

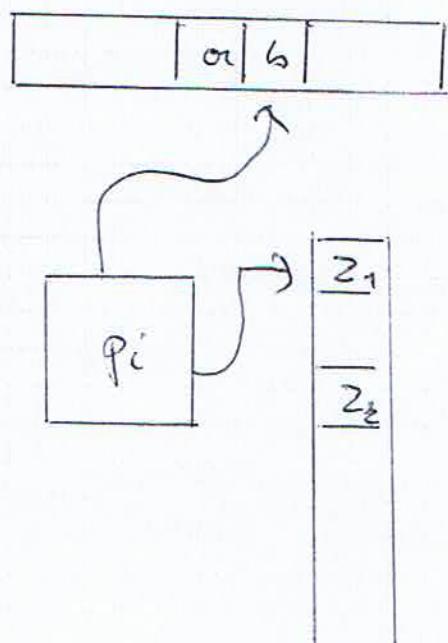
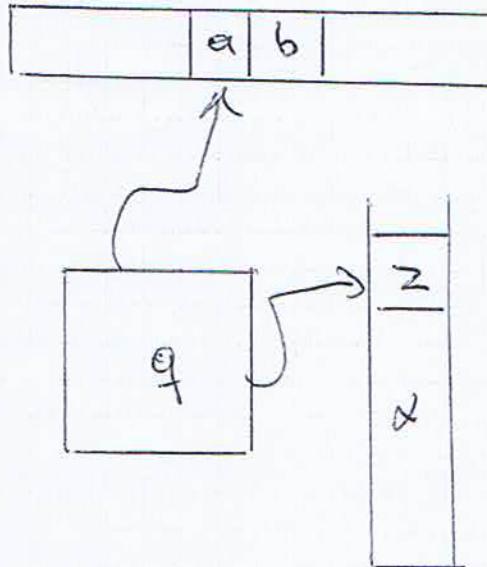
- M04
1. Q eine endliche, nicht leere Menge von Zuständen,
 2. Σ eine endliche, nicht leere Menge von Eingäle symbole,
 3. Γ „ „ „ „ „ „ „ Kellersymbole,
 4. $q_0 \in Q$ der Aufgangs- oder Startzustand,
 5. $z_0 \in \Gamma$ das Startsymbol des Kellers,
 6. $F \subseteq Q$ die Menge der Endzustände und
 7. die Übergangsfunktion δ eine Abbildung
 $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \mapsto 2^{Q \times \Gamma^*}$

simol.

• Interpretation von

$$\delta(q, a, z) = \{(p_1, z_1), (p_2, z_2), \dots, (p_t, z_t)\},$$

wobei $q, p_i \in Q$, $a \in \Sigma$, $z \in \Gamma$, $z_i \in \Gamma^*$.



$$j_i = z_1 z_2 \dots z_\ell$$

Redensart eines NKA.

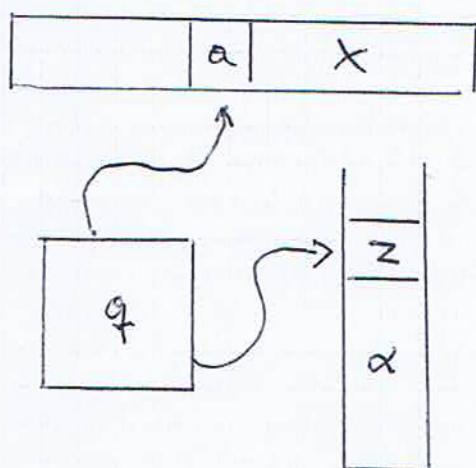
- Interpretation von

$$S(q, \varepsilon, Z) = \{(p_1, z_1), \dots, (p_t, z_t)\}$$

analog.

Bemerkung:

- ΔA ist eine nichtdeterministische Maschine
- δ ist nur für nichtleeren Keller definiert
- Konfiguration Element aus $Q \times \Sigma^* \times \Gamma^*$
- Startkonfiguration $= \{q_0\} \times \Sigma^* \times \{Z_0\}$
- Konfiguration $(q, \alpha x, Zx)$:



- $(q, \varepsilon, Zx) \rightsquigarrow$ Eingabe ist ganz gelesen.
- Übergangsrelation \vdash_M :

$$(q, \alpha x, Zx) \vdash_M (q', x, yx) \quad \text{falls } (q', y) \in \delta(q, a, Z)$$

\vdash_M^* reflexive, transitive Hülle von \vdash_M

$$\cdot L(M) = \left\{ w \in \Sigma^* \mid (q_0, w, z_0) \xrightarrow{M}^* (p, \varepsilon, z) \text{ für ein } p \in F, z \in \Gamma^* \right\}$$

die von M akzeptierte Sprache.

- Ein Kellerautomat $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ heißt deterministisch, falls $\forall a \in \Sigma \cup \{\varepsilon\}, q \in Q, z \in \Gamma$

$$1. |\delta(q, a, z)| \leq 1 \text{ und}$$

$$2. (\exists a \in \Sigma : \delta(q, a, z) \neq \emptyset) \Rightarrow \delta(q, \varepsilon, z) \neq \emptyset$$

Ziel:

Für beliebige gegebene LFG $G = (V, \Sigma, P, S)$

Konstruktion eines KA M_G mit $L(M_G) = L(G)$.

Berechnungen:

Item

$$p := X \rightarrow X_1 X_2 \dots X_{n_p} \in P$$

$$(p, i) \quad 0 \leq i \leq n_p \quad \rightsquigarrow$$

$$[X \rightarrow X_1 X_2 \dots X_i, X_{i+1} \dots X_{n_p}]$$

$It_G = \{(p, i) \mid p \in P, 0 \leq i \leq n_p\}$ Menge
aller Items von G.

Definition von M_G :

$M_G = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, wobei

$$Q = It_G \cup \{[S' \rightarrow .S], [S' \rightarrow S.] \},$$

$$q_0 = [S' \rightarrow .S], F = \{[S' \rightarrow S.] \},$$

$$\Gamma = Q \cup \{\perp\}, Z_0 = \perp \text{ und}$$

$$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$$

Idee:

Definiere δ derart, dass M_G eine Linksausleitungen in G simuliert.

Bsp.:

Linksausleitung:

$$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow \\ aaaSas \Rightarrow aabbbaa$$

$\overbrace{\quad}^{\cong}$

$$S' \Rightarrow .S \Rightarrow .aAS \Rightarrow a.AS \Rightarrow a.SbAS$$

$$\Rightarrow a.aibAS \Rightarrow a a.b AS \Rightarrow a a b . AS$$

$$\Rightarrow a a b . b a S \Rightarrow a a b b . a S \Rightarrow a a b b a . S$$

$$\Rightarrow a a b b a . a \Rightarrow a a b b a a .$$

- Eine Produktion heißt abgearbeitet, wenn der Punkt die rechte Seite einer Produktion ganz passiert ist.

nicht vollständig abgearbeitete Produktionen
 \leadsto Keller.

Arten von Übergängen:

(E) Expandieren

$$\delta([x \rightarrow \beta.A]_j, \varepsilon, z) = \{([A \rightarrow .\alpha], [x \rightarrow \beta.A]_j z) \\ A \rightarrow \alpha \in P\}$$

Die am weitesten links stehende Variable A wird durch eine ihrer Alternativen ersetzt. Der Keller wird expandiert.

(L) Lesen

$$\delta([x \rightarrow \varphi.\alpha\psi], a, z) = \{([x \rightarrow \varphi a.\psi], z)\}$$

(R) Reduzieren nächste Eingabe-Symbol wird gelesen.

$$\delta([x \rightarrow \alpha.], \varepsilon, [w \rightarrow \mu.x.v]) = \\ \{([w \rightarrow \mu x.v], \varepsilon)\}$$

x ist ganz aus X abgeleitet \leadsto

Bearbeitungsplat. kann über x geschlossen werden.

Satz 4.3

106

Seien $G = (V, \Sigma, P, S)$ eine cfg und der zu G assizierte FA $M_G = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ wie oben definiert. Dann gilt $L(M_G) = L(G)$.

Beweis:

Wir zeigen allgemeiner:

$\forall A \in N, w \in \Sigma^*$ gilt genau dann $A \xrightarrow{G} w$, wenn $A \rightarrow \alpha \in P$ existiert, so dass $\forall \gamma \in \Gamma^*$ gilt: $([A \rightarrow \cdot : \alpha], w, \gamma) \vdash_G^* ([A \rightarrow \alpha \cdot], \varepsilon, \gamma)$.

$\stackrel{\alpha}{\Rightarrow}$ Induktion über die Länge der Ableitung von w aus A .

Annahme: $A \xrightarrow{G} w$

\Rightarrow

$w = a_1 a_2 \dots a_k$, $a_i \in \Sigma$, $1 \leq i \leq k$ und $A \rightarrow a_1 a_2 \dots a_k \in P$.

Definition von (L) $\Rightarrow \forall \gamma \in \Gamma^*$:

$([A \rightarrow \cdot a_1 a_2 \dots a_k], a_1 a_2 \dots a_k, \gamma)$

$\vdash_{M_G}^* ([A \rightarrow a_1 a_2 \dots a_k \cdot], \varepsilon, \gamma)$

Sei $m \geq 1$.

Annahme: Beh. gilt für alle Ableitungen der Länge $\leq m$.

Sei $A \xrightarrow[\mathcal{G}]^{m+1} w$.

(107)

\Rightarrow

$w = w_1 w_2 \dots w_k$, $A \xrightarrow[\mathcal{G}]{} x_1 x_2 \dots x_k \xrightarrow[\mathcal{G}]^m w_1 w_2 \dots w_k$

und $x_i \xrightarrow[\mathcal{G}]^{m_i} w_i$, $m_i \leq m$ für $1 \leq i \leq k$.

1. Fall: $x_i \in N$

$\xrightarrow[\text{Ind. Var.}]{} \quad \forall \gamma \in \Gamma^*$.

$\exists \beta_i \in V^*$ mit

$([x_i \rightarrow \beta_i], w_i, \gamma) \vdash_{M_G}^* ([x_i \rightarrow \beta_i], \varepsilon, \gamma)$

\Rightarrow

$([A \rightarrow x_1 \dots x_{i-1} \cdot x_i x_{i+1} \dots x_k], w_i, \gamma)$

$\vdash^{(E)} ([x_i \rightarrow \beta_i], w_i, [A \rightarrow x_1 \dots x_{i-1} \cdot x_i x_{i+1} \dots x_k])$

$\vdash^* ([x_i \rightarrow \beta_i], \varepsilon, [A \rightarrow x_1 \dots x_{i-1} \cdot x_i \dots x_k] \gamma)$

$\vdash^{(R)} ([A \rightarrow x_1 \dots x_i \cdot x_{i+1} \dots x_k], \varepsilon, \gamma)$

2. Fall $x_i \in \Sigma$

$\Rightarrow w_i = x_i$

$\xrightarrow[\text{Def. von } L]{} \quad \forall \gamma \in \Gamma^* :$

$([A \rightarrow x_1 x_2 \dots x_{i-1} \cdot x_i \dots x_e], w_i, \gamma)$

$\vdash^{\text{L}} ([A \rightarrow x_1 \dots x_i \cdot x_{i+1} \dots x_e], \varepsilon, \gamma)$

Insgesamt gilt also $\vdash \gamma \in \Gamma^*$

$([A \rightarrow \cdot x_1 \dots x_e], w_1 w_2 \dots w_e, \gamma)$

$\vdash^* ([A \rightarrow x_1 \dots x_e \cdot], \varepsilon, \gamma).$

\Leftarrow^u analog mittels Induktion über die Anzahl der Rechenschritte.



Die umgekehrte Richtung gilt auch.

Satz 4.4: (ohne Beweis)

Sei $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ ein LKA.

Dann gibt es eine LfG G_M mit $L(G_M) = L(M)$.

Ziel:

Deterministische Simulation von M_G .

Beobachtung:

- Wenn wir den Zustand von M_G immer oben auf den Kellei schreiben, dann verbleibt noch das Problem, den Kellei deterministisch zu ...

1. Ansatz

Gleichzeitige parallele Summation aller möglichen Kellerinhalte

Problem:

Es sind gleichzeitig exponentiell viele verschiedene Kellerinhalte möglich.



Aussetz schreitet an dem zu großen Speicheraufwand und der zu großen Laufzeit.

Beobachtung:

Sowohl CFG G als auch der VA M_G haben eine feste Größe



Zu jedem Zeitpunkt können nur konstant viele verschiedene Items oberstes Kellelement sein.



Es gibt nur konstant viele Möglichkeiten eventuell exponentiell viele Kellerinhalte zu modifizieren



2. Ansatz

- Realisierung aller Kellerainhölte in stark komprimierter Form durch einen gerichteten Graphen.

Durchführung:

- Knoten sind mit Items markiert
- Es gibt genau einen Knoten mit Einenggrad 0. Diese ist mit $[S' \rightarrow, S]$ markiert und heißt Startknoten.
- Knoten mit Ausenggrad 0 heißen Endknoten.
- Zu jedem Zeitpunkt korrespondieren die Pfade vom Startknoten zu einem Endknoten eindeutig zu den möglichen Kellerainhölten.

Ziel: Idee:

Zu jedem Zeitpunkt gleichzeitige Bearbeitung aller Endknoten.

Schnelligkeit:

Arten der durchführbaren Schritte kann für verschiedene Endknoten verschieden sein.

Diese kann wir folgt behoben werden

1. Möglichkeit:

Führe einen Teleschnitt ent über Kürz, wenn dies für alle Endketten in Σ^* ist. Übung

2. Möglichkeit:

Transformiere die grammatische zuerst in eine äquivalente CFG G' , die gewisse Normalform-eigenschaften besitzt, so daß obige Schwierigkeit nicht eintritt.

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

- G ist in Greibach-Normalform (GNF), falls jede Produktion in P von der Form $A \rightarrow \alpha$ mit $\alpha \in \Sigma V^*$ ist.
- Strenger GNF, falls jede Produktion in P von der Form $A \rightarrow \alpha$ mit $\alpha \in \Sigma^N$ ist.
- 2-Standardform (2GNF), falls G in GNF und $|\alpha| \leq 3 \wedge A \rightarrow \alpha \in P$.

Ziel:

Deterministische Simulation des KAs M_G unter der Voraussetzung, daß G in strenger GNF ist.

G in strenger GNF \Rightarrow

Auf Expansionsschritt folgt immer ein Lese-
schritt.



Zusammenfassen von Expansions- und Lese-
schritt



- Beim Expansionsschritt sind nur Alternativen $a\alpha$ zu berücksichtigen, für die α das nächste zu lesende Symbol des Eingangs ist.
- Endknoten des Graphen, die nicht expandiert werden können, da für diese keine überlängige Alternativen existieren, können entfernt werden.
Ein Knoten, dessen sämtliche Nachfolger auf diese Art und Weise entfernt werden, kann selbst entfernt werden.
- Nach dem Expansionsschritt werden solange Reduktionen durchgeführt, wie dies möglich ist.
- Danach nächster Expansionsschritt für alle Endknoten des Graphen.



Wir ordnen die Knoten im Graphen mit dem Zeitpunkt, zu dem sie dem Graphen hinzugefügt werden.

Algorithmus Allgemeine kontextfreie Analyse (ALKFA)

Eingabe: cfg $G_i = (V, \Sigma, P, S)$ in strenger GNF und $w = a_1 a_2 \dots a_n \in \Sigma^*$.

Ausgabe: 1, falls $w \in L(G_i)$ und 0 sonst.

Methode:

(1) Initialisiere den Graphen $\mathcal{G} = (V, E)$ durch

$$V := \{ [S \rightarrow .S]_0 \}; \\ E := \emptyset;$$

(2) $i := 1;$

while $i \leq n$

do

(E) Erweitere alle Endknoten von \mathcal{G} und führe gleichzeitig den korrespondierenden Leseschritt durch. D.h.,

für jeden Endknoten $[A \rightarrow x_1.Bx_2]_i$ und jede Alternative $a; \beta$ für B erweitere V und E durch

$$V := V \cup \{ [B \rightarrow a; \beta]_i \};$$

$$E := E \cup \{([A \rightarrow \alpha_1.B\alpha_2]_e, [B \rightarrow \alpha_i.\beta])\}$$

Entferne alle Endknoten, die nicht expandiert wurden, d.h., einen Index $< i$ lieben.

Entferne solange Knoten, deren Nachfolger alle entfernt wurden, bis kein solcher Knoten mehr existiert.

(R) Reduziere den Graphen \mathcal{G} .

$i := i + 1$

odl;

(3) if $[S^i \rightarrow S_o]_n \in V$

then

Ausgabe := 1

else

Ausgabe := 0

fⁱ.

Durchführung von (R):

- Nach (E) Endknoten, deren Beendigungspunkt hinter der gesuchten rechten Seite der Produktion steht (reduzierbares Knoten).

Dies ist der Fall, wenn die rechte Seite der korrespondierenden Produktion α_i ist.

- Ein reduzierbares Knoten wird aus \mathcal{G} entfernt.

Möglichkeiten für Knoten u während (R):

- Alle seine Nachfolger sind reduzierbar und werden entfernt.

~>

Bearbeitungspunkt des Knotens u wird um eine Position nach rechts verschoben.

- Er hat sowohl Nachfolger, die entfernt werden, als auch Nachfolger, die nicht entfernt werden.

~>

Wir duplizieren u nebst den eingehenden Kanten und schließen den Bearbeitungspunkt des Duplikats um eine Position nach rechts.

- Alle seine Nachfolger werden nicht entfernt.

~>

Tue bzgl. u nichts.

Beobachtung:

Der Knoten u bzw. sein Doppelknoten kann im Fall 1 bzw. im Fall 2 selbst wiederum reduzierbar werden.

Korrektheit:

Zu beweisen ist folgende Behauptung:

Beh.:

Nach jeder Durchführung des Werkes der Wuile-Schleife korrespondieren die Pfade vom Startknoten zu einem Endknoten in G eindeutig zu den möglichen Kellerei-
blättern.

Bew.:

Übung

Aufwandsanalyse:

• Speicherplatz:

begrenzt durch maximale Größe der Graphen
 $G = (V, E)$.

- pro Schleifendurchlauf wird dasselbe Blatt maximal einmal kreiert. Jedes Blatt korrespondiert zu einem Hen.

\Rightarrow

pro Schleifendurchlauf konstant viele Blätter.

$$\Rightarrow |V| = O(n)$$

$$\Rightarrow |E| = O(n^2)$$

• benötigte Zeit:	pro Schleife = Durchlaufzeit	insgesamt
- Initialisierung:		$O(1)$
- Expansion des Blätterzugs	pro Blatt	
zsgl. Leseschritt		$O(n^2)$
- Entfernen eines Knotens	$O(\text{Eingangsgrad})$	$O(n^2)$
- Reduktionsschritt		
topologische Rückwärtstiefensuche		$O(n^2)$
Duplizieren		
direkt		$O(n^2)$
intelligent		$O(n^3)$

Konstante hängt von $|G_1|$ ab.

Satz 4.5

Sei $G_1 = (V, \Sigma, P, S)$ eine cfp in strenger GNF, $w \in \Sigma^*$ und $|w| = n$. Dann kann in $O(n^3)$ Zeit und $O(n^2)$ Platz entschieden werden, ob $w \in L(G_1)$.

- benötigte Zeit, falls G_1 endentig und reduziert ist:

Beobachtung:

Die Durchführung des Reduktionsschrittes ist im allgemeinen Fall so aufwendig, da

während des Rückwärts-Tiefensuche ein Knoten über viele seiner ausgängenden Kanten besucht werden kann.

Falls gezeigt werden kann, dass während der Rückwärts-Tiefensuche jedes Knoten über maximal eine seiner ausgängenden Kanten betreten werden kann, dann reduziert sich die pro Schließendurchlauf für den Reduktionszyklus benötigte Zeit auf $O(n)$, was zu einer Gesamtlaufzeit von $O(n^2)$ führen würde.

Annahme:

In G existiert ein Knoten $[A \rightarrow x.B\beta]_i$, der während des i -ten Reduktionszyklus über zwei verschiedenen ausgängenden Kanten besucht wird.

\Rightarrow

In G existieren zwei verschiedene Linksausleitungen der Linkssetform

$$\alpha_1\alpha_2 \dots \alpha_i \circ \beta$$

G_1 reduziert \Rightarrow

$$\beta \xrightarrow[\alpha]^* w \quad \text{für ein } w \in \Sigma^*$$

\Rightarrow

(119)

es existieren zwei verschiedene Linksaufzüge von $a_1 a_2 \dots a_i w$ in G .

Dies ist ein Widerspruch zur Eindeutigkeit von G .

Satz 4.6.

Sei $G_r = (V, \Sigma, P, S)$ eine eindeutige, reduzierte CFG in strenger GNF, $x \in \Sigma^*$ und $|x| = n$.

Dann kann in Zeit $O(n^2)$ und Platz $O(n^2)$ entschieden werden, ob $x \in L(G_r)$ ist.

4.3 Normalformen für kontextfreie Grammatiken.

Sei $G = (V, \Sigma, P, S)$ eine CFG. Seien $A, B \in N$.

Eine Produktion $A \rightarrow \varepsilon$ heißt ε -Regel. Eine Produktion $A \rightarrow B$ heißt Kettenregel.

Satz 4.7

Sei $G_r = (V, \Sigma, P, S)$ eine CFG. Wir können aus G_r eine CFG $G'_r = (V', \Sigma, P', S')$ mit

1. $L(G'_r) = L(G_r)$,
2. $A \rightarrow \varepsilon \in P' \Leftrightarrow \varepsilon \in L(G_r)$ und $A = S'$ und
3. S' erscheint nicht auf der rechten Seite einer Produktion in P'

konstruieren.

Beweis:

Seien

$$n = |N| \text{ und } W_1 = \{ A \in N \mid A \rightarrow \varepsilon \in P \}.$$

Für $\lambda \geq 1$ sei

$$W_{\lambda+1} = W_\lambda \cup \{ A \in N \mid A \rightarrow \alpha \in P \text{ für ein } \alpha \in W_\lambda^* \}.$$

Eigenschaften:

1. $W_i \subseteq W_{i+1} \quad \forall i \geq 1$.
2. $W_i = W_{i+1} \Rightarrow W_i = W_m \quad \forall m \geq 1$.
3. $W_{n+1} = W_n$.
4. $W_n = \{ A \in N \mid A \xrightarrow[G]{*} \varepsilon \}$.
5. $\varepsilon \in L(G) \Leftrightarrow S \in W_n$.

Definiere

$$G' = (V \cup \{S'\}, \Sigma, P', S') , \text{ wobei}$$

$$P' = \{ S' \rightarrow S \} \cup \{ S' \rightarrow \varepsilon \mid S \in W_n \}$$

$$\cup \{ A \rightarrow A_1 A_2 \dots A_\ell \mid \ell \geq 1, A_i \in V,$$

$$\exists \alpha_1, \dots, \alpha_{\ell+1} \in W_n^*, \text{ so dass}$$

$$A \rightarrow \alpha_1 A_1 \alpha_2 A_2 \dots \alpha_\ell A_\ell \alpha_{\ell+1} \in P \}.$$

Konstruktion \Rightarrow

- S' erscheint nicht auf der rechten Seite einer Produktion.

$$\cdot \varepsilon \in L(G') \Leftrightarrow S' \rightarrow \varepsilon \in P' (\Rightarrow S \in W_n \Leftrightarrow \varepsilon \in L(G))$$

Mittels Induktion über die Länge der Ableitungen
lässt sich $L(G') = L(G)$ beweisen. Übung

Eine CFG, die Satz 4.12 erfüllt, heißt
 ε -frei.

Satz 4.8

Für jede CFG $G = (V, \Sigma, P, S)$ gibt es eine CFG $G' = (V, \Sigma, P', S)$, so dass $L(G') = L(G)$ und P' nur Produktionen der Form

$$S \rightarrow \varepsilon$$

$$A \rightarrow \alpha \quad \alpha \in (V \setminus \{S\})^+, \quad |\alpha| \geq 2$$

$$A \rightarrow a \quad a \in \Sigma$$

enthält.

Beweis:

O.B.d.A. erfülle G bereits Satz 4.12.

Seien für $A \in N$

$$W_0(A) = \{A\} \quad \text{und für } i > 0$$

$$W_{i+1}(A) = W_i(A) \cup \{B \in N \mid C \rightarrow B \in P \text{ für ein } C \in W_i(A)\}$$

Eigenschaften:

1. $W_i(A) \subseteq W_{i+1}(A) \quad \forall i > 0.$
2. $W_i(A) = W_{i+1}(A) \Rightarrow W_i(A) = W_{i+m}(A) \quad \forall m \geq 1$
3. $W_n(A) = W_{n+m}(A)$, wobei $n = |N|$.
4. $W_n(A) = \{B \in N \mid A \xrightarrow[G]{} B\}.$

Definiere

$G' = (V, \Sigma, P', S)$, wobei

$$P' = \bigcup_{A \in N} \left\{ A \rightarrow \alpha \mid \alpha \notin N \text{ und } B \rightarrow \alpha \in P \text{ für ein } B \in W_n(A) \right\}.$$

$L(G') = L(G)$ kann leicht verifiziert werden.



Eine cfg $G = (V, \Sigma, P, S)$ ist in Chomsky-Normalform, falls P nur Produktionen der Form

$$A \rightarrow BC \quad B, C \in N \setminus \{S\}$$

$$A \rightarrow a$$

$$S \rightarrow \epsilon$$

enthält.

Satz 4.9

Für jede cfg $G = (V, \Sigma, P, S)$ existiert eine cfg $G' = (V', \Sigma, P', S)$ in Chomsky-Normalform mit $L(G') = L(G)$.

Beweis:

O. B. d. A erfülle G bereits Satz 4.8

Idee:

(1) Kreiere CFG $\bar{G} = (\bar{V}, \Sigma, \bar{P}, S)$, die nur Produktionen der Form

$$A \rightarrow \alpha \quad \alpha \in (N \cup \Sigma)^* \text{ und } |\alpha| > 2$$

$$A \rightarrow a \quad a \in \Sigma$$

$$S \rightarrow \varepsilon$$

enthält.

(2) Ersetze in \bar{P} jede Produktion $A \rightarrow \alpha$ mit $|\alpha| > 2$ durch $|\alpha|-1$ Produktionen

$$A \rightarrow \alpha_1, A \rightarrow \alpha_2, \dots, A \rightarrow \alpha_{|\alpha|-1}$$

für die $|\alpha_i| = 2$, $1 \leq i \leq |\alpha|-1$.

$G \sim \bar{G}$:

- $\forall a \in \Sigma$ tre
- kreiere neue Hilfsvariable H_a ,
- füge $H_a \rightarrow a$ dem Produktionsystem hinzu und
- ersetze in allen Produktionen, deren rechter Seiten Länge ≥ 2 liegen, das Terminalzeichen a durch H_a .

Konstruktion $\Rightarrow L(\bar{G}) = L(G)$.

$\bar{G} \rightsquigarrow G'$:

(124)

- Ersetze jede Produktion $A \rightarrow B_1 B_2 \dots B_r$, $r > 2$ durch die Produktionen

$$A \rightarrow B_1 C_1, C_1 \rightarrow B_2 C_2, \dots, C_{r-3} \mapsto B_{r-2} C_{r-2}$$

$$\text{und } C_{r-2} \rightarrow B_{r-1} B_r,$$

wobei C_1, C_2, \dots, C_{r-2} neue Symbole sind.

Konstruktion $\Rightarrow L(G') = L(\bar{G})$.

■

Sei $G = (V, \Sigma, P, S)$ eine beliebige ϵ -freie cff.

Ziel:

Entwicklung eines Verfahrens, das G in eine äquivalente cff $G' = (V', \Sigma, P', S)$ in GNF transformiert.

Idee

Transformiere G zunächst in eine cff, deren Produktionen bis auf eventuell vorhandene Kettenregeln die Schreibdr.-Normalformbedingungen erfüllen und eliminiere dann die Kettenregeln.

Eine clg $G = (V, \Sigma, P, S)$ ist in erweiterter GNF, wenn jede Produktion von der Form

$$A \rightarrow a\alpha \quad \text{mit } a \in \Sigma, \alpha \in (V \setminus \{S\})^*,$$

$$A \rightarrow B \quad \text{mit } B \in N \setminus \{S\} \text{ oder}$$

$$S \rightarrow \varepsilon$$

ist.

Ziel:

Gegeben eine beliebige E-regel clg $G = (V, \Sigma, P, S)$, möchten wir Produktionen vom Typ $A \rightarrow B\alpha$, $B \in N \setminus \{S\}$ durch Produktionen, die die Greibach-Normalformbedingungen erfüllen, ersetzen.

Idee:

- Konstruktion einer clg $G_B = (V_B, V, P_B, S_B)$ $\forall B \in N \setminus \{S\}$, so dass

1. G_B ist in erweiterter GNF d.h., für jede Regel $A \rightarrow \alpha \in P_B$ gilt

$$\alpha = af \quad \text{mit } a \in V \text{ oder}$$

$$\alpha \in N_B = V_B \setminus V.$$

2. $S_B \rightarrow \alpha \in P_B \Rightarrow \alpha = af$ mit $a \in \Sigma$ und $f \in (V_B \setminus \{S_B\})^*$.

Bläckle: V ist das Terminalalphabet von G_B .

- Die Idee ist nun eine zu G äquivalente CFG H in erweiterter GNF zu konstruieren, indem wir
 - in G jede Produktion $A \rightarrow B\alpha$, $B \in N \setminus \{S\}$ durch die Menge $\{A \rightarrow \alpha_j \mid S_B \rightarrow \alpha_j \in P_B\}$ von Produktionsmenge ersetzen
 - $\forall B \in N \setminus \{S\}$ die Produktionsmenge

$$\bar{P}_B = P_B \setminus \{S_B \rightarrow \alpha \mid \alpha \text{ ist Alternative von } S_B\}$$
 hinzufügen und

$$A \rightarrow C\alpha \in \bar{P}_B$$
 durch die Produktionsmenge $\{A \rightarrow \alpha_j \alpha \mid S_C \rightarrow \alpha_j \in P_C\}$ ersetzen.

Konstruktion von G_B :

Betrachte Beobachtungen der Form

$$B \Rightarrow \alpha_j \quad \text{oder} \quad B \xrightarrow[\text{em}]{} C\alpha \Rightarrow \alpha_j \alpha,$$

wobei $\alpha \in \Sigma$, $C \in N \setminus \{S\}$ und $\alpha_j, \alpha \in (V \setminus \{S\})^*$.

Beobachtung:

- Bis auf die letzte Ersetzung werden mit Alternativen aus $N(V \setminus \{S\})^*$ gewählt.

- Die letzte Ersetzung wählt für C eine Alternative aus $\Sigma(V \setminus \{S\})^*$

Eine derartige Linksausleitung heißt terminal.
Wir schreiben dann

$$B \xrightarrow[\text{term}]{} \alpha \gamma \quad \text{bzw. } B \xrightarrow[\text{term}]{} \alpha \gamma \alpha.$$

Sei

$$L_B = \{ \alpha \beta \in \Sigma(V \setminus \{S\})^* \mid B \xrightarrow[\text{term}]{} \alpha \beta \}$$

Ziel:

Konstruktion einer CFG $G_B = (V_B, V, P_B, S_B)$,
so dass

1. $L(G_B) = L_B$ und
2. jede Alternative einer Variablen mit einem Symbol in V beginnt oder selbst eine Variable ist.

Sei $N_B = V_B \setminus V$. Zur Konstruktion von P_B betrachte eine terminale Linksausleitung

$$B \Rightarrow D_1 x_1 \Rightarrow D_2 x_2 x_1 \Rightarrow \dots \Rightarrow D_t x_t x_{t-1} \dots x_1 \Rightarrow \alpha_j x_t \dots x_1$$

genauer.

Es gilt

- * $\alpha \in \Sigma$, $D_i \in N \setminus \{S\}$ und $\gamma, \alpha_i \in (V \setminus \{S\})^*$ für $1 \leq i \leq t$.
- * Der korrespondierende Terminalstring in L_B ist dann

$$\alpha \gamma \alpha = \alpha \gamma \alpha_t \dots \alpha_1$$

Für $A \in N$ sei

$$W(A) = \{ C \in N \mid A \xrightarrow[G]{*} C \}$$

D.h., $W(A)$ ist die Menge der Variablen, die aus A unter allmäiger Anwendung von Kettenregeln abgeleitet werden können.

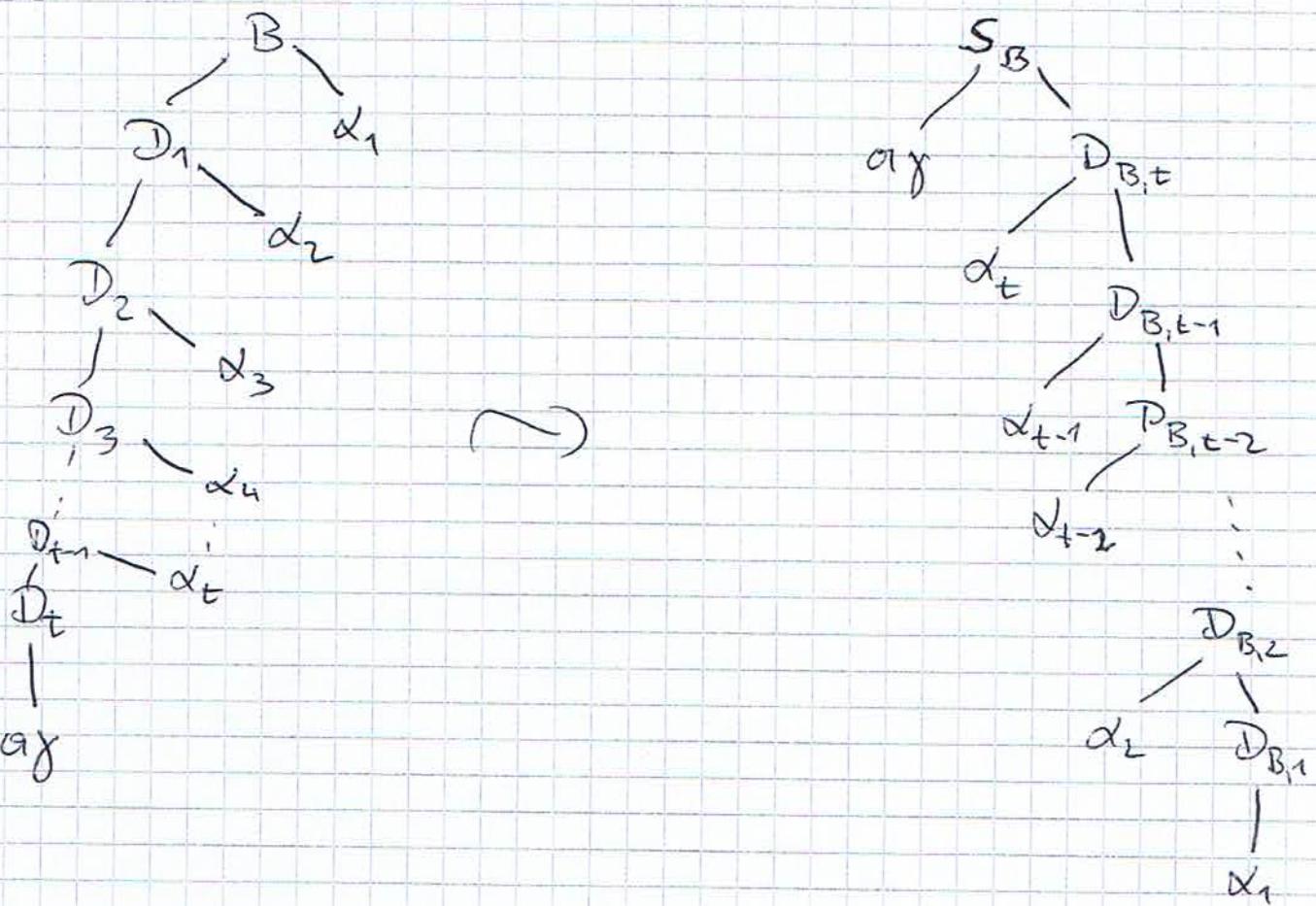
Ziel:

Definition der Produktionen in P_B derart, dass eine terminale Linksschaltung rückwärts durch eine Rechtschaltung summiert wird.

Rechtschaltung zur originalen Linksschaltung:

$$S_B \Rightarrow \alpha \gamma D_{B,t} \Rightarrow \alpha \gamma \alpha_t D_{B,t-1} \Rightarrow \dots \Rightarrow \alpha \gamma \alpha_t \dots \alpha_1 D_{B,1} = \\ \Rightarrow \alpha \gamma \alpha_t \dots \alpha_1$$

Korrespondierenden Ableitungsbäume:



Summierende Rechenableitung enthält drei Arten von Produktionsen:

1. Produktionsen mit dem Startsymbol S_B , die Startproduktionen (\cong Produktionsen in G_1 , deren rechten Seiten mit einem Symbol aus Σ beginnen). auf der linken Seite
2. Produktionsen mit einer Variable aus $N_B \setminus \{S_B\}$ auf der rechten Seite, die inneren Produktionsen (\cong Produktionsen in G_1 , deren rechten Seiten mit Symbol aus $N \setminus \{S\}$ beginnen und die nicht zu finalen Produktionsen korrespondieren)

e) $\forall B, C \in N \setminus \{S\}$, $B \neq C$ gilt $N_B \cap N_C = \emptyset$.

Beweis:

a) Ausgehend von einer beliebigen Ableitung in G_B bzw. in G kann a) leicht durch Konstruktion der entsprechenden Ableitung in der anderen Grammatik G bzw. G_B bewiesen werden.

b) $|G_B| \leq |G|$ folgt direkt aus folgenden Beobachtungen:

- i) Jede Produktion von G , deren rechte Seite mit einem Symbol aus Σ beginnt korrespondiert zu höchstens zwei Startproduktionen von G_B und somit zu einer Produktion.
- ii) Jede andere Produktion korrespondiert zu höchstens einer inneren und höchstens einer finalen Produktion.
- iii) Die Länge einer Startproduktion ist höchstens gleich der Länge der entsprechenden Produktion in G plus 1. Die Länge einer anderen Produktion ist höchstens gleich der Länge der entsprechenden Produktion in G .
- c) - e) folgen direkt aus der Konstruktion.

(13)

3. Produktionen mit keiner Variablen aus $N_B \setminus \{S\}$ auf die rechten Seite, die Produktoren ($\hat{\cong}$ Produktoren, deren linken Seiten in $W(B)$ und deren erstes Symbol der rechten Seite in $N \setminus \{S\}$ sind.)

\sim

$$G_B = (V_B, V, P_B, S_B), \text{ wobei}$$

$$V_B = \{A_B \mid A \in N\} \cup V \quad \text{und}$$

$$P_B = \left\{ \begin{array}{l} S_B \rightarrow \alpha \gamma \mid C \rightarrow \alpha \gamma \in P \text{ für } C \in W(B), \\ \alpha \in \Sigma, \gamma \in V^* \end{array} \right\}$$

$$\cup \left\{ S_B \rightarrow \alpha \gamma C_B \mid C \rightarrow \alpha \gamma \in P, \alpha \in \Sigma, \gamma \in V^* \right\}$$

$$\cup \left\{ C_B \rightarrow \alpha D_B \mid D \rightarrow C \alpha \in P, D \in N \setminus \{S\}, \right. \\ \left. C \in N, \alpha \in V^* \right\}$$

$$\cup \left\{ C_B \rightarrow \alpha \mid D \rightarrow C \alpha \in P, D \in W(B), \right. \\ \left. C \in N, \alpha \in V^+ \right\}$$

Lemma 43

Die Grammatik G_B besitzt folgende Eigenschaften:

- a) $L(G_B) = L_B$.
- b) $|G_B| \leq \sum |G_i|$.
- c) $S_B \rightarrow \alpha \in P_B \Rightarrow \alpha = \alpha \delta \text{ für ein } \delta \in \Sigma$.
- d) G_B ist bzgl. der Terminalalphabet V in erweiterte GNF.

Algorithmus $G \rightsquigarrow H$

Eingabe:cfg $G = (V, \Sigma, P, S)$, $G_B = (V_B, \Sigma, P_B, S_B)$
 $\forall B \in N \setminus \{S\}$.

Ausgabe: zu G äquivalente cfg $H = (V', \Sigma, P', S)$
in erweiterter GNF.

Methode:

(1) $P' := P;$

(2) for alle $B \in N \setminus \{S\}$
do

$$P' := P' \cup P_B$$

od;

(3) Ersetze in P' für alle $B, E \in N \setminus \{S\}$

- jede Produktion $A \rightarrow B\alpha$ durch
 $A \rightarrow S_B\alpha$ und

- jede Produktion $A_E \rightarrow B\alpha$ durch
 $A_E \rightarrow S_B\alpha$;

(4) Ersetze in P' für alle $B, E \in N \setminus \{S\}$

- jede Produktion $A \rightarrow S_B\alpha$ durch
 $\{A \rightarrow \alpha_j \alpha \mid S_B \rightarrow \alpha_j \in P_B\}$ und

- jede Produktion $A_E \rightarrow S_B\alpha$ durch
 $\{A_E \rightarrow \alpha_j \alpha \mid S_B \rightarrow \alpha_j \in P_B\};$

(5) for alle $B \in N \setminus \{S\}$

do

$$P' := P' \setminus \{S_B \rightarrow \alpha \mid S_B \rightarrow \alpha \in P_B\}$$

Lemma 4.4

Die Grammatik $H = (V^1, \Sigma, P^1, S)$ besitzt folgende Eigenschaften:

- $L(H) = L(G_1)$.
 - $|H| = O(|G_1|^3)$
 - H ist in erweiterter GNF.
 - $\forall B \in N \setminus \{S\}$ wurde G_B durch eine äquivalente ctg G'_B der Größe $O(|G_1|^2)$ ersetzt.
 - Falls G_1 keine Kettenregeln enthält, dann ist H bereits in GNF.
-
- Wspkt f) Alle Kettenregeln in P^1 sind von der Form
bei freier $D_E \rightarrow C_E, E \in N \setminus \{S\}$.
Bspkt. falls $\alpha \in N$.

Beweis:

- Schritt (1) und Schritt (2) initialisieren P' , indem sie P und $\forall B \in N \setminus \{S\}$ das Produktionsystem P_B einfügen.
Konstruktionen \Rightarrow Schritt (3) ändert die generierte Sprache nicht. Schritt (4) ersetzt nur einige Variablen durch alle möglichen Alternativen. Schritt (5) entfernt Produktionsketten, die linke Seiten auf keiner rechten Seite einer Produktion vorkommen.
- $\Rightarrow L(H) = L(G_1)$.

b) Beweise, dass $|G_{iB}| = O(|G_i|)$ $\forall i \in N \setminus \{S\}$. (1.34)

Nach der Durchführung von Schritt (3) ist daher die Größe der Grammatik $O(|G_i|^2)$ und somit nach Durchführung von Schritt (4) $O(|G_i|^3)$.

c) Konstruktion $\Rightarrow H$ ist in erweiterter CNF.

d) Da vor der Durchführung von Schritt (4) $\forall i \in N \setminus \{S\}$ $|G_{iB}| = O(|G_i|)$ folgt direkt aus der Konstruktion, dass Schritt (4) G_{iB} durch eine äquivalente G_{iB'} der Größe $O(|G_i|^2)$ ersetzt. diskutiere den schlimmsten Fall.

e) - f) Nur während der Konstruktion von inneren Produktionen, die zu Produktionen $C \rightarrow D \in P$, also zu Kettenregeln in G_i, entsprechen, entstehen Kettenregeln in H. Hieraus folgen direkt e) und f).

Gegebenenfalls

Elimination der Kettenregeln in H unter Verwendung des Standardverfahrens.

~

Da $|N| \leq |G_i|$ und $|G_{iB}'| \leq O(|G_i|^2)$ wird wiederum jede Grammatik G_{iB'} durch eine äquivalente Grammatik G_{iB''} der Größe $O(|G_i|^3)$ ersetzt.

Insgesamt haben wir folgenden Satz bewiesen:

Satz 4.10

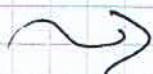
Sei $G = (V, \Sigma, P, S)$ eine beliebige ϵ -freie cfp.
 Dann existiert eine äquivalente cfp $G' = (V, \Sigma, P', S)$ in GNF, so dass $|G'| = O(|G|^3)$, falls G keine Kettenregeln enthält, und $|G'| = O(|G|^4)$ andernfalls.



135a

4.4. Eigenschaften von kontextfreien Sprachen

- gegeben Sprache L . Gesucht sei ein Akzeptor für L .

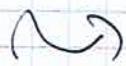


Es ist nötig, Werkzeuge dafür zu haben, zu zeigen, dass eine gegebene Sprache nicht kontextfrei ist.



Pumping-Lemma.

- Konstruktion von cfl mittels Operationen



Abschlusseigenschaften.

Bsp.: (Schnüring)

1350

$$S \rightarrow A$$

$$V = \{ S, A, B, C, a, b \}$$

$$A \rightarrow BC$$

$$B \rightarrow CA \mid b$$

$$C \rightarrow AB \mid a$$

$$G_A = (V_A, V, P_A, S_A)$$

$$V_A = \{ S_A, A_A, B_A, C_A \} \cup V$$

$$P_A = \{ S_A \rightarrow b B_A, S_A \rightarrow a C_A \}$$

$$\cup \{ B_A \rightarrow C_A, C_A \rightarrow A B_A, \\ A_A \rightarrow B C_A \}$$

$$\cup \{ B_A \rightarrow C \}$$

—

$$G_B = (V_B, V, P_B, S_B)$$

$$V_B = \{ S_B, A_B, B_B, C_B \} \cup V$$

$$P_B = \{ S_B \rightarrow b \}$$

$$\cup \{ S_B \rightarrow b B_B, S_B \rightarrow a C_B \}$$

$$\cup \{ B_B \rightarrow C_A, C_B \rightarrow A B_B, \\ A_B \rightarrow B C_B \}$$

$$\cup \{ C_B \rightarrow A \}$$

—

$$G_c = (V_c, V, P_c, S_c)$$

$$V_c = \{S_c, A_c, B_c, C_c\} \cup V$$

$$P_c = \{S_c \rightarrow a\}$$

$$\cup \{S_c \rightarrow bB_c, S_c \rightarrow aC_c\}$$

$$\cup \{B_c \rightarrow CA_c, C_c \rightarrow AB_c, \\ A_c \rightarrow BC_c\}$$

$$\cup \{A_c \rightarrow B\}$$

(1) + (2) ✓

(3) + (4) + (5)

$$S \rightarrow bB_A \mid aC_A$$

$$A \rightarrow bC \mid bB_B C \mid aC_B C$$

$$B \rightarrow aA \mid bB_c A \mid aC_c A \mid b$$

$$C \rightarrow bB_A B \mid aC_A B \mid a$$

$$A_A \rightarrow bC_A \mid bB_B C_A \mid aC_B C_A$$

$$B_A \rightarrow aA_A \mid bB_C A_A \mid aC_C A_A \mid \\ a \mid bB_C \mid aC_C$$

$$C_A \rightarrow bB_A B_A \mid aC_A B_A$$

$$A_B \rightarrow bC_B \mid bB_B C_B \mid aC_B C_B$$

$$B_B \rightarrow aA_B \mid bB_C A_B \mid aC_C A_B$$

$$C_B \rightarrow bB_A B_B \mid aC_A B_B \mid bB_A \mid aC_A$$

$A_C \rightarrow bC_C | bB_B C_C | \alpha C_B C_C | b | bB_b | \alpha C_B$

$B_C \rightarrow \alpha A_C | bB_C A_C | \alpha C_C A_C$

$C_C \rightarrow bB_A C_C | \alpha C_A C_C$

135c

4.4.1 Das Pumping-Lemma für kontextfreie Sprachen.

(136)

Lemma 4.5

Sei L eine beliebige kontextfreie Sprache. Dann gibt es eine Konstante n , die nur von L abhängt, so dass jedes $z \in L$ mit $|z| \geq n$ geschrieben werden kann als $z = uvwxy$ mit

1. $|vwx| \geq 1$,
2. $|vwx| \leq n$ und
3. $uv^i w x^i y \in L \quad \forall i \geq 0$.

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine cfP in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$.

Beobachtung:

- Ableitungsäume bzgl. cfP in CNF besitzen, bis auf die Wurzel von Blättern, nur innere Knoten mit zwei Söhnen.
- Blätteräume mit $\geq 2^k$ Blättern haben Höhe $\geq k$.



Ableitungsäume bzgl. cfP in CNF für Stmgs der Länge 2^k haben Höhe $\geq k+1$.

Setze

$$k = |N| \text{ und } n = 2^k.$$

\Rightarrow

Falls $z \in L(G)$ und $|z| > n$, dann enthält jede Ableitungsbäume T für z einen Pfad P der Länge $> k+1$.

Sei P solch ein Pfad.

\Rightarrow

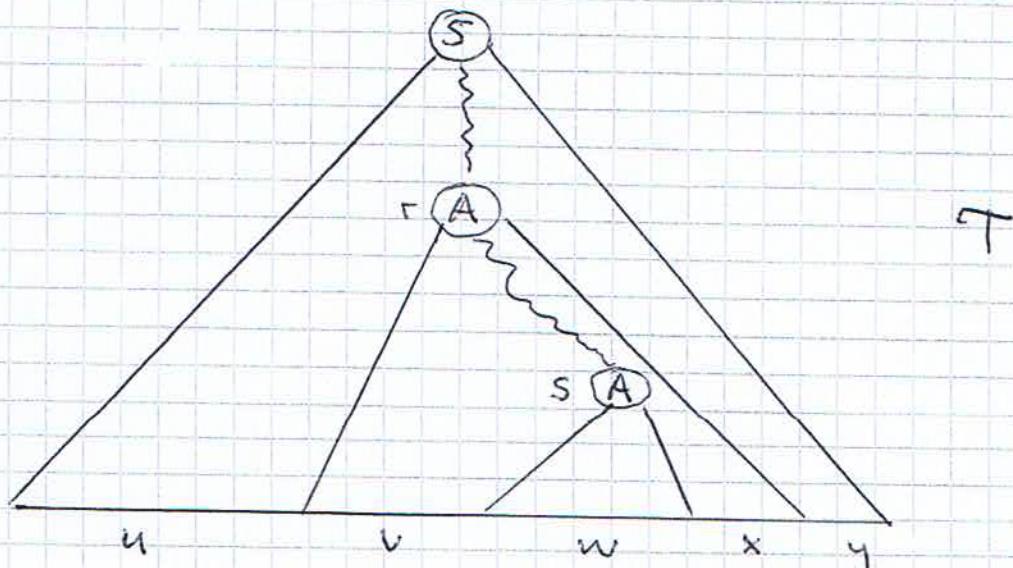
P enthält mindestens $k+2$ Knoten.

Bis auf den letzten Knoten, dem Blatt, enthält alle Knoten Markierungen aus N .

\Rightarrow

Mindestens eine Variable erscheint zweimal als Markierung auf P .

Sind r und s die beiden untersten Knoten auf P mit gleicher Markierung.



G in CNF und die zu Größen r zugehörige
diseante Regel hat die Form $A \rightarrow BC$

$$\Rightarrow |v_x| \neq \epsilon \Rightarrow |v_x| \geq 1.$$

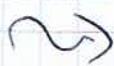
Modifikation von T:

1. Ersetze Teilbaum mit Wurzel r durch Teilbaum mit Wurzel s.



Ablösungsbaum für $uw\bar{y} = uv^0w\bar{x}^0\bar{y}$.

2. Ersetze den Teilbaum mit Wurzel s durch den Teilbaum mit Wurzel r



Ablösungsbaum für $uv^2w\bar{x}^2\bar{y}$.

* (i-1)-malige Ausführung



Ablösungsbaum für $uv^iw\bar{x}^i\bar{y}$.

Bsp.:

Beachte

$$L = \{a^i b^j c^i d^j \mid i \geq 1 \text{ und } j \geq 1\}.$$

Annahme: L ist kontextfrei.

Sei n die Konstante des Pumping-Lemmas. 13.

Betrachte

$$z = a^n b^n c^n d^n$$

Pumping-Lemma \Rightarrow

Wir können schreiben

$$z = uvwxy \text{ mit } |vwx| \leq n \text{ und } vx \neq \epsilon$$



vwx ist Teilstring mit nur einem Symboltyp oder

vwx " " " exakt zwei Symboltypen.

1. Fall: ein Symboltyp



uvw hat jeweils n von drei verschiedenen Symbolen und $< n$ vom vierten Symboltyp

$\Rightarrow uvwy \notin L$

2. Fall: zwei Symboltypen

z.B. b 's und c 's. (andere Fälle analog)

\Rightarrow

In uwv fehlen

- einige b's oder
- einige c's oder
- beides.

Annahme: Es fehlen einige b's

\Rightarrow

Der uwv n d's enthält:

uwv $\notin L$

Annahme: Es fehlen einige c's

\Rightarrow

Der uwv n a's enthält

uwv $\notin L$

4.4.2 Abschlußeigenschaften:

Satz 4.11

Die Klasse der kontextfreien Sprachen ist unter Vereinigung, Konkatenation und Kleene-Abschluß abgeschlossen.

Beweis:

Seien L_1 und L_2 kontextfreie Sprachen, die durch die CFG $G_1 = (V_1, \Sigma_1, P_1, S_1)$ und