

# Das Leerheitsproblem

## Definition

Das Leerheitsproblem für CFG:

- Eingabe: Eine CFG  $G = (N, T, P, S)$
- Frage: Ist  $L(G) = \emptyset$ ?

## Theorem

*Das Leerheitsproblem für CFG läßt sich in polynomieller Zeit lösen.*

## Beweis.

Es ist  $L(G) = \emptyset$  gdw.  $S \notin pre^*(T^*)$ . □

# Unerreichbare Symbole

## Definition

Finden von unerreichbaren Symbolen einer CFG:

- Eingabe: Eine CFG  $G = (N, T, P, S)$
- Ausgabe: Eine Liste der unerreichbaren Symbole von  $G$ :  
 $\{ A \in N \mid \text{es gibt kein } \alpha A \beta \in (N \cup T)^* \text{ mit } S \xRightarrow{*} \alpha A \beta \}$

## Theorem

*Unerreichbare Symbole einer CFG lassen sich in polynomieller Zeit finden.*

## Beweis.

$A$  unerreichbar gdw.  $S \notin \text{pre}^*((N \cup T)^* A (N \cup T)^*)$  □

# Nullierbare Symbole

## Definition

Finden von nullierbaren Symbolen einer CFG:

- Eingabe: Eine CFG  $G = (N, T, P, S)$
- Ausgabe: Eine Liste der nullierbaren Symbole von  $G$ :  
 $\{ A \in N \mid A \xRightarrow{*} \epsilon \}$

## Theorem

*Nullierbare Symbole einer CFG lassen sich in polynomieller Zeit finden.*

## Beweis.

Die Menge der nullierbaren Symbole ist  $N \cap pre^*(\{\epsilon\})$ . □

Später: Wie entfernt man nullierbare Symbole?

# Das Endlichkeitsproblem für CFG

## Definition

Das Endlichkeitsproblem für CFG:

- Eingabe: Eine CFG  $G = (N, T, P, S)$
- Frage: Ist  $L(G)$  endlich?

## Theorem

*Das Endlichkeitsproblem für CFG läßt sich in polynomieller Zeit lösen.*

## Beweis.

Ersetze  $G$  durch  $G'$  mit  $L(G') = L(G) \setminus \{\epsilon\}$ , aber  $G'$  enthält keine unproduktiven, unerreichbare oder nullierbare Symbole.

$|L(G)| = \infty$  gdw. es gibt  $A \in N$  mit

$A \in pre_{G'}^* \left( (N \cup T)^+ A (N \cup T)^* \cup (N \cup T)^* A (N \cup T)^+ \right)$ .



# Das Schnittleerheitsproblem für CFG

## Definition

Das Schnittleerheitsproblem für CFG:

- Eingabe: Zwei CFG  $G_1$  und  $G_2$
- Frage: Ist  $L(G_1) \cap L(G_2) = \emptyset$ ?

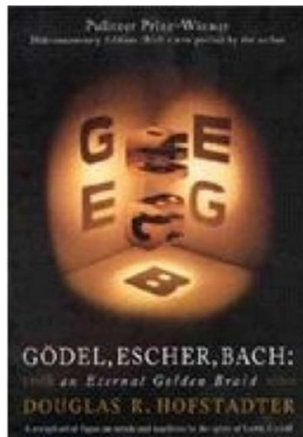
## Theorem

*Das Schnittleerheitsproblem für CFG ist „nicht berechenbar“.*

## Beweis.

Idee: Gäbe es einen Algorithmus für dieses Problem, dann wäre auch das Post'sche Korrespondenzproblem lösbar. □

# Weitere Buchempfehlung



# Das Post'sche Korrespondenzproblem

Gegeben ist eine Menge von Karten dieser Form:

10	0	001
0	001	1

Frage:

Kann ich Karten dieser Art so aneinanderlegen, daß oben und unten dasselbe Wort entsteht?

(Jede Karte kann beliebig oft verwendet werden)

In *Berechenbarkeit und Komplexität* wird folgendes bewiesen:

## Theorem

*Das Post'sche Korrespondenzproblem ist nicht entscheidbar.*

## Theorem

*Das Schnittleerheitsproblem für CFG ist nicht entscheidbar.*

## Beweis.

Es sei

$$I := \left\{ \begin{array}{|c|} \hline u_1 \\ \hline \\ \hline v_1 \\ \hline \end{array}, \begin{array}{|c|} \hline u_2 \\ \hline \\ \hline v_2 \\ \hline \end{array}, \dots, \begin{array}{|c|} \hline u_n \\ \hline \\ \hline v_n \\ \hline \end{array} \right\}$$

eine PCP-Instanz.

Konstruiere zwei Grammatiken:

$$G_1: S \rightarrow u_1 S v_1^R \mid \dots \mid u_n S v_n^R \mid \$$$

$$G_2: S \rightarrow 0S0 \mid 1S1 \mid \$$$

Behauptung:  $L(G_1) \cap L(G_2) \neq \emptyset$  gdw.  $I$  lösbar.





# Das Eindeutigkeitsproblem für CFG

## Definition

Das Eindeutigkeitsproblem für CFG:

- Eingabe: Eine CFG  $G$
- Frage: Ist  $G$  eine eindeutige CFG?

## Theorem

*Das Eindeutigkeitsproblem für CFG ist nicht berechenbar.*

## Beweis.

Idee: Gäbe es einen Algorithmus für dieses Problem, dann wäre auch das Post'sche Korrespondenzproblem lösbar. □

## Beweis.

Es sei

$$I := \left\{ \begin{array}{|c|} \hline u_1 \\ \hline \\ \hline v_1 \\ \hline \end{array}, \begin{array}{|c|} \hline u_2 \\ \hline \\ \hline v_2 \\ \hline \end{array}, \dots, \begin{array}{|c|} \hline u_n \\ \hline \\ \hline v_n \\ \hline \end{array} \right\}$$

eine PCP-Instanz.

$$S \rightarrow A \mid B$$

$$A \rightarrow u_1 A x_1 \mid u_2 A x_2 \mid \dots \mid u_n A x_n \mid u_1 x_1 \mid u_2 x_2 \mid \dots \mid u_n x_n$$

$$B \rightarrow v_1 B x_1 \mid v_2 B x_2 \mid \dots \mid v_n B x_n \mid v_1 x_1 \mid v_2 x_2 \mid \dots \mid v_n x_n$$

Behauptung:

Die Grammatik ist eindeutig gdw.  $I$  keine Lösung besitzt. □

# Das Universalitätsproblem für CFG

## Definition

Das Universalitätsproblem für CFG:

- Eingabe: Eine CFG  $G = (N, T, P, S)$
- Frage: Ist  $L(G) = T^*$ ?

## Theorem

*Das Universalitätsproblem für CFG ist nicht berechenbar.*

## Beweis.

Idee: Post'sches Korrespondenzproblem. □

## Beweis.

$$I := \left\{ \begin{array}{|c|} \hline u_1 \\ \hline \\ \hline v_1 \\ \hline \end{array}, \begin{array}{|c|} \hline u_2 \\ \hline \\ \hline v_2 \\ \hline \end{array}, \dots, \begin{array}{|c|} \hline u_n \\ \hline \\ \hline v_n \\ \hline \end{array} \right\}$$

- $L_1 = \{ h^{-1}(w\$w^R) \mid w \in \{0, 1\}^* \}$ ,  
 $h: 0 \mapsto 0, 1 \mapsto 1, \dot{\cdot} \mapsto \epsilon, \$ \mapsto \$$
- $L_2 = \{ u_{i_1} \dot{\cdot} \dots \dot{\cdot} u_{i_k} \$ v_{i_k}^R \dot{\cdot} \dots \dot{\cdot} v_{i_1}^R \mid 1 \leq i_1, \dots, i_k \leq n \}$

Technisch aufwendig, aber gar nicht so schwer:

$\{0, 1, \dot{\cdot}, \$\}^* \setminus L_1$  und  $\{0, 1, \dot{\cdot}, \$\}^* \setminus L_2$  sind kontextfreie Sprachen.

Dann auch  $L := \{0, 1, \dot{\cdot}, \$\}^* \setminus (L_1 \cap L_2)$  eine CFL.

Es gilt aber  $L = \{0, 1, \dot{\cdot}, \$\}^*$  gdw.  $I$  keine Lösung hat. □

# Das Sprachäquivalenzproblem für CFG

## Definition

Das Sprachäquivalenzproblem für CFG:

- Eingabe: Zwei CFGs  $G_1$  und  $G_2$
- Frage: Ist  $L(G_1) = L(G_2)$ ?

## Theorem

*Das Sprachäquivalenzproblem für CFG ist nicht berechenbar.*

## Beweis.

Gegeben  $G_1 = (N, T, P, S)$ . Konstruiere  $G_2$  mit  $L(G_2) = T^*$ .

$L(G_1) = L(G_2)$  gdw.  $L(G_1) = T^*$ .

Wäre das Sprachäquivalenzproblem berechenbar, dann wäre auch das Universalitätsproblem berechenbar. □

# Das Inklusionsproblem für CFG

## Definition

Das Inklusionsproblem für CFG:

- Eingabe: Zwei CFGs  $G_1$  und  $G_2$
- Frage: Ist  $L(G_1) \subseteq L(G_2)$ ?

## Theorem

*Das Inklusionsproblem für CFG ist nicht berechenbar.*

## Beweis.

Gegeben  $G_1 = (N, T, P, S)$ . Konstruiere  $G_2$  mit  $L(G_2) = T^*$ .

$L(G_2) \subseteq L(G_1)$  gdw.  $L(G_1) = T^*$ .

Wäre das Inklusionsproblem berechenbar, dann wäre auch das Universalitätsproblem berechenbar. □

# Entfernen von $\epsilon$ -Produktionen

## Theorem

*Es sei  $G = (N, T, P, S)$  eine CFG mit  $\epsilon \notin L(G)$ .*

*Dann gibt es eine CFG  $G'$  ohne  $\epsilon$ -Produktionen, die dieselbe Sprache wie  $G$  erzeugt.*

## Beweis.

Idee:

- 1 Finde alle nullierbaren Symbole  $Z \subseteq N$ .
- 2 Für jede Produktion  $A \rightarrow \alpha B \gamma$  mit  $B \in Z$ , erzeuge zusätzlich eine neue Produktion  $A \rightarrow \alpha \gamma$  (wiederhole dies rekursiv).
- 3 Streiche alle Produktionen der Form  $A \rightarrow \epsilon$ .



# Beispiel

$$S \rightarrow AaS \mid b$$

$$A \rightarrow Sb \mid aAA \mid \epsilon$$

$$B \rightarrow AA \mid AB \mid BAa \mid b$$

Welche Symbole sind nullierbar?

$A$  und  $B$

---

$$S \rightarrow AaS \mid aS \mid b$$

$$A \rightarrow Sb \mid aAA \mid aA \mid a$$

$$B \rightarrow AA \mid AB \mid A \mid B \mid BAa \mid Ba \mid Aa \mid a \mid b$$



## Definition

Eine Grammatik  $G = (N, T, P, S)$  ist in *Chomsky-Normalform*, wenn sie nur Produktionen folgender Form enthält:

$$A \rightarrow a \text{ und } A \rightarrow BC,$$

wobei  $a \in T$  und  $A, B, C \in N$ .

Beispiel:

$$S \rightarrow R_a A \mid R_b B$$

$$A \rightarrow S R_a \mid R_a R_a \mid R_b R_a$$

$$B \rightarrow S R_b \mid R_a R_b \mid R_b R_b$$

$$R_a \rightarrow a$$

$$R_b \rightarrow b$$

# Transformation in CNF

- ①  $\epsilon$ -Produktionen entfernen
- ② Neues Nonterminal  $R_a$  für jedes  $a \in T$
- ③ Ersetze  $A \rightarrow aBbC$  durch  $A \rightarrow R_aBR_bC$
- ④ Neue Regel  $R_a \rightarrow a$  für jedes  $a \in T$
- ⑤ Ersetze  $A \rightarrow B_1B_2B_3 \cdots B_k$  durch  $A \rightarrow B_1B_{23\dots k}$  (neues Symbol)
  - $B_{23\dots k} \rightarrow B_2B_{34\dots k}$
  - $B_{34\dots k} \rightarrow B_3B_{45\dots k}$
  - ...
  - $B_{k-1,k} \rightarrow B_{k-1}B_k$

Jetzt fast in CNF. Aber es gibt noch „Kettenregeln“  $A \rightarrow B$ .

# Beispiel

$$S \rightarrow aSa \mid bSb \mid a \mid b$$

$$S \rightarrow R_aSR_a \mid R_bSR_b \mid R_a \mid R_b$$

$$R_a \rightarrow a$$

$$R_b \rightarrow b$$

$$S \rightarrow R_aA$$

$$S \rightarrow R_bB$$

$$A \rightarrow SR_a$$

$$B \rightarrow SR_b$$

$$R_a \rightarrow a$$

$$S \rightarrow R_a$$

$$R_b \rightarrow b$$

$$S \rightarrow R_b$$

# Kettenregeln

So können wir Kettenregeln eliminieren:

Falls die Regel  $A \rightarrow B$  existiert, dann füge für jede Produktion  $C \rightarrow \alpha A \beta$  eine Produktion  $C \rightarrow \alpha B \beta$  hinzu.

Füge zu  $S \rightarrow A$  die Produktionen  $S \rightarrow \beta$  mit  $A \rightarrow \beta \in P$  hinzu.

Dann streiche alle Kettenregeln.

## Theorem

*Es gibt einen Algorithmus, der zu einer Grammatik  $G$  eine Grammatik  $G'$  in Chomsky-Normalform berechnet, wobei  $L(G') = L(G) \setminus \{\epsilon\}$ .*

# Beispiel

$$S \rightarrow R_a A$$

$$S \rightarrow R_b B$$

$$A \rightarrow S R_a$$

$$B \rightarrow S R_b$$

$$R_a \rightarrow a$$

$$S \rightarrow R_a$$

$$R_b \rightarrow b$$

$$S \rightarrow R_b$$

$$S \rightarrow R_a A \mid R_b B \mid a \mid b$$

$$A \rightarrow S R_a \mid R_a R_a \mid R_b R_a$$

$$B \rightarrow S R_b \mid R_a R_b \mid R_b R_b$$

$$R_a \rightarrow a$$

$$R_b \rightarrow b$$

# Linksrekursion

## Definition

Eine Regel  $A \rightarrow A\alpha$  ist *linksrekursiv*.

Linksrekursion läßt sich eliminieren:

Für  $A \in N$  seien

- $A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_k$  die linksrekursiven und
- $A \rightarrow \beta_1 \mid \dots \mid \beta_l$  die anderen Regeln.

Ersetze  $A \rightarrow A\alpha_1 \mid \dots \mid A\alpha_k$  durch

- ①  $A \rightarrow \beta_1 Z \mid \dots \mid \beta_l Z$  und
- ②  $Z \rightarrow \alpha_1 Z \mid \dots \mid \alpha_k Z \mid \alpha_1 \mid \dots \mid \alpha_k$ ,

wobei  $Z$  ein neues Symbol ist.

# Beispiel

$$S \rightarrow SS \mid AS \mid SB$$

$$S \rightarrow AS \mid ASZ$$

$$Z \rightarrow S \mid B \mid SZ \mid BZ$$

# Greibach-Normalform

## Definition

Eine CFG ist in *Greibach-Normalform* (GNF) falls jede Regel von der Form  $A \rightarrow aBCD \dots$  ist, d.h.  $A \rightarrow \beta$  mit  $\beta \in TN^*$ .

## Theorem

Es sei  $G$  ein CFG mit  $\epsilon \notin L(G)$ . Dann gibt es eine CFG  $G'$  in GNF mit  $L(G') = L(G)$ .



Starte mit  $G$  in CNF ohne unnütze und nullierbare Symbole.

O.b.d.A. sei  $N = \{A_1, \dots, A_n\}$ .

1. Schritt:

Ändere  $G$  so, daß es keine Regeln  $A_j \rightarrow A_j \alpha$  mit  $j \leq i$  gibt.

Nehmen wir an, dies gilt schon für  $i = 1, \dots, k$ .

Sei  $l$  die kleinste Zahl, für die es eine Regel

$A_{k+1} \rightarrow A_l \alpha$  gibt.

Es seien  $A_l \rightarrow \beta_1 \mid \dots \mid \beta_m$  alle Regeln für  $A_l$ .

Ersetze  $A_{k+1} \rightarrow A_l \alpha$  durch  $A_{k+1} \rightarrow \beta_1 \alpha \mid \dots \mid \beta_m \alpha$ .

Wiederhole dies, bis es keine Regel

$A_{k+1} \rightarrow A_l \alpha$  mit  $l < k + 1$  mehr gibt.

Jetzt kann es noch Regeln  $A_{k+1} \rightarrow A_{k+1}\alpha$  geben, die wir mit Hilfe eines neuen Symbols eliminieren (das größer ist).

Alle Regeln haben jetzt die Form  $A_i \rightarrow A_j\alpha$  mit  $j > i$  oder  $A_i \rightarrow a\alpha$ .

Wir ersetzen  $A_i \rightarrow A_j\alpha$  durch  $A_i \rightarrow \beta_1\alpha | \dots | \beta_k\alpha$  falls

$A_j \rightarrow \beta_1 | \dots | \beta_k$  bis jede rechte Seite mit einem Terminalsymbol beginnt.

# Beispiel

$$S \rightarrow AB \mid a$$

$$A \rightarrow AA \mid b$$

$$B \rightarrow AB \mid a$$

$$S \rightarrow AB \mid a$$

$$A \rightarrow b \mid bZ_1$$

$$B \rightarrow AB \mid a$$

$$Z_1 \rightarrow A \mid AZ_1$$

$$S \rightarrow AB \mid a$$

$$A \rightarrow b \mid bZ_1$$

$$B \rightarrow bB \mid bZ_1B \mid a$$

$$Z_1 \rightarrow A \mid AZ_1$$

$$S \rightarrow AB \mid a$$

$$A \rightarrow b \mid bZ_1$$

$$B \rightarrow bB \mid bZ_1B \mid a$$

$$Z_1 \rightarrow b \mid bZ_1 \mid bZ_1 \mid bZ_1Z_1$$

$$S \rightarrow bB \mid bZ_1B \mid a$$

$$A \rightarrow b \mid bZ_1$$

$$B \rightarrow bB \mid bZ_1B \mid a$$

$$Z_1 \rightarrow b \mid bZ_1 \mid bZ_1Z_1$$

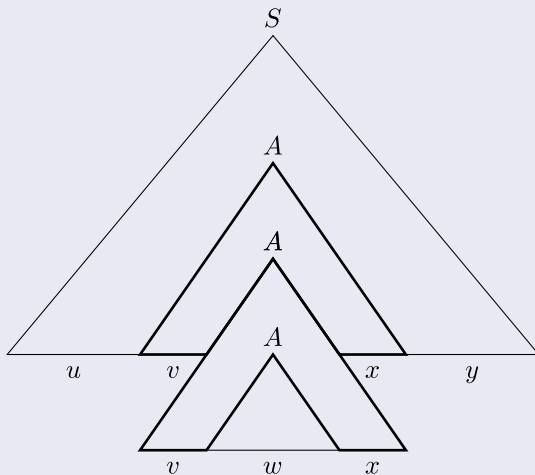
# Das Pumping-Lemma für CFLs

## Theorem

*Für jede CFL  $L$  gibt es eine Zahl  $n$  für die gilt: Jedes Wort  $z \in L$  mit  $|z| > n$  hat eine Zerlegung  $z = uvwxy$  mit*

- 1  $|vwx| \leq n$
- 2  $|vx| > 0$
- 3  $uv^iwx^iy \in L$  für jedes  $i \in \mathbf{N}_0$

## Beweis.



# Beispiel

$$L = \{ a^n b^n c^n \mid n > 0 \}.$$

Sei  $n$  die Konstante des Pumping-Lemmas.

$$a^n b^n c^n = uvwxy \text{ mit } |vwx| \leq n \text{ und } |vx| > 0.$$

Dann enthält  $vx$  kein  $a$  oder kein  $c$ .

Dann gilt leider  $uv^2wx^2y \notin L$ . Widerspruch!

Also ist  $L$  keine CFL.