Nichtterminalsymbole, welche in Terminalsymbole überführt werden können und $\overline{P}$ ist die Menge der Produktionen, welche Terminalsymbole produzieren.

Formal:

$$\begin{aligned}
&\exists w \in \Sigma^* \text{ mit } A \overset{*}{\Rightarrow} w \Leftrightarrow A \in V_N \\
&L = \emptyset \Leftrightarrow S \notin V_N \\
&\overline{G} =_{Df} \left(\overline{V}, \Sigma, \overline{P}, S\right) \text{ mit} \\
&\overline{\mathrm{V}} =_{\mathrm{Df}} V' \cap \left(V_N \cup \Sigma\right) = V_N \cup \Sigma \\
&\overline{P} =_{Df} P' \cap \left(\overline{V} \times \overline{V}^*\right)
\end{aligned} \tag{15.6}$$

Somit haben wir nun nur noch Nichtterminalsymbole, welche in Terminalsymbole überführt werden können.

- Schritt 2: Nun gilt zu überprüfen welche Symbole eigentlich erreichbar sind. Dazu betrachtet man alle Nichtterminalsymbole A, welche auf der Rechten Seite einer Produktion auftauchen. Man definiert rekursiv die Übernahme aller Nichtterminalsymbole A, welche über ein weiteres Nichtterminalsymbol B erreichbar sind, das wiederum vom Startsymbol erreichbar ist. Auf diese Weise bekommt man alle Symbole, welche in Terminalsymbole überführt werden können und die vom Startzustand aus erreichbar sind. Ist dann einmal Gleichheit von zwei aufainander folgenden Mengen H_i und H_{i-1} erreicht, so bleibt diese Gleichheit bestehen. Dieser Fall muss auftreten, da der Symbolvorrat endlich ist.

  Formal drückt man das so aus:

$$H_0 =_{Df} \left\{S \middle| S \in \overline{V}\right\} \tag{15.7}$$

$$H_1 =_{Df} \left\{A \middle| S \to \alpha A \beta \in \overline{P}, \alpha, \beta \in \overline{V}^*\right\} \cup H_0 \tag{15.8}$$

$$H_n =_{Df} \left\{A \middle| B \to \alpha A \beta \in \overline{P}, B \in H_{n-1}, \alpha, \beta \in \overline{V}^*\right\} \cup H_{n-1} \tag{15.9}$$

$$\exists N : H_N = H_{N-1} \quad \Rightarrow \quad H_{N+K} = H_N \quad \forall K \tag{15.10}$$

  Ein Nichtterminalsymbol A ist also dann enthalten, wenn es erreichbar ist. Wieder gilt auch die Umkehrung. Ist ein Element erreichbar, so ist es auch in der Menge enthalten. Schließlich ist V um all die Nichtterminalsymbole reduziert, die nicht erreichbar sind. Ebenso werden die Produktionen aus P genommen, welche nicht verwendet werden können.

  Formal:

$$\begin{aligned}
&A \in H_N \Leftrightarrow S \overset{*}{\Rightarrow} \alpha A \beta, \alpha, \beta \in \overline{V}^* \\
&V =_{Df} \overline{V} \cap \left(H_N \cup \Sigma\right) = H_N \cup \Sigma \\
&P =_{Df} \overline{P} \cap \left(V \times V^*\right)
\end{aligned} \tag{15.11}$$

15.1.2 Das Problem der Ausführung der beiden Schritte

Kann nun Schritt 2 das Ergebnis von Schritt 1 kaputtmachen? Angenommen es gäbe kein Wort w, welches von A erzeugt wird. Dann muss aufgrund von Schritt 1 ein Symbol existieren, das Abgeleitet werden könnte, jedoch aufgrund von Schritt 2 entfernt worden ist. Die Produktion war also nicht erreichbar. Formal:

angenommen $A \overset{*}{\not\Rightarrow} w, \forall w$

dann gilt nach Schritt 1 : $A \overset{*}{\Rightarrow} \alpha B \beta \Rightarrow \alpha\gamma\beta \overset{*}{\Rightarrow} w$

aufgrund von Schritt 2 : $\{B,\gamma\} \not\subseteq (H_N \cup \Sigma)^*$

Da aber A erreichbar ist, ist auch B und alle Nichtterminalsymbole in γ erreichbar.

Die Antwort darauf ist also *Nein*, jedoch ergibt sich ein Problem wenn man Schritt 2 vor Schritt 1 ausführt. Dies wird durch das folgende Beispiel deutlich:

Beispiel:

$S \to AD$
$S \to D$
$A \to BC$
$B \to bB$
$C \to cC$
$C \to c$
$D \to d$
$F \to f$

Für nebenstehende Grammatik ergibt sich durch den ersten Schritt zunächst:

$V_1 = \{C, D, F\}$

Hier wurden also die Nichtterminalsymbole genommen, die direkt in Terminalsymbolen enden.

$V_2 = \{C, D, F\} \cup \{S\} = \{S, C, D, F\}$

Nun wurde das Startsymbol aufgrund der Produktion $S \to D$ übernommen.

$V_3 = \{S, C, D, F\} = V_2 = V_N$

Die weitere Betrachtung zeigt, dass keine weiteren Nichtterminalsymbole übernommen werden müssen also folgt:

$\overline{V} = \{S, C, D, F, b, c, d, f\}$

$\overline{P}$: $S \to D$
$C \to cC$
$C \to c$
$D \to d$
$F \to f$

Mit $\overline{P}$ wie nebenstehend definiert, kann man die Grammatik $\overline{G} = (\overline{V}, \Sigma, \overline{P}, S)$ aufstellen.

Für den zweiten Schritt ergibt sich dann:

$$H_0 = \{S\}$$
$$H_1 = \{S\} \cup \{D\}$$
$$H_2 = \{S, D\} = H_1 = H_N$$

P: $S \to D$
$D \to d$

Dabei wurde jeweils nach der Erreichbarkeit der Symbole gesehen. Zuerst wird das Startsymbol eingeschlossen, das einzige von dort aus zu erreichende Symbol ist dann D gewesen.

Was passiert nun, wenn man diese Reihenfolge vertauscht?

$S \to AD$
$S \to D$
$A \to BC$
$B \to bB$
$C \to cC$
$C \to c$
$D \to d$

Durch die Anwendung des zweiten Schrittes zuerst ergibt sich:

$H_0 = \{S\}$
$H_1 = \{S\} \cup \{A, D\}$
$H_2 = \{S, A, D\} \cup \{B, C\}$
$H_3 = \{S, A, D, B, C\} = H_2 = H_N$

Dabei wurde jeweils nach der Erreichbarkeit der Symbole gesehen. Zuerst wird das Startsymbol eingeschlossen, dann übernimmt man die Symbole A, D und schließlich noch B, C.

Wendet man nun den ersten Schritt an, so übernimmt man zuerst C, D und kommt zu:

$\bar{P}$: $S \to D$
$C \to cC$
$C \to c$
$D \to d$

$V_1 = \{C, D\}$

nun nimmt man noch das Startsymbol hinzu, da gilt $S \to D$:

$V_2 = \{C, D\} \cup \{S\} = \{S, C, D\}$

Für alle folgenden Iterationen gilt:

$V_3 = \{S, C, D\} = V_2 = V_N$

Man bekommt also ein anderes Ergebnis. Die überflüssigen (nicht erreichbaren) Produktionen $C \to cC$ und $C \to c$ wurden mit übernommen. Der Grund dafür liegt hier bei der Produktion $A \to BC$ hier kann B nicht in ein Terminalsymbol überführt werden. C hingegen kann in ein Terminalsymbol überführt werden. Es ist also wichtig die Reihenfolge einzuhalten.

Wie kann man die Grammatik nun noch weiter minimieren?

15.1.3 Zyklische Produktionen

Sei nun G eine Grammatik. Dann heißt $A \to B \in P$ eine zyklische Produktion. Da wir eine kontextfreie Grammatik gegeben haben, kann man zyklische Produktionen herausschneiden. Die Ableitungsschritte der zyklischen Produktionen müssen dann jedoch simuliert werden.

Satz:

Sei $L \subseteq \Sigma^*$ eine kontextfreie Sprache. Dann gibt es eine kontextfreie Grammatik G mit:

1. $L = L(G)$
2. $\forall A, B \in V - \Sigma : A \to B \notin P$

Beweis:

Sei $G' = (V', \Sigma, P', S)$ eine kontextfreie Grammatik mit $L(G') = L$. Dann gilt für alle $A \in V - \Sigma$:

$$\begin{aligned} &V_A^{(1)} = \{B \mid A \to B \in P'\} \\ &V_A^{(n)} = V_A^{(n-1)} \cup \left\{C \,\middle|\, B \to C \in P' \wedge B \in V_A^{(n-1)}\right\} \\ &\exists N : V_A^{(N)} = V_A^{(N+1)} \wedge k \geq 0 : V_A^{(N)} = V_A^{(N+k)} \end{aligned} \tag{15.12}$$

Als neue Produktionen gelten also die „alten“ Produktionsregeln, ohne zyklische Produktionen, jedoch erweitert um die Produktionen, die durch den Zyklus erreicht werden könnten.

$$P =_{Df} \left[P' - \left\{ A \to B \middle| A \to B \in P' \right\} \right] \cup \left\{ A \to \alpha \middle| \alpha \notin V - \Sigma \wedge B \in V_A^{(N)} \right\} \tag{15.13}$$

Offensichtlich ist nun $L = L(G') = L(G)$ für $G = \{V', \Sigma, P, S\}$.

Beispiel:

$S \to AB$
$A \to C$
$B \to D$
$C \to c$
$D \to DD$
$D \to d$
$D \to E$
$E \to f$

$V_S^{(N)} = \varnothing$
$V_E^{(N)} = \varnothing$
$V_{\hat{A}}^{(1)} = \{C\}$
$V_A^{(2)} = \{C\} = V_A^{(N)}$
$V_B^{(1)} = \{D\}$
$V_B^{(2)} = \{D, E\} = V_B^{(N)}$
$V_D^{(1)} = \{E\} = V_D^{(N)}$

$P:$ $S \to AB$
$D \to DD$
$C \to c$
$D \to d$
$E \to f$
$A \to c$
$B \to d$
$B \to DD$
$B \to f$
$D \to f$

15.1.4 Zusammenfassung

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ heißt reduziert, genau dann wenn:

1. P hat keine Produktionen vom Typ $A \to B$ mit $A, B \in V - \Sigma$. Es gibt also keine zyklischen Produktionen.
2. Es gibt keine ε-Produktionen. Das Startsymbol kann nur dann gleich ε sein, wenn es in der Sprache ist.
   - S taucht auf keiner rechten Seite auf,
   - Alle Symbole $A \in (V - \Sigma) - \{S\}$ können keine Produktionen $A \to \varepsilon$ haben.
   - $S \to \varepsilon \in P \Leftrightarrow \varepsilon \in L(G)$
3. Jedes Nichtterminalsymbol ist brauchbar. G hat keine überflüssigen Symbole und Produktionen d.h.:
   - $\forall A \in V - \Sigma \exists w \in \Sigma^* : A \overset{*}{\Rightarrow} w$
   - $\forall A \exists \alpha : \alpha, \beta \in V^* : S \overset{*}{\Rightarrow} \alpha A \beta$

Satz:

Sei L kontextfrei. Dann existiert eine reduzierte kontextfreie Grammatik $G = (V, \Sigma, P, S)$ mit $L = L(G)$.

Beweis:

Sei L kontextfrei. Dann existiert eine kontextfreie Grammatik G' mit $L = L(G')$:

a) Man modifiziere G', um Eigenschaft 2) zu erhalten, und erhält G_1,

b) Man modifiziere G_1 zu G_2 mit Eigenschaft 1)

c) Man modifiziert schließlich G_2 zu G mit Eigenschaft 3)

G ist dann reduziert und es gilt weiterhin $L = L(G)$.

Bemerkung:

Es ist jedoch darauf zu achten die Reihenfolge zu beachten. Wie wir schon an dem Beispiel gesehen haben, kann die Ausführung in anderer Reihenfolge das Ergebnis verändern. Insbesondere bedeutet das für den letzten Beweis:

a) man muss A) vor B) ausführen, da A) Produktionen vom Typ $A \rightarrow B$ mit $A, B \in V - \Sigma$ einführen kann.

b) B) zerstört nicht die Eigenschaft 2)

c) C) zerstört nicht die Eigenschaften 1) und 2)

16 Vorlesung vom 6. Juni 2000

16.1 Chomsky-Normalform

Satz (Chomsky-Normalform):

Sei $G=(V,\Sigma,P,S)$ eine reduzierte kontextfreie Grammatik. Dann existiert eine reduzierte kontextfreie Grammatik $G'=(V',\Sigma,P',S)$ mit

1. $L(G)=L(G')$ (16.1)

2. P‘ enthält nur Produktionen vom Typ

$$\begin{array}{lll} A\to BC & \text{mit} & A,B,C\in V-\Sigma \\ A\to a & \text{mit} & A\in V-\Sigma, a\in\Sigma \\ S\to\varepsilon & \Leftrightarrow & \varepsilon\in L(G') \end{array} \tag{16.2}$$

Diese reduzierte Form, die nur noch einen binären Ableitungsbaum erlaubt, heißt auch „*Chomsky-Normalform*“.

Beweis

Sei $G=(V,\Sigma,P,S)$ reduziert. Man konstruiert *G‘* in mehreren Schritten. Ziel ist es, alle Produktionen, die auf der rechten Seite mehr als zwei Nichtterminale oder mehr als ein Terminal bzw. das leere Wort stehen haben, umzubauen. Hierzu bauen wir neue Produktionsregeln aus den alten (*P*) auf:

$$P_1:=\{A\to a \mid A\to a\in P\} \tag{16.3}$$

diese Menge nimmt also alle diejenigen Regeln aus *P* auf, die gemäß unserer Bedingung 2) ohnehin in *P‘* enthalten sein dürften. Wir definieren nun einen Homomorphismus *h()*, einerseits die Terminale auf neue Nichtterminale (in eckigen Klammern geschrieben) abbildet und andererseits die Nichtterminale ohne Änderung überträgt:

$$\begin{array}{ll} h(a):=[a] & \forall a\in\Sigma \\ h(A):=A & \forall A\in V-\Sigma \end{array} \tag{16.4}$$

Damit bauen wir eine neue Produktionsmenge auf, die alle Produktionen aufnimmt, die in *P* auf der rechten Seite mehr als zwei symbole (Terminale bzw. Nichtterminale) enthielten. Damit haben wir sämtliche Regeln aus *P* abgehandelt.

$$P_2:=\{A\to h(\alpha) \mid A\to\alpha\wedge|\alpha|\geq 2\} \tag{16.5}$$

Wir müssen jetzt P_1 und P_2 noch zusammenfassen und die neuen Nichtterminale in eckigen Klammern zurück auf die ursprünglichen Terminale abbilden:

$$P_3:=P_1\cup P_2\cup\{[a]\to a \mid a\in\Sigma\} \tag{16.6}$$

Wir erhalten nun also offensichtlich $G_3=(V_3,\Sigma,P_3,S)$ mit $L(G)=L(G_3)$. Dabei gilt:

$$V_3=V\cup\{[a] \mid \forall a\in\Sigma\}. \tag{16.7}$$

Nun haben wir in P_2 noch Regeln enthalten, die mehrere Nichtterminale auf der rechten Seite stehen haben. Diese reduzieren wir nun durch Aufblähen unserer Produktionsregelmenge, indem wir jede Regel in mehrere aufsplitten. Dies wird durch das spätere Beispiel noch deutlicher.

Wir ersetzen also jede Produktion aus P_3 mit der Form $A \rightarrow B_1 B_2 \dots B_n$ durch

$$\begin{aligned} &A \rightarrow B_1 \langle B_2 B_3 \dots B_n \rangle \\ &\langle B_i B_{i+1} \dots B_n \rangle \rightarrow B_i \langle B_{i+1} \dots B_n \rangle \quad \forall 2 \le i \le n-2 \\ &\langle B_{n-1} B_n \rangle \rightarrow B_{n-1} B_n \end{aligned} \tag{16.8}$$

Dabei sind die $\langle B_i \dots B_n \rangle$ neue Nichtterminale, die *V'* hinzuzufügen sind. Somit erhält man $G' = (V', \Sigma, P', S)$ mit $L(G) = L(G')$.

Satz:

Sei $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik in Chomsky-Normalform. Dann gilt:

$$\begin{aligned} &\forall n \ge 1 : (|x| = n) \wedge (x \in L(G)) \\ &\Downarrow \\ &S \overset{2n-1}{\underset{G}{\Rightarrow}} x \end{aligned} \tag{16.9}$$

Die Anzahl der Schritte (*2n-1*) läßt sich aus dem folgenden Schaubild ersehen:

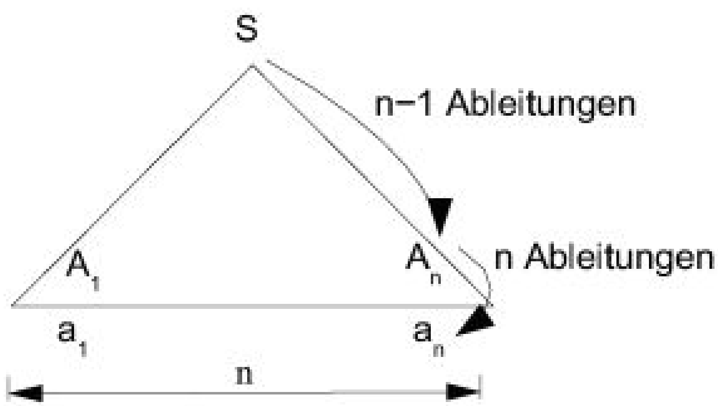

Abbildung 33

Beispiel

Wir wollen die Grammatik $G = (\{S, A\}, \{a, b, c\}, P, S)$ mit $P = \{S \rightarrow aAbb, S \rightarrow aAcb, A \rightarrow ab\}$ auf Chomsky-Normalform bringen und wenden hierzu die aus dem obigen Satz bekannten zwei Schritte an:

P	Schritt 1	Schritt 2	
$S \rightarrow aAbb$	$S \rightarrow [a]A[b][b]$	S	$\rightarrow [a]\langle A[b][b]\rangle$
		$\langle A[b][b]\rangle$	$\rightarrow A\langle [b][b]\rangle$
		$\langle [b][b]\rangle \rightarrow [b][b]$	
$S \rightarrow aAcb$	$S \rightarrow [a]A[c][b]$	S	$\rightarrow [a]\langle A[c][b]\rangle$
		$\langle A[c][b]\rangle$	$\rightarrow A\langle [c][b]\rangle$
		$\langle [c][b]\rangle \rightarrow [c][b]$	

P	Schritt 1	Schritt 2
$A \to ab$	$A \to [a][b]$	$A \to [a][b]$
	$[a] \to a$	$[a] \to a$
	$[b] \to b$	$[b] \to b$
	$[c] \to c$	$[c] \to c$

Damit haben wir eine Grammatik in Chomsky-Normalform mit den in Schritt 2 bezeichneten Produktionen und den Nichtterminalen:

$$V = \left\{ S, A, [a], [b], [c], \left\langle A[b][c] \right\rangle, \left\langle [b][b] \right\rangle, \left\langle A[c][b] \right\rangle, \left\langle [c][b] \right\rangle \right\} \tag{16.10}$$

17 Pushdown-Automaten

17.1 Einführung

Wir kommen nun zu dem Thema der „*Pushdown-Automaten*" („*Kellerautomaten*"). Wie wir später sehen werden, beschreiben kontextfreie Grammatiken und Pushdown-Automaten (*PDA*) dieselben Sprachen.

Im Wesentlichen ist ein PDA ein endlicher Automat, der sowohl über ein Band als auch über einen Stack (Keller) gesteuert wird. Weiter besitzt er eine endliche Kontrolle. Der Stack ist eine Symbolfolge über einem Alphabet. Wir stellen uns den Keller um 90° gekippt vor, so daß das erste Kellersymbol („Top") immer das am weitesten links stehende Symbol ist.

Der Automat kann auf zwei Arten arbeiten: Entweder liest er ein Symbol der Eingabe und kann in Abhängigkeit des aktuellen Zustandes und des obersten Elementes des Stacks den zustand wechseln. Das oberste Element des Stacks wird dann entfernt und eventuell durch ein neues Alphabetsymbol, einer Symbolfolge oder durch das leere Wort ersetzt. Sodann geht der Lesekopf eine Position auf dem Band nach rechts.

In der zweiten Arbeitsweise liest der Automat kein Symbol (eine sogenannte „ε-Bewegung ") vom Eingabeband (der Lesekopf bewegt sich also nicht). Genau wie oben können Zustand und/oder Stack verändert werden.

Bildlich kann man sich das ganze wie in Abbildung 34 vorstellen:

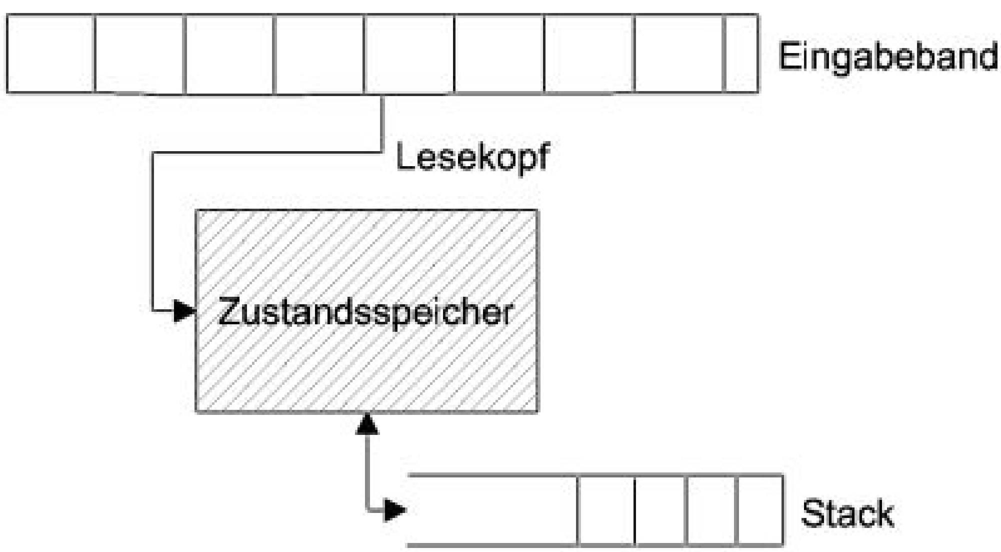

Abbildung 34

17.2 Formale Definition, Konfiguration, Akzeptierung

Definition (Pushdown-Automat)

Ein *Pushdown-Automat* (*PDA*, *Kellerautomat*) ist ein 7-Tupel $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ mit:

1. Q ist eine endliche Menge von Zuständen
2. Σ ist eine endliche Menge von Eingabesymbolen
3. Γ ist eine endliche Menge von Kellersymbolen
4. $q_0 \in Q$ ist der Anfangszustand

5. $Z_0 \in \Gamma$ ist das Anfangskellersymbol (also das Symbol, mit dem der Keller initialisiert wird)
6. $F \subseteq Q$ ist die Menge der akzeptierenden Zustände
7. $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \{A \mid A \subseteq Q \times \Gamma^* \wedge A \text{ endlich}\}$

In Punkt 7 ist die Bedingung „*A* endlich" eingefügt, da $Q \times \Gamma^*$ unendlich sein könnte (wegen der Kleeneschen Hülle).

Wir können zwei Feststellungen machen:

1. Ein PDA ist in der Regel nicht deterministisch. Dies ergibt sich aus der Tatsache, daß δ eine Funktion ist, die auf eine Menge abbildet.
2. Die Stillstandsbewegung des „Lesekopfs" ergibt sich aus der Funktionsverwendung $\delta(q,\varepsilon,A)$. Dies bedeutet, dass das Eingabesymbol nicht gelesen wird und nur eine Zustandsveränderung und/oder Stackveränderung vorgenommen wird.

Definition (Konfiguration)

Sei $M = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ ein PDA. Dann ist $c \in Q \times \Sigma^* \times \Gamma^*$ eine *Konfiguration* von *M*. Die Bedeutung einer solchen Konfiguration $c = (q, ax, A\alpha)$ ist:

- q augenblicklicher Zustand
- ax noch verbleibende Eingabe mit *a* als nächstem Symbol, $a \in \Sigma, x \in \Sigma^*$
- $A\alpha$ Kellerinhalt, mit *A* als oberstem Symbol, $A \in \Gamma, \alpha \in \Gamma^*$

Anmerkung: Eine Konfiguration kann man also als den Gesamtzustand eines PDAs ansehen. Sie beschreibt zu jedem Zeitpunkt genau den Zustand eines PDAs.

Definition

Sei $M = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ ein PDA. Dann definieren wir die folgenden Operationen:

$$(q,ax,A\alpha) \overset{1}{\mapsto} (q',x,u\alpha) \Leftrightarrow (q',u) \in \delta(q,a,A) \tag{17.1}$$

$$(q,ax,A\alpha) \overset{1}{\mapsto} (q',ax,u\alpha) \Leftrightarrow (q',u) \in \delta(q,\varepsilon,A) \tag{17.2}$$

Seien *c* und *c*' zwei beliebige Konfigurationen. Dann gilt:

$$\begin{gathered} c \overset{n}{\mapsto} c' \quad \big| n \geq 1 \\ \Updownarrow \\ \exists c_0 = c, c_1, c_2 \ldots, c_{n-1}, c_n = c' : \quad c_i \overset{1}{\mapsto} c_{i+1} \forall 0 \leq i \leq n-1 \end{gathered} \tag{17.3}$$

Für alle Konfigurationen *c* gilt:

$$c \overset{0}{\mapsto} c \tag{17.4}$$

$$c \overset{*}{\mapsto} c' \Leftrightarrow \exists n \geq 0 : \quad c \overset{n}{\mapsto} c' \tag{17.5}$$

Diese Definition bezieht sich also auf die Übergänge innerhalb des Automaten, die durch die Funktion δ realisiert werden und gibt uns eine einfachere Notation.

Die folgende Definition zeigt uns, dass wir drei Möglichkeiten haben, wie wir die Akzeptierung eines Wortes definieren. Es ist daher immer wichtig, zu wissen, von welcher Akzeptierungsart gesprochen wird.

Definition (Akzeptierungsarten)

Sei $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ ein PDA. Dann existieren drei Möglichkeiten der Akzeptierung:

1. Akzeptierung durch akzeptierende Zustände und leeren Stack:

$$T(M):=\left\{x \middle| x\in\Sigma^* \quad\wedge\quad (q_0,x,Z_0)\overset{*}{\mapsto}(f,\varepsilon,\varepsilon) \quad\wedge\quad f\in F\right\} \tag{17.6}$$

2. Akzeptierung nur durch akzeptierende Zustände:

$$N(M):=\left\{x \middle| x\in\Sigma^* \quad\wedge\quad (q_0,x,Z_0)\overset{*}{\mapsto}(f,\varepsilon,\alpha) \quad\wedge\quad f\in F,\alpha\in\Gamma^*\right\} \tag{17.7}$$

3. Akzeptierung nur durch leeren Stack:

$$L(M):=\left\{x \middle| x\in\Sigma^* \quad\wedge\quad (q_0,x,Z_0)\overset{*}{\mapsto}(q,\varepsilon,\varepsilon) \quad\wedge\quad q\in Q\right\} \tag{17.8}$$

Beispiel eines PDA

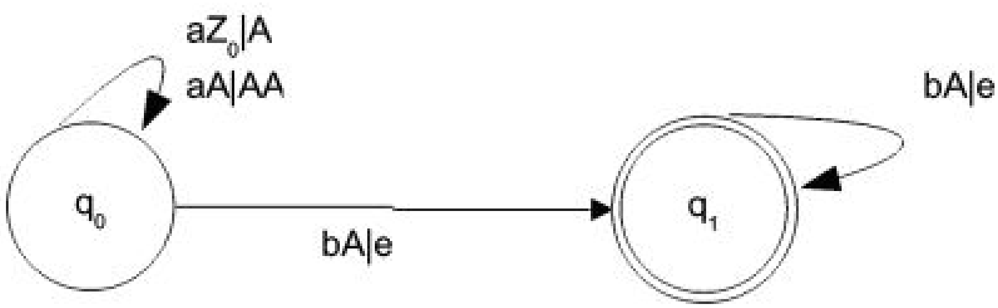

Abbildung 35

Dieser PDA stellt die Sprache $L(M)=\left\{a^nb^n \middle| n\geq 1\right\}=T(M)$ dar. Dabei bedeutet $aZ_0\mid A$, dass bei Lesen des Zeichens a und wenn Z_0 das oberste Element des Stacks ist, Z_0 vom Stack entfernt wird und durch A ersetzt wird.

Würde man die Akzeptierung vom Typ 2) annehmen, so gälte $N(M)=\left\{a^nb^m \middle| n\geq 1, m\leq n\right\}$, da bei dieser Variante bereits akzeptiert wird, wenn nur ein akzeptierender Zustand vorliegt.

In diesem Beispiel realisiert der Stack also eine Art Zähler.

17.3 Äquivalenz von PDA und kontextfreier Grammatik

Satz (Äquivalenz Akzeptierungsarten):

Sei $L\subseteq\Sigma^*$ eine Sprache. Dann sind die folgenden Aussagen gleichwertig:

1. $L=T(M_1)$ für einen PDA M_1
2. $L=T(M_2)$ für einen PDA M_2
3. $L=T(M_3)$ für einen PDA M_3

Beweis

1)⇒2):

Sei $L = T(M_1)$ für $M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_0^1, Z_0^1, F_1)$. Wir definieren einen zweiten PDA:

$$M_2 := (Q_1 \cup \{p\}, \Sigma, \Gamma_1 \cup \{Y\}, \delta_2, q_0^1, Y, \{p\}) \tag{17.9}$$

mit $p \notin Q_1, Y \notin \Gamma_1$. Die Übergangsfunktion definieren wir so:

$$\begin{array}{lll} \delta_2(q_0^1, \varepsilon, Y) & := & \{(q_0^1, Z_0^1, Y)\} \\ \delta_2(f, \varepsilon, Y) & := & \{(p, \varepsilon)\}\ \forall f \in F_1 \\ \delta_2(q, a, A) & := & \delta_1(q, a, A) \qquad q \notin F_1, a \in (\Sigma_1 \cup \varepsilon) \end{array} \tag{17.10}$$

Man kann verifizieren, dass gilt:

$$(q_0^1, x, Z_0^1) \overset{*}{\underset{M_1}{\mapsto}} (f, \varepsilon, \varepsilon) \quad \Leftrightarrow \quad (q_0^1, x, Y) \overset{*}{\underset{M_2}{\mapsto}} (p, \varepsilon, \varepsilon) \tag{17.11}$$

und damit $T(M_1) = N(M_2)$.

Bildlich kann man sich die Konstruktion so vorstellen, dass zu Anfang ein „*Stoppersymbol*“ Y auf den Stack gelegt wird, dann der alte Automat M_1 normal laufen gelassen wird. Wenn dieser ein Wort akzeptiert, dann ist der Stack normalerweise leer. In unserem Falle wird also noch Y auf dem Stack liegen. Wenn das der Fall ist, dann wird in den Zustand p verzweigt, der der einzige akzeptierende Zustand ist. Aufgrund dieses Zustandes akzeptiert nun der Automat M_2 aufgrund von $N(M_2)$ alleinig wegen des Zustandes.

2)⇒3):

Sei $L = N(M_2)$ für $M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_0^2, Z_0^2, F_2)$. Wir definieren einen neuen PDA M_3 wie folgt:

$$M_3 := (Q_2 \cup \{p\}, \Sigma, \Gamma_3, \delta_3, q_0^2, Y, \varnothing) \tag{17.12}$$

mit $p \notin Q_2, Y \notin \Gamma_2$. Wir haben dann $\Gamma_3 := \Gamma_2 \cup \{Y\}$ und definieren die Übergangsfunktion so:

$$\begin{array}{lll} \delta_3(q_0^2, \varepsilon, Y) & := & \{(q_0^2, Z_0^2 Y)\} \\ \delta_3(f, \varepsilon, A) & := & \{(p, \varepsilon)\} \cup \delta_2(f, \varepsilon, A) \quad \forall A \in \Gamma_3, f \in F_2 \\ \delta_3(p, \varepsilon, A) & := & \{(p, \varepsilon)\} \qquad \forall A \in \Gamma_3 \\ \delta_3(q, a, A) & := & \delta_2(q, a, A) \qquad \text{sonst}, \forall a \in \Sigma \cup \{\varepsilon\} \end{array} \tag{17.13}$$

Bei der zweiten Zeile von (17.13) zeigt sich der Nichtdeterminismus. Hier steht $\{(p, \varepsilon)\}$ für den Start des Löschens des Stacks und $\delta_2(f, \varepsilon, A)$ für das Weiterarbeiten in der alten Maschine.

Die Funktionsweise ist die folgende: Solange wir in einem Zustand aus dem alten Automaten M_2 sind, arbeiten wir im alten Automaten (letzte Regel). Sind wir in einem akzeptierenden Zustand des alten Automaten, so löschen wir den Stack (zweite Regel erster Teil bzw. dritte Regel) oder arbeiten im alten Automaten weiter (zweite Regel zweiter Teil), da man auch in einem akzeptierenden Zustand im alten Automaten weiterarbeiten könnte ohne direkt abzubrechen.

Dies alles zeigt uns (was man auch formal beweisen könnte), dass gilt

$$(q_0^2, x, Z_0^2) \overset{*}{\underset{M_2}{\mapsto}} (f, \varepsilon, \alpha) \quad f \in F, \alpha \in \Gamma^* \quad \Leftrightarrow \quad (q_0^2, x, Y) \overset{*}{\underset{M_3}{\mapsto}} (p, \varepsilon, \varepsilon) \tag{17.14}$$

und damit $N(M_2) = L(M_3)$.

3)$\Rightarrow$1):

Sei $L = L(M_3)$ für $M_3 = (Q_3, \Sigma, \Gamma_3, \delta_3, q_0^3, Z_0^3, F_3)$. Dann definieren wir einfach einen Automaten, der die gesamte Zustandsmenge akzeptiert:

$$M_1 := (Q_3, \Sigma, \Gamma_3, \delta_3, q_0^3, Z_0^3, Q_3) \tag{17.15}$$

Damit gilt offensichtlich $L(M_3) = T(M_1)$, da der Automat M_1 mit der Definition von T nur dann ein Wort akzeptiert, wenn der Zustand akzeptiert wird und der Stack leer ist. Da M_3 aber nur dann akzeptiert, wenn der Stack leer ist und in M_1 alle Zustände akzeptieren, gilt dies bereits.

17.3.1 Äquivalenz PDA und kontextfreie Grammatiken

Satz (Äquivalenz PDA/ kontextfreie Grammatik):

Sei $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik in Chomsky-Normalform. Dann existiert ein PDA M, so dass $L(G) = L(M)$.

Beweis:

Wir definieren zunächst einen PDA und zeigen dann, dass das Gesagte gilt. Sei also

$$M := (\{q\}, \Sigma, V, \delta, q, S, \varnothing) \tag{17.16}$$

der PDA. Wir betrachten nun die Definition von $L(M) = \left\{ x \middle| x \in \Sigma^*, (q, x, S) \overset{*}{\mapsto} (q, \varepsilon, \varepsilon) \right\}$. Hieraus folgt die Definition der Übergangsfunktion:

$$\begin{aligned} \delta(q, \varepsilon, A) &:= \{(q, \alpha) | A \to \alpha \in P\} && A \in (V - \Sigma) \\ \delta(q, a, a) &:= \{(q, \varepsilon)\} && a \in \Sigma \end{aligned} \tag{17.17}$$

Mit der ersten Regel können wir das jeweils oberste (linkeste) Stacksymbol durch Produktionen ersetzen und damit den Stack „aufblähen“ (nichtdeterministisch, wie man natürlich an der Mengenklammer sieht). Dabei handelt es sich um reine ε-Bewegungen, d.h. es wird kein Zeichen der Eingabe gelesen, sondern es werden lediglich sämtliche möglichen Produktionen des obersten Stacksymbols, das ein Nichtterminal ist, „durchprobiert“.

Die zweite Regel dient dazu, Terminale wieder vom Stack löschen zu können. Diese Löschen kann man sich wie ein „Matching“ vorstellen, d.h. ein Terminal wurde als „möglich“ erkannt und vom Stack gelöscht. Dabei wird der Lesekopf auf das nächste Eingabesymbol weitergeschoben.

Wir vollziehen den Beweis nun in zwei Schritten. Die beiden Schritte zusammen ergeben den Gesamtbeweis:

$$\begin{aligned} &1. && \forall n \geq 1: A \overset{n}{\Rightarrow} w \quad (w \in \Sigma^*) && \Rightarrow && (q, w, A) \overset{*}{\mapsto} (q, \varepsilon, \varepsilon) \\ &2. && \forall n \geq 1: (q, w, A) \overset{n}{\mapsto} (q, \varepsilon, \varepsilon) && \Rightarrow && A \overset{*}{\Rightarrow} w \end{aligned} \tag{17.18}$$

1. und 2. zusammen ergeben die Aussage, wie wohl offensichtlich ist (1 ist die eine Richtung und 2 die andere).

Beweis von 1:

Wir beweisen per Induktion:

Induktionsverankerung (n=1):

Es sei $A \overset{1}{\Rightarrow} w$. Dann folgt daraus $A \to w \in P$ und $w = a \in \Sigma$ (also dass w ein einzelnes Eingabesymbol ist). Hieraus folgt mit der Definition von δ (Punkt 1)(siehe (17.17)): $(q,a) \in \delta(q,\varepsilon,a)$. Daraus wiederum folgt:

$$(q,a,A) \overset{1}{\mapsto} (q,a,a) \overset{1}{\mapsto} (q,\varepsilon,\varepsilon) \tag{17.19}$$

(der letzte Ableitungsschritt gilt wegen Punkt 2 der Definition von δ (siehe (17.17)).

Induktionsannahme n:

Für alle $m \le n$ gelte die folgende Implikation:

$$A \overset{m}{\Rightarrow} w \quad w \in \Sigma^* \;\Rightarrow\; \quad (q,w,A) \overset{*}{\mapsto} (q,\varepsilon,\varepsilon) \tag{17.20}$$

Induktionsschritt ($n \to n+1$):

Wir haben $A \overset{n+1}{\Rightarrow} w$. Dies ist äquivalent zu $A \overset{1}{\Rightarrow} BC \overset{n}{\Rightarrow} w \quad (n \ge 1)$. Die Einzelableitung gilt, weil wir angenommen haben, dass die kontextfreie Grammatik *G* in Chomsky-Normalform vorliegt. Mit Regel 1 der Definition von δ (siehe (17.17)) folgt:

$$(q,BC) \in \delta(q,\varepsilon,A) \tag{17.21}$$

also $(q,w,A) \overset{1}{\mapsto} (q,w,BC)$. Es gilt weiter $w = w_1 w_2$ mit $B \overset{m_1}{\Rightarrow} w_1$ und $C \overset{m_2}{\Rightarrow} w_2$ mit $m_1, m_2 \le n$. Damit gilt nach zweimaliger Anwendung der Induktionsannahme

$$\begin{aligned} (q,w,BC) \quad &= (q,w_1 w_2 BC) \\ &\overset{*}{\mapsto} (q,w_2,C) \\ &\overset{*}{\mapsto} (q,\varepsilon,\varepsilon) \end{aligned} \tag{17.22}$$

Damit gilt insgesamt

$$\begin{aligned} (q,w,A) \quad &\overset{*}{\mapsto} (q,w,BC) \\ &\overset{*}{\mapsto} (q,\varepsilon,\varepsilon) \end{aligned} \tag{17.23}$$

und damit die Behauptung.

18 Vorlesung vom 8. Juni 2000

18.1 Für jede CFL existiert ein PDA

18.1.1 Fortsetzung des Beweises vom 06. Juni 2000

Der Beweis zur Äquivalenz von PDA und kontextfreier Grammatik wurde in der Vorlesung vom 06. Juni 2000 begonnen, und wird hier fortgeführt.

Es ist zu zeigen:

1: „Hin-Richtung“:

$$\forall n \geq 1 \quad : \quad \left(A \overset{n}{\Rightarrow} w \right) \quad \Rightarrow \quad \left((q, w, A) \overset{*}{\mapsto} q, \varepsilon, \varepsilon \right) \tag{18.1}$$

2: „Rückrichtung“:

$$\forall n \geq 1 \quad : \quad \left((q, w, A) \overset{n}{\mapsto} q, \varepsilon, \varepsilon \right) \quad \Rightarrow \quad \left(A \overset{*}{\Rightarrow} w \right) \tag{18.2}$$

Das Symbol „*A*“ sei hierbei ein Nichtterminal der kontextfreien Grammatik.

Die „Hin-Richtung“ wurde bereits gezeigt. Nun folgt der Beweis der „Rückrichtung“, d.h. wir müssen zeigen, dass jedes Wort, welches unser Automat akzeptiert, auch durch die Grammatik erzeugt werden kann. Diesen Beweis erbringen wir durch vollständige Induktion über die Länge der Produktion *n*.

Wir erinnern uns: Der PDA wurde so konstruiert, dass die Terminalsymbole der kontextfreien Grammatik als Bandalphabet dienen, und sowohl die Terminalsymbole als auch die Nichtterminalsymbole der kontextfreien Grammatik als Stacksymbole dienen.

Induktionsverankerung (n=1):

Für *n=1* gilt:

$$\begin{aligned} &(q, w, A) \overset{1}{\mapsto} (q, \varepsilon, \varepsilon) \\ &\Downarrow \\ &\delta(q, w, A) \ni (q, \varepsilon) \quad \wedge \quad |A| = 1 \end{aligned} \tag{18.3}$$

Bei der Definition unseres Kellerautomaten, dem ja eine kontextfreie Grammatik zugrunde liegt, haben wir einen Zustandsübergang genau dann hinzu genommen, wenn für diese kontextfreie Grammatik mindestens eine von zwei Bedingungen erfüllt ist:

$$\begin{aligned} &\vdots \\ &\Downarrow \\ &\left((w = \varepsilon) \wedge (A \to \varepsilon \in P) \right) \vee (w = A) \end{aligned} \tag{18.4}$$

Das bedeutet in geschriebener Sprache, dass entweder a) kein Symbol vom Band gelesen wird, und ein auf dem Stack liegendes „Nichtterminal“[13] durch das „Terminal“ „leeres Wort“ ersetzt wird, oder b) das vom Band gelesene „Terminal“ dem (einzigen) Zeichen auf dem Stack entspricht, und der Stack daher leergeräumt wird.

Fall a) tritt definitionsgemäß (siehe Definition unseres PDA) genau dann ein, wenn in der kontextfreien Grammatik ein Übergang des Nichtterminales in das Terminal „leeres Wort“ existiert. Da die kontextfreie Grammatik in Chromsky-Normalform ist, kann es sich nur um den Übergang des Startsymboles *S* in das leere Wort handeln. Unser Nichtterminalsymbol *A* muss dann also („von vorne

[13] Man kann bei einem PDA natürlich nicht von „Terminalen“ und „Nichtterminalen“ sprechen. Diese Bezeichnungen beruhen auf der Bedeutung der jeweiligen Symbole in Bezug auf die kontextfreie Grammatik, die dem PDA zugrunde liegt.

herein") das Startsymbol S gewesen sein, und es muss eine Produktionsregel geben, die das Startsymbol S in das leere Wort übergehen lässt.

In diesem Fall handelt es sich um einen ε-Übergang:

$$\begin{aligned}
&\vdots \\
&\Downarrow \\
&(A \to \varepsilon \in P) \wedge (w = \varepsilon) \\
&\Downarrow \quad \text{Chromsky-Normalform} \\
&(S \to \varepsilon \in P) \wedge (w = \varepsilon)
\end{aligned} \tag{18.5}$$

Da es also diese Produktionsregel gibt, gilt:

$$S \overset{1}{\Rightarrow} \varepsilon \quad \text{bzw.} \quad S \overset{*}{\Rightarrow} \varepsilon \tag{18.6}$$

Damit ist für die Produktionslänge *n=1* und den Fall a) gezeigt, dass jedes Wort, dass unser PDA akzeptiert, von der kontextfreien Grammatik erzeugt wird.

Fall b) ist noch einfacher. Fall b) kann nämlich gar nicht eintreten, da das Symbol auf dem Stack (*A*) ein „Nichtterminalsymbol" ist. Demzufolge wird es auf dem Band nie das Zeichen *A* geben, und Fall b) tritt nie ein. Damit ist für die Produktionslänge *n=1* (für Fall a) und Fall b)) gezeigt, dass jedes Wort, dass unser PDA akzeptiert, von der kontextfreien Grammatik erzeugt wird.

Induktionsannahme:

$$\forall m \le n \quad : \quad \left((q,w,A) \overset{m}{\mapsto} q,\varepsilon,\varepsilon \right) \quad \Rightarrow \quad \left(A \overset{*}{\Rightarrow} w \right) \tag{18.7}$$

Induktionsschritt ($n \to n+1$):

$$(q,w,A) \overset{n+1}{\mapsto} (q,\varepsilon,\varepsilon) \tag{18.8}$$

Wir betrachten nun ausdrücklich nur Ableitungen, die eine Länge größer als zwei haben. Dies geschieht aber in Übereinstimmung mit dem Skript zur Vorlesung[14]. Der Fall, dass das „Nichtterminal" *A*, das auf dem Stack liegt, im ersten Zustandsübergang durch ein Terminal ersetzt wird, kann nicht eintreten. Könnte er eintreten, wäre der Stack nach dem zweiten Zustandsübergang bereits leer – und damit die Verarbeitung des PDA beendet; das ist aber ein Widerspruch zu der Annahme, dass mindestens drei Zustandsübergänge erfolgen.

Daher gilt:

$$\begin{aligned}
&(q,w,A) \overset{1}{\mapsto} (q,w,BC) \overset{n}{\mapsto} (q,\varepsilon,\varepsilon) \\
&\Downarrow \\
&\delta(q,\varepsilon,A) \ni (q,BC)
\end{aligned} \tag{18.9}$$

Dieser Übergang des Stacksymboles *A* in die Stacksymbole *B* und *C* kann nur definiert worden sein, wenn die kontextfreie Grammatik die entsprechende Produktion enthält:

$$\begin{aligned}
&\vdots \\
&\Downarrow \\
&A \to BC \in P
\end{aligned} \tag{18.10}$$

Der Ausdruck (18.9) bedeutet nichts anderes, als dass das vorhandene „Nichtterminal" auf dem Stack durch zwei andere „Nichtterminale" ersetzt wird. Je nach Länge des Eingabewortes geschieht dies einige Male. Letzten Endes werden aber alle so auf dem Stack abgelegten „Nichtterminale" wieder „abgebaut", das bedeutet Stück für Stück durch Vergleichen mit dem Eingabewort durch das leere

[14] Dieses Vorgehen ist an sich beweistechnisch nicht korrekt.

Wort ersetzt – bis der Stack irgendwann leer ist. Insbesondere bedeutet das, dass „*C*“ nicht vom Stack gelöscht werden kann, bevor nicht auch „*B*“ vom Stack gelöscht wurde. Und wenn wir einen Schritt weiter denken, und davon ausgehen, dass sowohl *C* als auch *B* auf dem Stack durch mehrere andere „Nichtterminale“ ersetzt werden, bedeutet das trotzdem, dass die „Nichtterminale“, die aus *B* hervorgegangen sind, nicht gelöscht werden können, bevor nicht die „Nichtterminale“ gelöscht wurden, die aus *C* hervorgegangen sind.

Daher lässt sich die Verarbeitung des Eingabewortes *w* wie folgt zerlegen:

$$\begin{aligned} \exists w_1, w_2 \in \Sigma^* \quad : \quad (w_1 w_2 = w) \quad & \wedge \left((q, w_1, B) \overset{m_1}{\mapsto} (q, \varepsilon, \varepsilon) \right) \wedge \\ & \wedge \left((q, w_2, C) \overset{m_2}{\mapsto} (q, \varepsilon, \varepsilon) \right) \end{aligned} \tag{18.11}$$

mit $m_1, m_2 \leq n$ und $m_1 + m_2 = n$. Unter Zuhilfenahme der Induktionsannahme folgt daraus:

$$B \overset{*}{\Rightarrow} w_1 \quad \wedge \quad C \overset{*}{\Rightarrow} w_2 \tag{18.12}$$

Den Übergang von *A* nach *BC* haben wir selber in (18.9) definiert, daher gilt:

$$A \overset{*}{\Rightarrow} BC \overset{*}{\Rightarrow} w_1 C \overset{*}{\Rightarrow} w_1 w_2 \overset{*}{\Rightarrow} w \tag{18.13}$$

Damit ist der Induktionsbeweis abgeschlossen:

$$\left((q, w, A) \overset{n+1}{\mapsto} (q, \varepsilon, \varepsilon) \right) \quad \Rightarrow \quad A \overset{*}{\Rightarrow} w \tag{18.14}$$

Gesamtbeweis:

Wenn wir nun in den (bewiesenen) Behauptungen (18.1) und (18.2) das allgemeine „Nichtterminal“ *A* durch das spezielle Nichtterminal, das Startsymbol *S* ersetzen, erhalten wir:

$$\begin{aligned} & \forall n \geq 1 \quad : \quad \left(S \overset{n}{\Rightarrow} w \right) \quad \Rightarrow \quad \left((q, w, S) \overset{*}{\mapsto} q, \varepsilon, \varepsilon \right), \\ & \forall n \geq 1 \quad : \quad \left((q, w, S) \overset{n}{\mapsto} q, \varepsilon, \varepsilon \right) \quad \Rightarrow \quad \left(S \overset{*}{\Rightarrow} w \right) \\ & \Downarrow \\ & w \in L(M) = w \in L(G) \\ & \Updownarrow \\ & L(M) = L(G) \end{aligned} \tag{18.15}$$

19 Vorlesung vom 15. Juni 2000

19.1 Greibach Normalform (GNF)

19.1.1 Vorbemerkung

Neben der Chomsky Normalform ist auch die Greibach Normalform als weitere standardisierte Darstellungsform für kontextfreie Grammatiken bekannt. Hier sind alle Produktionen derart gestaltet, dass die rechte Seite immer mit einem Terminalsymbol beginnt und keines oder beliebig viele Nichtterminalsymbolen folgen.

$$A \to a B_1 B_2 ... B_k \qquad k \geq 0 \tag{19.1}$$

Um zu zeigen, dass sich jede kontextfreie Grammatik für Sprachen ohne leeres Wort ε in Greibach Normalform bringen lässt, wie wir in den folgenden Abschnitten zeigen werden, muss es möglich sein, äquivalente Grammatiken aufzustellen, bei denen mehrere Ableitungsschritte in einzelnen Produktionen zusammengefasst werden. Dass solche Zusammenfassungen möglich sind, wird im Folgenden gezeigt.

Eine A-Produktion ist eine Produktion der Form

$$A \to \alpha \tag{19.2}$$

mit einem Nichtterminalsymbol A auf der linken Seite. Bei einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ gebe es eine A-Produktion $A \to \alpha_1 B a_2$ mit B-Produktionen

$$B \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s \tag{19.3}$$

Dann gibt es eine aus G hervorgehende kontextfreie Grammatik $G_1 = (V, \Sigma, P_1, S)$ mit neuen Produktionen

$$A \to \alpha_1 \beta_1 \alpha_2 \mid \alpha_1 \beta_2 \alpha_2 \mid \ldots \mid \alpha_1 \beta_s \alpha_2 \tag{19.4}$$

die durch Ableiten von $A \to \alpha_1 B a_2$ aus G mit $B \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s$ entstehen. Die Produktion $A \to \alpha_1 B a_2$ ist dann überflüssig und fehlt in G_1. Dann gilt $L(G) = L(G_1)$.

Beweis:

Der Beweis ist einfach, da sich $L(G_1) \subseteq L(G)$ ergibt, wenn man für jede Ableitung

$$A \underset{G}{\Rightarrow} \alpha_1 B \alpha_2 \underset{G}{\Rightarrow} \alpha_1 \beta \alpha_2 \tag{19.5}$$

in G_1 das entsprechende

$$A \underset{G_1}{\Rightarrow} \alpha_1 \beta \alpha_2 \tag{19.6}$$

verwendet. Umgekehrt ist $L(G) \subseteq L(G_1)$, da $A \to \alpha_1 B a_2$ die einzige Produktion ist, die in G und nicht in G_1 enthalten ist. Immer, wenn $A \to \alpha_1 B a_2$ benötigt wird, folgt in einem späteren Ableitungsschritt $B \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s$, was man in G1 in einem einzigen Ableitungsschritt mit $A \to \alpha_1 \beta_1 \alpha_2 \mid \alpha_1 \beta_2 \alpha_2 \mid \ldots \mid \alpha_1 \beta_s \alpha_2$ realisiert wird. Folglich ist $L(G) = L(G_1)$.

19.1.2 Umwandlung von linksrekursiven in rechtsrekursive Produktionen

Mit der im vorangegangen Abschnitt vorgestellten Möglichkeit, Zusammenfassungen von Ableitungen vorzunehmen, kann man bereits viele Produktionen in Greibach Normalform überführen. Problematisch wird es allerdings bei linksrekursiven Produktionen, die Nichtterminalsymbole auf sich selbst und weitere Symbole abbilden, da diese nicht ohne Weiteres in die gewünschte Form zu bringen sind.

Solche Produktionen der Gestalt

$$A \to A\alpha \qquad A \in V, \alpha \in (V \cup \Sigma)^* \tag{19.7}$$

heißen linksrekursiv. Um sie kompatibel für die Greibach Normalform zu machen, kann man äquivalente rechtsrekursive Produktionen einführen. Gegeben sei eine Menge von A-Produktionen

$$\begin{aligned} &A \to A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_r \quad \alpha_i \in (V \cup \Sigma)^* \\ &A \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s \qquad\qquad\qquad \beta_j \in ((V - A) \cup \Sigma) \cdot (V \cup \Sigma)^* \end{aligned} \tag{19.8}$$

wobei die oberen Produktionen linksrekursiv und die unteren nicht linkrekursiv sind. Damit können die Wörter abgeleitet werden, die durch den regulären Ausdruck

$$\underbrace{(\beta_1 \mid \beta_2 \mid \ldots \mid \beta_s)}_{\text{1 mal}}\underbrace{(\alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_r)^*}_{\text{n mal}} \tag{19.9}$$

beschrieben werden. Dabei handelt es sich um Wörter, die aus einem β_j an erster Stelle und beliebig vielen folgenden α_i bestehen. Diese Wörter lassen sich genauso durch Hinzufügen eines neuen Nichtterminalsymbols B mit Produktionen der folgenden Form erzeugen

$$\begin{aligned} &A \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s \\ &A \to \beta_1 B \mid \beta_2 B \mid \ldots \mid \beta_s B \\ &B \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_r \\ &B \to \alpha_1 B \mid \alpha_2 B \mid \ldots \mid \alpha_r B \end{aligned} \tag{19.10}$$

Damit liegen dann keine linksrekursiven, sondern rechtsrekursive Abbildungen vor, die die gleichen Wörter generieren und die zugehörigen Grammatiken somit die gleiche Sprache beschreiben.

Formaler gesprochen, gibt es zu einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ mit linksrekursiven Produktionen eine entsprechende Grammatik $G_1 = ((V \cup B_k), \Sigma, P_1, S)$ mit rechtsrekursiven Produktionen. Dabei sind alle Produktionen identisch, außer den linksrekursiven, die durch rechtsrekursive ersetzt werden.

G habe nun Produktionen der Form

$$\begin{aligned} &A \to A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_r \quad \alpha_i \in (V \cup \Sigma)^* \\ &A \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_s \qquad\qquad\qquad \beta_j \in ((V - A) \cup \Sigma) \cdot (V \cup \Sigma)^* \end{aligned} \tag{19.11}$$

und G_1 entsprechend zusätzliche

$$\begin{aligned} &A \to \beta_1 B \mid \beta_2 B \mid \ldots \mid \beta_s B \\ &B \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_r \\ &B \to \alpha_1 B \mid \alpha_2 B \mid \ldots \mid \alpha_r B \end{aligned} \tag{19.12}$$

wobei die $A \to A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_r$ in G_1 nicht mehr auftauchen. Dann gilt $L(G) = L(G_1)$.

Die folgende Illustration verdeutlicht die Ableitungsschritte.

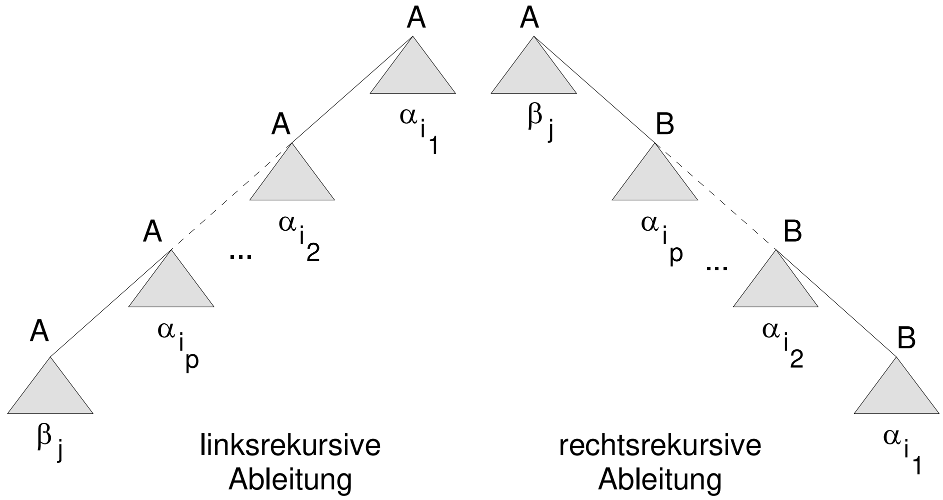

Abbildung 36 - Links- und rechtrekursive Ableitungen

Beweis:

In G endet jede Ableitung von $A \rightarrow A\alpha_i$ mit $A \rightarrow \beta_i$. Damit erhält man

$$A \underset{G}{\Rightarrow} A\alpha_{i_1} \underset{G}{\Rightarrow} A\alpha_{i_2}\alpha_{i_1} \underset{G}{\Rightarrow} \ldots \underset{G}{\Rightarrow} A\alpha_{i_p}\alpha_{i_{p-1}}\ldots\alpha_{i_1} \underset{G}{\Rightarrow} \beta_j\alpha_{i_p}\alpha_{i_{p-1}}\ldots\alpha_{i_1} \tag{19.13}$$

was in G_1 folgendermaßen erzielt wird

$$A \underset{G_1}{\Rightarrow} \beta_j B \underset{G_1}{\Rightarrow} \beta_j\alpha_{i_p} B \underset{G_1}{\Rightarrow} \beta_j\alpha_{i_p}\alpha_{i_{p-1}} B \underset{G_1}{\Rightarrow} \ldots \underset{G_1}{\Rightarrow} \beta_j\alpha_{i_p}\alpha_{i_{p-1}}\ldots\alpha_{i_1} \tag{19.14}$$

Da beide Ableitungen zu den gleichen Wörtern führen, die dem oben genannten regulären Ausdruck entsprechen, gilt $L(G) = L(G_1)$. Somit wurde gezeigt, dass sich linksrekursiven Produktionen unter Zuhilfenahme neuer Nichtterminalsymbols B_k sowie weiterer Produktionen in rechtsrekursive umwandeln lassen und dabei die mit der Grammatik erzeugte Sprache $L(G)$ erhalten bleibt.

19.1.3 Kontextfreie Grammatiken in Greibach Normalform (GNF)

Satz:

Jede kontextfreie Sprache L ohne leeres Wort ε kann durch eine kontextfreie Grammatik G generiert werden, deren Produktionen die Form

$$A \rightarrow aB_1B_2...B_k \qquad k \geq 0 \tag{19.15}$$

haben. A und B_i sind Nichtterminalsymbole, a ist ein Terminalsymbol. Sind alle Produktionen von dieser Form, heißt die Grammatik in Greibach Normalform.

Beweis:

Der Nachweis über die Richtigkeit dieser Aussage wird mit Hilfe eines Algorithmus erbracht, der eine beliebige Grammatik $G_1 = (V_1, \Sigma, P_1, S_1)$, die sich bereits in Chomsky Normalform (CNF) befindet, in eine Grammatik $G = (V, \Sigma, P, S)$ in Greibach Normalform (GNF) umwandelt.

Der Umwandlungsprozess wird in drei Schritten vorgenommen. Zunächst nummerieren wir die Nichtterminalsymbole der Ausgangsgrammatik G_1 mit i = 1...m

$$V_1 = \{A_1, A_2, \ldots, A_m\} \tag{19.16}$$

mit dem Ziel, alle Produktionen in die Form

$$A_i \rightarrow A_j\gamma \qquad i < j \tag{19.17}$$

zu überführen. Dazu durchlaufen wir alle A_i-Produktionen mit $A_i \to A_j\gamma$ und i = 1...m und ersetzen jeweils - wie in der Vorbemerkung gezeigt - die A_j mit allen A_j-Produktionen. Für den Fall, dass eine linksrekursive Produktion $A_i \to A_i\gamma$ auftritt, werden wie im vorherigen Abschnitt beschrieben, ein neues Nichtterminalsymbol und neue Produktionen eingeführt und die linksrekursive in eine rechtsrekursive Produktion umgewandelt.

Am Ende dieses ersten Schrittes haben wir Produktionen erhalten, für die mit $A_i \to A_j\gamma$ i < j gilt. Die Produktionen für A_m sind dann in Greibach Normalform $A_m \to aB_1B_2\ldots B_k$. Algorithmisch betrachtet sieht die Sortierung nach i < j so aus

```
for i := 1 to m do
  for j := 1 to i - 1 do
    foreach Aᵢ -> Aⱼα ∈ P₁ do
      Füge hinzu Aᵢ -> β₁α|β₂α|...|βₛα für alle
         Produktionen Aⱼ -> β₁α|β₂α|...|βₛα ;
      Entferne Aᵢ -> Aⱼα ;
    next
  next j

  if Aᵢ -> Aᵢα ∈ P₁ then
    Füge Nichtterminalsymbol Bᵢ hinzu ;
    Füge hinzu Aᵢ -> β₁Bᵢ|β₂Bᵢ|...|βₛBᵢ für alle
       Produktionen Aⱼ -> β₁|β₂|...|βₛ mit Aᵢ nicht erstem Zeichen von ß₁ ;
    Füge hinzu Bᵢ -> α₁|α₂|...|αᵣ ;
    Füge hinzu Bᵢ -> α₁Bᵢ|α₂Bᵢ|...|αᵣBᵢ ;
    Entferne Aᵢ -> Aᵢα ;
  endif
next i
```

Im zweiten Schritt können wir die so geordneten Produktionen in Greibach Normalform überführen, indem wir sukzessive alle A_i-Produktionen beginnend bei A_m (das ja bereits ein Terminalsymbol als erstes Zeichen auf der rechten Seite und dann wegen der ursprünglichen Chomsky Normalform von G1 folgende Nichtterminalsymbole hat, also in Greibach Normalform ist) in die rechte Seite der A_{i-1}-Produktion einsetzen. Da $A_i \to A_j\gamma$ i < j für alle Produktionen gilt, sind so im Anschluss an diesen Schritt alle Produktionen in Greibach Normalform. Algorithmisch kann das folgendermaßen beschrieben werden

```
for i := m - 1 downto 1 do
  foreach Aᵢ -> Aⱼα ∈ P₁ mit i < j do
    Füge hinzu Aᵢ -> β₁α|β₂α|...|βₛα für alle
       Produktionen Aⱼ -> β₁α|β₂α|...|βₛα ;
    Entferne Aᵢ -> Aⱼα ;
  next
next I
```

Im Anschluss daran befinden sich alle A_i-Produktionen in Greibach Normalform. Bleiben nur noch die möglicherweise hinzugekommenen B_i-Produktionen, deren rechte Seite im dritten und letzten Schritt analog umgeformt werden.

Da die ursprüngliche Grammatik G_1 in Chomsky Normalform ist, besteht die rechte Seite jeder Produktion entweder aus einem Terminalsymbol oder aus zwei Nichtterminalsymbolen. Durch die in den beiden ersten Schritten vorgenommen Umformungen sind deswegen keine Terminalsymbole weiter nach rechts gewandert, sondern können nach wie vor nur an erster Stelle stehen, wie es die Greibach Normalform fordert.

Bei den B_i-Produktionen besteht die rechten Seite ausschließlich aus Nichtterminalsymbolen. Da nach dem zweiten Schritt alle A_i-Produktionen in Greibach Normalform sind, muss im dritten Schritt lediglich das erste Nichtterminalsymbol der rechten Seite einer B_i-Produktion durch die rechte Seite der jeweiligen A_i-Produktion ersetzt werden (siehe Vorbemerkung) und folglich sind dann auch die B_i-Produktionen in der gewünschten Greibach Normalform.

19.1.4 Beispiel 1

Gegeben sei eine Grammatik $G = \left(\{A_1, A_2, A_3\}, \{a, b\}, P, A_1\right)$ mit den zugehörigen Produktionen

$$\begin{aligned} &A_1 \to A_2 A_3 \\ &A_2 \to A_3 A_1 \mid b \\ &A_3 \to A_1 A_2 \mid a \end{aligned} \tag{19.18}$$

Diese wird im Folgenden in Greibach Normalform überführt.

1. Schritt

Das Umwandlungsverfahren in Greibach Normalform beruht auf Ausgangsgrammatiken in Chomsky Normalform. Die hier gegebene Grammatik liegt in CNF vor, da auf der rechten Seite entweder zwei Nichtterminalsymbole oder ein Terminalsymbol stehen. Wir können daher direkt mit der Umwandlung in GNF beginnen.

Die Produktionen müssen alle in die Form $A_i \to A_j \gamma$ mit i < j gebracht und mögliche linksrekursive Produktionen in rechtsrekursive umgewandelt werden. Für die A_1- und A_2-Produktionen ist die Bedingung bereits gegeben, in die A_3-Produktion wird die rechte Seite von A_1 eingesetzt.

Dann erhält man

$$\begin{aligned} &A_1 \to A_2 A_3 \\ &A_2 \to A_3 A_1 \mid b \\ &A_3 \to A_2 A_3 A_2 \mid a \end{aligned} \tag{19.19}$$

und in einem weiteren Schritt wird A_2 in der rechten Seite von A_3 durch A_3A_1 und b ersetzt, so das man folgende Menge von Produktionen erhält

$$\begin{aligned} &A_1 \to A_2 A_3 \\ &A_2 \to A_3 A_1 \mid b \\ &A_3 \to A_3 A_1 A_3 A_2 \mid b A_3 A_2 \mid a \end{aligned} \tag{19.20}$$

Die A_3-Produktion entpuppt sich jetzt als eine linksrekursive Produktion, die durch Hinzufügen eines Nichtterminalsymbols B_3 und weiterer Produktionen entfernt wird.

$$\begin{aligned} &A_1 \to A_2 A_3 \\ &A_2 \to A_3 A_1 \mid b \\ &A_3 \to b A_3 A_2 B_3 \mid a B_3 \mid b A_3 A_2 \mid a \\ &B_3 \to A_1 A_3 A_2 B_3 \mid A_1 A_3 A_2 \end{aligned} \tag{19.21}$$

2. Schritt

Da sich jetzt alle Produktionen der Bedingung $A_i \to A_j \gamma$ mit i < j genügen und $A_m = A_3$ bereits in Greibach Normalform ist, können wir die anderen A_i-Produktionen auch beginnend bei $A_{m-1} = A_2$ in die gewünschte Form umgestalten.

Für A_2 erhalten wir dann

$$\begin{aligned}
&A_1 \to A_2 A_3 \\
&A_2 \to bA_3A_2B_3A_1 \mid aB_3A_1 \mid bA_3A_2A_1 \mid aA_1 \mid b \\
&A_3 \to bA_3A_2B_3 \mid aB_3 \mid bA_3A_2 \mid a \\
&B_3 \to A_1A_3A_2B_3 \mid A_1A_3A_2
\end{aligned} \tag{19.22}$$

und entsprechend für A_1

$$\begin{aligned}
&A_1 \to bA_3A_2B_3A_1A_3 \mid aB_3A_1A_3 \mid bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3 \\
&A_2 \to bA_3A_2B_3A_1 \mid aB_3A_1 \mid bA_3A_2A_1 \mid aA_1 \mid b \\
&A_3 \to bA_3A_2B_3 \mid aB_3 \mid bA_3A_2 \mid a \\
&B_3 \to A_1A_3A_2B_3 \mid A_1A_3A_2
\end{aligned} \tag{19.23}$$

Damit sind alle A_i-Produktionen in Greibach Normalform. Fehlen noch die B_i-Produktionen.

3. Schritt

In das Nichtterminalsymbol der rechten Seite der beiden B_3-Produktionen werden alle fünf A_1-Produktionen eingesetzt, so dass 10 neue Produktionen entstehen.

$$\begin{aligned}
&A_1 \to bA_3A_2B_3A_1A_3 \mid aB_3A_1A_3 \mid bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3 \\
&A_2 \to bA_3A_2B_3A_1 \mid aB_3A_1 \mid bA_3A_2A_1 \mid aA_1 \mid b \\
&A_3 \to bA_3A_2B_3 \mid aB_3 \mid bA_3A_2 \mid a \\
&B_3 \to bA_3A_2B_3A_1A_3A_3A_2B_3 \mid aB_3A_1A_3A_3A_2B_3 \mid bA_3A_2A_1A_3A_3A_2B_3 \\
&B_3 \to aA_1A_3A_3A_2B_3 \mid bA_3A_3A_2B_3 \\
&B_3 \to bA_3A_2B_3A_1A_3A_3A_2 \mid aB_3A_1A_3A_3A_2 \mid bA_3A_2A_1A_3A_3A_2 \\
&B_3 \to aA_1A_3A_3A_2 \mid bA_3A_3A_2
\end{aligned} \tag{19.24}$$

Somit erfüllen alle Produktionen die Bedingung der Greibach Normalform, dass die rechte Seite immer aus einem Terminalsymbol gefolgt von keinem oder beliebig vielen Nichtterminalsymbolen besteht. Zu bemerken ist, dass aus den 5 Produktionen der Grammatik in Chomsky Normalform immerhin 24 Produktionen sowie ein weiteres Nichtterminalsymbol in Greibach Normalform geworden sind.

19.1.5 Beispiel 2

Gegeben sei eine Grammatik $G = (\{A,S\},\{a,b\},P,S)$ mit den zugehörigen Produktionen

$$\begin{aligned}
&S \to AA \mid a \\
&A \to SS \mid b
\end{aligned} \tag{19.25}$$

1. Schritt

Das Umwandlungsverfahren in Greibach Normalform beruht auf Ausgangsgrammatiken in Chomsky Normalform. Die hier gegebene Grammatik liegt in CNF vor, wir können daher direkt beginnen.

Zunächst werden die Nichtterminalsymbole durchnummeriert. Sagen wir, S sei das erste und A das zweite Nichtterminalsymbol, also A_1 = S und A_2 = A, wobei wir aufgrund der Einfachheit auf diese Umbenennung verzichten können und die Reihenfolge im Hintergrund behalten. Dann ist die Zuordnung S → AA in Ordnung, A → SS muss jedoch umgewandelt werden, damit $A_i \to A_j\gamma$ mit i < j für alle Produktionen gilt.

Dadurch erhält man

$$\begin{aligned}
&S \to AA \mid a \\
&A \to AAS \mid aS \mid b
\end{aligned} \tag{19.26}$$

was eine linksrekursive Ableitung erzeugt. Daher führen wir ein neues Nichtterminalsymbol B_2 ein und wandeln in rechtsrekursive Produktion um

$$\begin{aligned} S &\to AA \mid a \\ A &\to aS \mid b \mid aSB_2 \mid bB_2 \\ B_2 &\to AS \mid ASB_2 \end{aligned} \tag{19.27}$$

Dann liegen die A-Produktionen schon in Greibach Normalform vor und wir können zum zweiten Schritt übergehen.

2. Schritt

Jetzt muss das erste A der rechten Seite von S durch die A-Produktionen ersetzt werden und man erhält

$$\begin{aligned} S &\to aSA \mid bA \mid aSB_2A \mid bB_2A \mid a \\ A &\to aS \mid b \mid aSB_2 \mid bB_2 \\ B_2 &\to AS \mid ASB_2 \end{aligned} \tag{19.28}$$

Schön! Jetzt liegen alle Produktion in der gewünschten Form vor, bis auf die B_2-Produktionen, was jedoch auch leicht bewerkstelligt ist.

3. Schritt

Das erste Nichtterminalsymbol der rechten Seite der B_2-Produktionen ist A und wird jeweils durch die beiden A-Produktionen ersetzt, so dass vier neue Produktionen entstehen.

$$\begin{aligned} S &\to aSA \mid bA \mid aSB_2A \mid bB_2A \mid a \\ A &\to aS \mid b \mid aSB_2 \mid bB_2 \\ B_2 &\to aSS \mid bS \mid aSB_2S \mid bB_2S \\ B_2 &\to aSSB_2 \mid bSB_2 \mid aSB_2SB_2 \mid bB_2SB_2 \end{aligned} \tag{19.29}$$

Somit liegen alle Produktionen die Greibach Normalform und die Umwandlung ist abgeschlossen.

20 Vorlesung vom 20. Juni 2000

20.1 Odgens Lemma

Satz:

Sei $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik. Dann existiert eine Konstante $k \geq 1$ (welche im Allgemeinen recht groß ist), so dass für alle Wörter z der Sprache mit $z \in L(G)$ und $|z| \geq k$, in denen mindestens k Positionen markiert sind, gilt:

Das Wort z kann geschrieben werden, als die Zerlegung $z = uvwxy$ mit den folgenden 4 Bedingungen:

1. w enthält mindestens eine markierte Position,
2. u und v enthalten beide markierte Positionen, oder aber x und y enthalten beide markierte Positionen,
3. vwx enthält höchstens k markiere Positionen
4. Es existiert ein Nichtterminal A mit:

$$S \overset{+}{\underset{G}{\Rightarrow}} uAy \overset{+}{\underset{G}{\Rightarrow}} uvAxy \overset{+}{\underset{G}{\Rightarrow}} \ldots \overset{+}{\underset{G}{\Rightarrow}} uv^i Ax^i y \overset{+}{\underset{G}{\Rightarrow}} uv^i wx^i y \quad \forall i \geq 0 \tag{20.1}$$

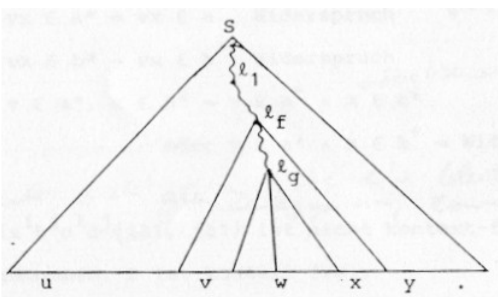

Abbildung 37

Das Bild zeigt, wie man sich dieses am besten vorstellen kann. Es gilt einen Weg vom Startsymbol S aus hin zu den markierten Blättern zu finden. Da nun der Weg sehr lang ist, existiert eine Schliefe. Man betrachtet dann den Teilbaum der letzten Wiederholung. Man sieht nun, dass w zumindest eine markierte Position enthalten muss.

Beweis des Satzes:

Es gelte:

$$\begin{aligned} m &=_{Df} \#(V - \Sigma) \\ l &=_{Df} \max.\left(n \middle| A \to \alpha \in P,\ |\alpha| = n\right) \\ k &=_{Df} l^{2 \cdot m + 3} \end{aligned} \tag{20.2}$$

Es sei dann $z \in L(G)$ mit $|z| \geq k$ und weiterhin mindestens k Positionen in z markiert. Schließlich sei T der Ableitungsbaum für z mit der Länge $\geq 2 \cdot m + 3$. Ein Knoten n heißt Verzweigungsknoten, wenn n mindestens zwei direkte Nachfolger hat. Ein Beispiel wäre, wenn der Knoten n zwei Nachfolger n_1 und n_2 hat, welche beide markierte Blätter unter ihren Nachfolgern haben.

Nun lässt sich ein Weg $n_1, \ldots, n_p$ durch den Ableitungsbaum T wie folgt konstruieren:

1. n_1 ist die Wurzel von T ,