

Eigenschaften von CNF-Grammatiken. Die Ableitungsbäume von CNF-Grammatiken sind stets Binärbäume. Innere Knoten, die für die Anwendung einer Regel der Form $A \rightarrow BC$ stehen, haben genau zwei Söhne. Innere Knoten mit nur einem Sohn stehen für die Anwendung einer Terminalregel $A \rightarrow a$. Diese sind nur auf der untersten Ebene möglich. Ist G eine CNF-Grammatik und $w \in \mathcal{L}(G)$ ein Wort der Länge n , so ist $n \geq 1$ und jede Ableitung von w in G besteht aus genau $2n-1$ Regelanwendungen, nämlich genau n Terminalregeln des Typs $A \rightarrow a$, mit denen die Zeichen von w generiert werden, und genau $n-1$ Regeln des Typs $A \rightarrow BC$. Für die Höhe zugehöriger Ableitungsbäume gilt:

Lemma 3.8 (Höhe von Ableitungsbäumen in CNF-Grammatiken). *Sei G eine CNF-Grammatik und $w \in \mathcal{L}(G)$ ein Wort der Länge n . Ist T ein Ableitungsbaum für w in G , so gilt:*

$$\log n + 1 \leq \text{Höhe von } T \leq n$$

Beweis. Die obere Schranke n ergibt sich aus der Tatsache, dass jeder Pfad in T von der Wurzel zu einem Blatt nur durch genau einen inneren Knoten führt, der für die Anwendung einer Terminalregel $A \rightarrow a$ steht. Da jede Ableitung von w genau $n-1$ Regeln des Typs $A \rightarrow BC$ einsetzt, ist die Höhe von T durch $(n-1) + 1 = n$ beschränkt.

Der Nachweis der unteren Schranke $\log n + 1$ kann durch Induktion nach n erfolgen. Genauer zeigen wir, dass die Ableitungsbäume aller Ableitungen $A \Rightarrow x_1 \Rightarrow \dots \Rightarrow x_{2n-1} = w$ mit beliebiger Variable A und einem Wort w gebildet aus Terminalzeichen mit $|w| = n$ mindestens die Höhe $\log n + 1$ haben. Der Induktionsanfang $n = 1$ ist klar, da Ableitungen von Wörtern der Länge 1 in CNF-Grammatiken nur aus der Anwendung einer Terminalregel $A \rightarrow a$ bestehen. Zugehörige Ableitungsbäume haben die Höhe 1. Nun zum Induktionsschritt $n-1 \implies n$, wobei $n \geq 2$. Sei $w = a_1 a_2 \dots a_n$ und T ein Ableitungsbaum von w , dessen Wurzel mit A beschriftet ist. In der zugehörigen Ableitung muss im ersten Schritt eine Regel $A \rightarrow BC$ eingesetzt werden. Die weiteren Ableitungsschritte sind aus den Schritten von Ableitungen $B \Rightarrow^* a_1 \dots a_m$ und $C \Rightarrow^* a_{m+1} \dots a_n$ zusammengesetzt, wobei $1 \leq m \leq n-1$. Diese beiden Ableitungen sind in T durch die Teilbäume der beiden mit B und C beschrifteten Söhne der Wurzel dargestellt. Eines der beiden Wörter $a_1 \dots a_m$ oder $a_{m+1} \dots a_n$ muss mindestens die Länge $n/2$ haben. Etwa $m \geq n/2$. Nach Induktionsvoraussetzung hat jeder Ableitungsbaum für $B \Rightarrow^* a_1 \dots a_m$ mindestens die Höhe $\log m + 1 \geq \log n$. Die Höhe von T ist daher mindestens $\log n + 1$. \square

3.3 Der Cocke-Younger-Kasami Algorithmus

Eine der zentralen algorithmischen Problemstellungen für Grammatiken ist das *Wortproblem*. Dieses hat als Eingabe eine Grammatik G und ein Wort w über das Terminalalphabet von G und fragt, ob w aus G hergeleitet werden kann, also ob $w \in \mathcal{L}(G)$. Eine wichtige Instanz ist das Parsingproblem des Compilers, bei dem w für den vom Benutzer geschriebenen Programmtext steht und der Parser zu prüfen hat, ob die syntaktischen Regeln der betreffenden Programmiersprache eingehalten wurden. Im Compilerbau werden hierfür sehr ausgefeilte Algorithmen für spezielle Arten kontextfreier Grammatiken eingesetzt. Wir behandeln hier einen konzeptionell einfachen Algorithmus, der von einer kontextfreien Grammatik in Chomsky Normalform ausgeht.

Im Folgenden sei $G = (V, \Sigma, \mathcal{P}, S)$ eine CNF-Grammatik und $w = a_1 a_2 \dots a_n \in \Sigma^+$. Wir schreiben $w_{i,j}$ für das Teilwort der Länge j , das an Position i beginnt, also $w_{i,j} = a_i a_{i+1} \dots a_{i+j-1}$.

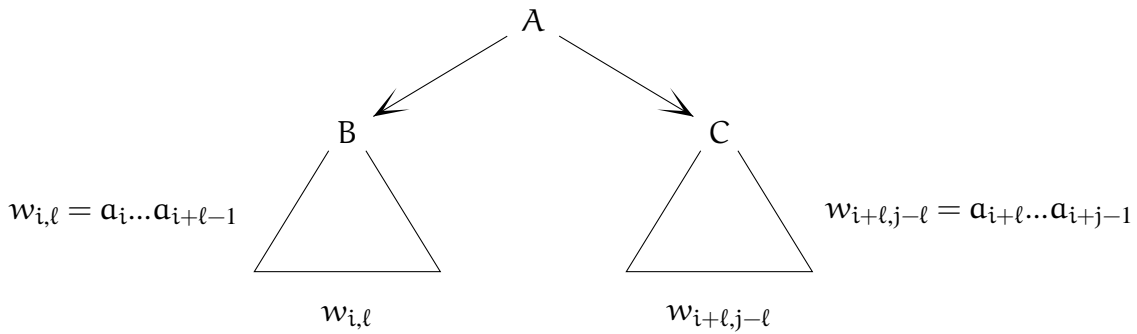


Abbildung 24: Ableitungsbaum für $w_{i,j} = a_i a_{i+1} \dots a_{i+j-1}$

Dabei ist $1 \leq i \leq n$ und $1 \leq j \leq n+1-i$. Für den Test, ob $w \in \mathcal{L}(G)$ liegt, wenden wir die Methode des dynamischen Programmierens zur Bestimmung der Variablenmengen

$$V[i,j] = \{A \in V : A \Rightarrow^* w_{i,j}\}$$

an. Der Bereich von i und j ist wie zuvor, also $1 \leq i \leq n$, $1 \leq j \leq n+1-i$. Die Menge $V[i,j]$ besteht also aus allen Variablen A , aus denen sich das Teilwort $w_{i,j}$ von w ableiten lässt. Die Antwort auf das Wortproblem für G und $w = w_{1,n}$ ergibt sich durch Betrachtung der Variablenmenge $V[i,j]$ für die Indizes $i = 1$ und $j = n$:

$$w \in \mathcal{L}(G) \quad \text{gdw} \quad S \in V[1,n].$$

Die Berechnung der Variablenmengen $V[i,j]$ erfolgt in Bottom-Up Manier mit zunehmender Länge j der betrachteten Teilwörter von w . Die Initialisierungsphase betrachten wir Teilwörter der Länge 1, also die einzelnen Zeichen von w . Da G in CNF ist, sind diese nur durch die Anwendung einer Terminalregel herleitbar. Also:

$$V[i,1] = \{A \in V : A \rightarrow a_i\}$$

Die Ableitungen, die mit einer Variablen A starten, und eines der Teilwörter von w der Länge $j \geq 2$ erzeugen, haben mindestens die Länge 2. Im ersten Ableitungsschritt muss also eine Regel der Form $A \rightarrow BC$ eingesetzt werden. Für $A \in V$ gilt daher $A \Rightarrow^* w_{i,j}$ genau dann, wenn es eine Regel $A \rightarrow BC$ gibt, so dass aus B und C ein nicht-leeres Präfix der Länge ℓ bzw. Suffix der Länge $j-\ell$ von $w_{i,j}$ herleitbar ist. Die Struktur eines zugehörigen Ableitungsbaums ist in Abbildung 24 dargestellt. Es gilt also:

$$A \Rightarrow^* w_{i,j} \quad \text{gdw} \quad \begin{cases} \text{es gibt eine Regel } A \rightarrow BC \text{ und eine Zahl } \ell \in \{1, \dots, j-1\} \\ \text{mit } B \Rightarrow^* w_{i,\ell} \text{ und } C \Rightarrow^* w_{i+l,j-\ell} \end{cases}$$

Wir erhalten folgende rekursive Charakterisierung der Variablenmengen $V[i,j]$.

$$V[i,j] = \bigcup_{\ell=1}^{j-1} \{A \in V : \text{es gibt eine Regel } A \rightarrow BC \text{ mit } B \in V[i,\ell] \text{ und } C \in V[i+l,j-\ell]\}$$

Diese Beobachtung wird im Algorithmus von Cocke-Younger-Kasami, kurz CYK-Algorithmus genannt, angewandt. Siehe Algorithmus 4 auf Seite 84. Dieser berechnet in Bottom-Up-Manier

$$V[1,1] = V[2,1] = \{A\} \quad \text{und} \quad V[3,1] = V[4,1] = \{B\}$$

gesetzt. Daraus ergibt sich $V[1,2] = V[3,2] = \emptyset$ und $V[2,2] = \{S\}$. Im zweiten Schleifendurchlauf erhalten wir

$$V[1,3] = \emptyset \quad \text{und} \quad V[2,3] = \{C\},$$

da $S \in V[2,2]$, $B \in V[3,1]$ und $C \rightarrow SB$. Wegen $A \in V[1,1]$ und $C \in V[2,3]$ ergibt sich $V[1,4] = \{S\}$. Die oben erwähnte Tabelle hat also die Gestalt:

$V[1,4]$	$\{S\}$			
$V[i,3]$	\emptyset	$\{C\}$		
$V[i,2]$	\emptyset	$\{S\}$	\emptyset	
$V[i,1]$	$\{A\}$	$\{A\}$	$\{B\}$	$\{B\}$
	a	a	b	b

Der Algorithmus terminiert also mit der Antwort "ja". ■

Weitere algorithmische Probleme für CFG. Das Leerheitsproblem "ist $\mathcal{L}(G) = \emptyset$ für gegebene CFG G ?" lässt sich mit dem auf Seite 76 skizzierten Markierungsalgorithmus zur Bestimmung nutzloser Variablen lösen. Das Startsymbol S ist genau dann nutzlos, wenn $\mathcal{L}(G) = \emptyset$. Auch das Endlichkeitsproblem "ist $\mathcal{L}(G)$ endlich?" ist für CFG effizient lösbar. Darauf gehen wir später ein. Viele andere algorithmische Fragestellungen für CFG, z.B. das Äquivalenzproblem "gilt $\mathcal{L}(G_1) = \mathcal{L}(G_2)$?" oder die Frage, ob $\mathcal{L}(G)$ regulär ist, sind jedoch unentscheidbar, d.h., nicht algorithmisch lösbar.

3.4 Eigenschaften kontextfreier Sprachen

Wir zeigen zunächst, dass jede kontextfreie Sprache eine Pumpeigenschaft besitzt, die für den Nachweis der Nicht-Kontextfreiheit nützlich sein kann. Später diskutieren wir Abschlusseigenschaften für die Klasse der kontextfreien Sprachen unter den fünf gängigen Kompositionsooperatoren für formale Sprachen.

3.4.1 Das Pumping Lemma für kontextfreie Sprachen

In Analogie zum Pumping Lemma für reguläre Sprachen gibt es ein notwendiges Kriterium für kontextfreie Sprachen. Wir werden dieses mit Hilfe der Ableitungsbäume für CNF-Grammatiken beweisen und später für den Entwurf eines Endlichkeitstests für kontextfreie Grammatiken benutzen.

Satz 3.11 (Pumping Lemma für kontextfreie Sprachen). *Sei L eine kontextfreie Sprache. Dann gibt es eine natürliche Zahl n , so dass sich jedes Wort $z \in L$ der Länge $\geq n$ wie folgt zerlegen lässt: $z = uvwxy$, wobei*

$$(1) \quad uv^kwx^ky \in L \text{ für alle } k \in \mathbb{N}$$

$$(2) |vx| \geq 1$$

$$(3) |vwx| \leq n.$$

Beweis. Wir können o.E. annehmen, dass $\varepsilon \notin L$. (Ist $\varepsilon \in L$, dann betrachten wir die Sprache $L \setminus \{\varepsilon\}$ anstelle von L .) Sei G eine CNF-Grammatik mit $L = \mathcal{L}(G)$ (siehe Satz 3.4, Seite 77). Weiter sei $N = |V|$ die Anzahl an Variablen in G . Die im Pumping Lemma genannte Konstante n ist wie folgt definiert:

$$n = 2^N$$

Sei $z \in L$ mit $|z| \geq n$. Zu zeigen ist, dass sich z in fünf Teilwörter u, v, w, x, y zerlegen lässt, so dass Eigenschaften (1), (2) und (3) erfüllt sind. Als Hilfsmittel für die Definition solcher Teilwörter u, v, w, x, y verwenden wir einen Ableitungsbaum T für z . Dieser ist ein Binärbaum mit $|z|$ Blättern. Innere Knoten mit genau einem Sohn repräsentieren eine Regel der Form $A \rightarrow a$. Sie haben die Höhe 1. Alle anderen inneren Knoten stehen für eine Regel der Form $A \rightarrow BC$. Sie haben genau zwei Söhne. Die Höhe von T sei $h+1$. Wegen Lemma 3.8 (Seite 82) gilt:

$$h \geq \log|z| \geq \log n = N$$

Sei $v_0 v_1 \dots v_{h+1}$ ein Pfad in T der Länge $h+1$ von der Wurzel v_0 zu einem Blatt v_{h+1} . Also ist v_{h+1} ein Zeichen $a \in \Sigma$ von z . Weiter sei A_i die Markierung des Knotens v_i , $i = 0, 1, \dots, h$. Dann ist $A_0 = S, A_1, \dots, A_h$ eine Folge von Variablen der Länge $h+1 \geq N+1$. Also gibt es eine Variable A und Indizes $i, j \in \{0, 1, \dots, h\}$ mit $i < j$, so dass $A = A_i = A_j$ und so dass die Variablen A_{i+1}, \dots, A_h paarweise verschieden sind.

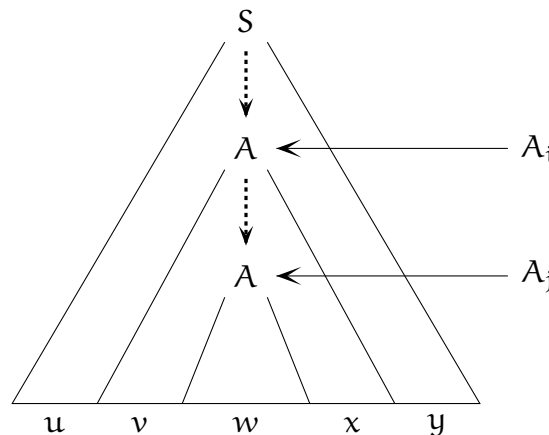


Abbildung 25: Zerlegung von z in $uvwxy$

Wir zerlegen z in $z = uvwxy$ gemäß der Skizze in Abbildung 25 auf Seite 86 und zeigen, dass die geforderten Bedingungen erfüllt sind.

ad (1). Ableitungen der Wörter $uv^k wx^k y$ ergeben sich durch Kombinieren der Ableitungen

$$S \Rightarrow^* uA_i y = uA y, \quad A = A_j \Rightarrow^* w, \quad A = A_i \Rightarrow^* vA_j x = vA x.$$

Es folgt $S \Rightarrow^* uv^k wx^k y$ (also $uv^k wx^k y \in L$) für alle $k \in \mathbb{N}$. Abbildung 26 skizziert die Ableitungsbäume für $k = 0$ und $k = 2$.

ad (2). Da G in CNF ist, ist $v \neq \varepsilon$ oder $x \neq \varepsilon$. (Da v_i ein innerer Knoten der Höhe > 1 ist, steht v_i für das Anwenden einer Regel der Gestalt $A_i \rightarrow BC$. Ist z.B. v_j ein Nachfolger des mit B markierten Sohns von v_i , so ist das aus C hergeleitete Wort ein Suffix von x mit $|x| \geq 1$.) Es folgt $|vx| \geq 1$.

ad (3). Wir betrachten den Teilbaum T' mit Wurzel v_i . T' ist ein Ableitungsbaum für die Ableitung $A_i \Rightarrow^* vwx$. Die Höhe von T' ist $h-i+1$, da $v_i, v_{i+1}, \dots, v_{h+1}$ ein längster Pfad in T' ist. Da A_{i+1}, \dots, A_h paarweise verschieden sind, gilt $h-i \leq N$. Die Höhe von T' ist daher $\leq N+1$. Da in jeder CNF-Grammatik die Höhe eines Ableitungsbaum für ein Wort der Länge ℓ mindestens $\log \ell + 1$ ist (siehe Lemma 3.8 auf Seite 82), folgt hieraus:

$$|vwx| \leq 2^{h-i} \leq 2^N = n$$

Die Zerlegung $z = uvwxy$ erfüllt also die drei geforderten Bedingungen. □

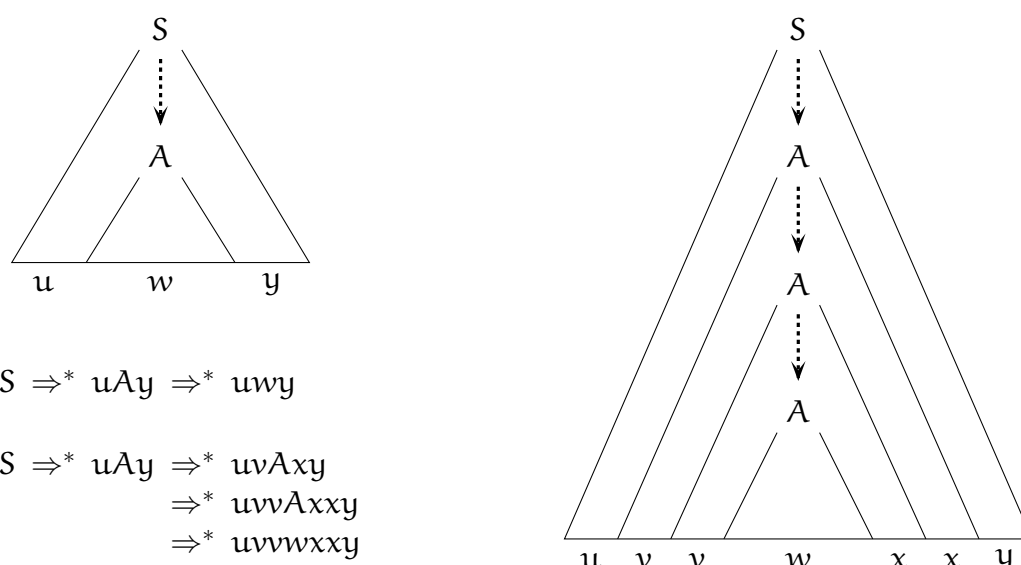


Abbildung 26: Pumpeigenschaft für $k = 0$ und $k = 2$

Nicht-kontextfreie Sprachen. Das Pumping Lemma kann oftmals hilfreich sein, um nachzuweisen, dass eine Sprache *nicht* kontextfrei ist. Als Beispiel betrachten wir die Sprache

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

und zeigen mit Hilfe des Pumping Lemmas, dass L nicht kontextfrei ist. Wir nehmen an, dass L kontextfrei ist und führen diese Annahme zu einem Widerspruch. Sei n die Zahl aus dem Pumping Lemma und $z = a^n b^n c^n$. Das Wort z liegt in L und hat die Länge $3n > n$. Es gibt also eine Zerlegung $z = uvwxy$, so dass die Eigenschaften (1), (2) und (3) aus dem Pumping Lemma erfüllt sind. Wegen $|vwx| \leq n$ (Bedingung (2)) ist vwx entweder ein Teilwort von $a^n b^n$ oder von $b^n c^n$. Wegen (1) ist wenigstens eines der Wörter v oder x nicht leer. Daher ist es nicht möglich, dass die Anzahl der Vorkommen der drei Symbole a , b und c in uv^2wx^2y gleich ist. Daher ist $uv^2wx^2y \notin L$. Dies steht im Widerspruch zur Pumpeigenschaft (3).