur Regeln der folgenden Form hat:

$$A \longrightarrow a \quad \text{und} \quad A \longrightarrow BC$$

mit $A, B, C \in N$, $a \in \Sigma$ und eventuell

$$S \longrightarrow \varepsilon,$$

wobei S nicht rechts vorkommt.

Beweis:

3. Ersetze in jeder Produktion $A \longrightarrow w$ mit $w \notin \Sigma$ alle Terminalsymbole a durch die zugehörigen X_a.

4. Produktionen $A \longrightarrow B_1 \ldots B_n$ für $n > 2$ werden ersetzt durch

$$A \longrightarrow B_1C_1, C_1 \longrightarrow B_2C_2, \ldots, C_{n-2} \longrightarrow B_{n-1}B_n$$

wobei die C_i jeweils neue Symbole sind.

Wortproblem für kontextfreie Sprachen:

Es sei G eine kontextfreie Grammatik.

Das Wortproblem für $L = L(G)$ ist die folgende Frage:

Gegeben: Ein Wort w.

Frage: Ist $w \in L(G)$?

Das Wortproblem für reguläre Sprachen

Für einen NEA $\mathcal{A}$ kann man $w \in L(\mathcal{A})$ entscheiden, da ein Pfad in $\mathcal{A}$, der w erkennt, genau die Länge $|w|$ haben muß.

Es gibt aber nur endlich viele Pfade dieser festen Länge.

Entsprechend kann eine Ableitung von w in einer rechtslinearen Grammatik G höchstens die Länge $|w|$ haben.

Bei allgemeinen kontextfreien Grammatiken kann man keine Schranke für die Länge einer Ableitung von w aus S angeben.

Bei Grammatiken in Chomsky-Normalform kann man eine derartige Schranke für die Länge einer Ableitung von $w \in \Sigma^*$ angeben:

- Produktionen der Form $A \longrightarrow BC$ verlängern um 1, d.h. sie werden $|w| - 1$-mal angewandt.

- Produktionen der Form $A \longrightarrow a$ erzeugen genau ein Terminalsymbol von w, d.h. sie werden genau $|w|$-mal angewandt.

Es folgt: $w \in L(G) \setminus \{\varepsilon\}$ wird durch eine Ableitung der Länge $2|w| - 1$ erzeugt. Diese kann man alle erzeugen.

Komplexität:

Da es aber $\geq 2^n$ Ableitungen der Länge n geben kann, liefert dies ein exponentielles Verfahren zur Entscheidung des Wortproblems.

Ziel: Angabe eines polynomiellen (kubischen) Verfahrens zur Entscheidung des Wortproblems.

Definition 8.13

Es sei $G = (N, \Sigma, P, S)$ eine kontextfreie Grammatik in Chomsky-Normalform und $w = a_1 \ldots a_n \in \Sigma^*$.

Wir definieren:

- $w_{ij} := a_i \ldots a_j$ (für $i \leq j$)
- $N_{ij} := \{A \in N \mid A \vdash_G^* w_{ij}\}$

Dresden

- $w_{ij} := a_i \dots a_j$ (für $i \leq j$)
- $N_{ij} := \{A \in N \mid A \vdash_G^* w_{ij}\}$

Mit dieser Notation gilt nun:

1. $S \in N_{1n}$ gdw $w \in L(G)$.
2. $A \in N_{ii}$ gdw $A \vdash_G^* a_i$ gdw $A \longrightarrow a_i \in P$.
3. $A \in N_{ij}$ für $i < j$ gdw $A \vdash_G^* a_i \dots a_j$

   gdw $\exists A \longrightarrow BC \in P$ und ein $k, i \leq k < j$ mit $B \vdash_G^* a_i \dots a_k$ und $C \vdash_G^* a_{k+1} \dots a_j$

   gdw $\exists A \longrightarrow BC \in P$ und $k, i \leq k < j$ mit $B \in N_{ik}$ und $C \in N_{(k+1)j}$

Dresden

Diese Überlegungen liefern das folgende iterative Verfahren zur Bestimmung von N_{1k}:

Algorithmus 8.14 (CYK-Algorithmus von Cocke, Younger, Kasami)

$\texttt{for } i := 1 \texttt{ to } n \texttt{ do}$

$N_{ii} := \{A \mid A \longrightarrow a_i \in P\}$

$\texttt{for } d := 1 \texttt{ to } n-1 \texttt{ do}$ (wachsende Differenz d)

$\texttt{for } i := 1 \texttt{ to } n-d \texttt{ do}$

$j := i + d$ (Differenz $j - i$ ist d)

$N_{ij} := \emptyset$

$\texttt{for } k := i \texttt{ to } j-1 \texttt{ do}$

$N_{ij} := N_{ij} \cup \{A \mid \exists A \longrightarrow BC \in P \text{ mit } B \in N_{ik} \text{ und } C \in N_{(k+1)j}\}$

Beachte: In der innersten Schleife sind die Differenzen $k - i$ und $j - (k+1)$ kleiner als das aktuelle d, also sind N_{ik} und $N_{(k+1)j}$ bereits berechnet.

Beispiel

$$\begin{aligned} P = \quad & \{S \longrightarrow SA,\ \ S \longrightarrow a, \\ & A \longrightarrow BS, \\ & B \longrightarrow BB,\ \ B \longrightarrow BS,\ \ B \longrightarrow b,\ \ B \longrightarrow c\} \end{aligned}$$

und $\qquad w = abacba.$

Satz 8.15

Für eine vorgegebene Grammatik in Chomsky-Normalform entscheidet Algorithmus 8.14 die Frage „$w \in L(G)$?“ in der Zeit $O(|w|^3)$.

Beweis:

Drei geschachtelte Schleifen, die jeweils $\leq |w| = n$ Interationen durchlaufen.

Daraus folgt: $|w|^3$ Schritte in der innersten Schleife.

Beachte:

Die Größe von G ist hier als konstant angenommen (fest vorgegebenes G).

Daher ist die Suche nach den Produktionen $A \longrightarrow BC$ und $A \longrightarrow a_i$ in konstanter Zeit möglich.