

5b

Die Chomsky-Normalform

PD Dr. Jan Johannsen

Institut für Informatik

Stand: 3. Juni 2026

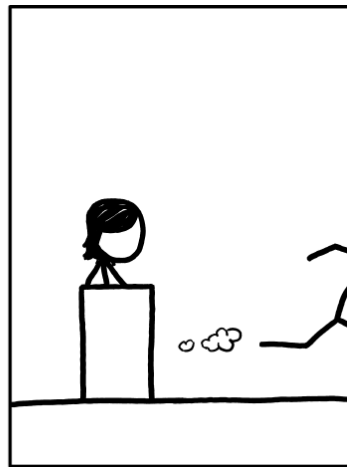
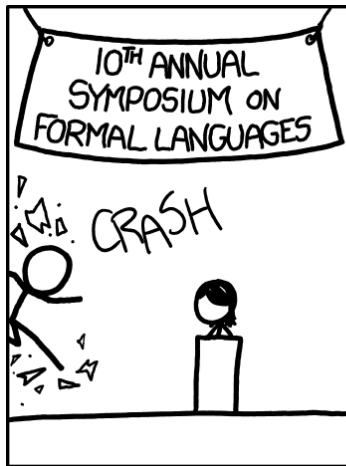
Basierend auf Folien von PD Dr. David Sabel und Prof. Dr. Jasmin Blanchette



Zusammenfassung:

- ▶ **Kontextfreie Sprachen** (*context-free languages*, **CFLs**) werden von **kontextfreien Grammatiken** (*context-free grammars*, **CFGs**) erzeugt.
- ▶ Das sind die **Typ 2-Sprachen** bzw. die **Typ 2-Grammatiken**.
- ▶ Bedingung: Alle linken Seiten der Produktionen bestehen aus genau einer Variablen, d.h. die Produktionen sind von der Form $A \rightarrow r$.

Kontextfreie Sprachen sind insbesondere nützlich um Sprachen mit Klammerungen zu beschreiben, z.B. Programmiersprachen.



xkcd.com/1090/

Beispiele für kontextfreie Sprachen

Die CFG $G_1 = (\{S, T\}, \{a, b\}, P_1, S)$ mit

$$P_1 := \{S \rightarrow \varepsilon \mid T, \\ T \rightarrow aTb \mid ab\}$$

erzeugt die Sprache $\{a^j b^j \mid j \in \mathbb{N}\}$, die daher kontextfrei ist.

Die CFG $G_2 = (\{E, M, A, Z, R\}, \{+, *, (,)\} \cup \{0, \dots, 9\}, P_2, E)$ mit

$$P_2 := \{E \rightarrow M \mid E + M, \\ M \rightarrow A \mid M * A, \\ A \rightarrow Z \mid (E), \\ Z \rightarrow 1R \mid \dots \mid 9R, \\ R \rightarrow 0R \mid \dots \mid 9R \mid \varepsilon\}$$

erzeugt einfache **arithmetische Ausdrücke** nach der „Punkt vor Strich“-Regel.

Normalformen von Grammatiken

- ▶ **Normalformen** fordern eine spezielle Form der Produktionen.
- ▶ Sie sind nützlich, wenn man Grammatiken analysiert oder Algorithmen auf Grammatiken formuliert: Man muss dann nur diese Form (statt aller erlaubten) von Produktionen betrachten.
- ▶ Wir betrachten zwei Normalformen:
die **Chomsky-Normalform** und die **Greibach-Normalform**.

Die Chomsky-Normalform

Definition

Eine CFG $G = (V, \Sigma, P, S)$ ist in **Chomsky-Normalform**, wenn für jede Produktion $A \rightarrow w \in P$ gilt: $w = a \in \Sigma$ oder $w = BC$ mit $B, C \in V$.

Beispiele:

- ▶ Die CFG $G = (\{A\}, \{ (,), [,] \}, \{ A \rightarrow (A) \mid () \mid [A] \mid [] \mid AA \}, A)$ ist **nicht** in Chomsky-Normalform. Nur die Produktion $A \rightarrow AA$ passt zum vorgeschriebenen Format.
- ▶ Die CFG $G' = (\{A, B, C, D, E, F, G\}, \{ (,), [,] \}, P, A)$ mit

$$P := \{ A \rightarrow BF \mid BC \mid DG \mid DE \mid AA, \\ B \rightarrow (, C \rightarrow), D \rightarrow [, E \rightarrow], F \rightarrow AC, G \rightarrow AE \}$$

ist in Chomsky-Normalform, und erzeugt dieselbe Sprache wie G .

Eigenschaften der Chomsky-Normalform

Für CFG G in Chomsky-Normalform gilt, dass Ableitungen eines Worts $w \in L(G)$ immer genau aus $2 \cdot |w| - 1$ **Ableitungsschritten** bestehen.

Zudem sind die dazugehörigen Syntaxbäume immer **Binärbäume** (Bäume, wobei jeder Knoten 0 oder 2 Kinder hat), bis auf die letzte Schicht, die Produktionen $A \rightarrow a$ anwendet.

Sei die CFG $G' = (\{A, B, C, D, E, F, G\}, \{ (,), [,] \}, P, A)$ mit

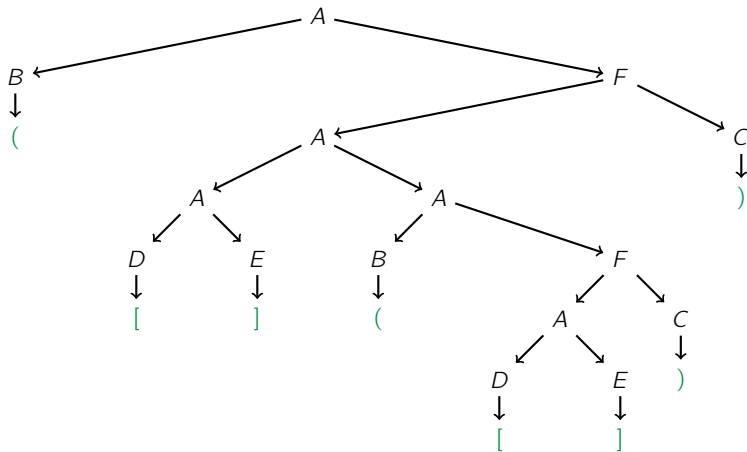
$$P := \{ A \rightarrow BF \mid BC \mid DG \mid DE \mid AA, \\ B \rightarrow (, C \rightarrow), D \rightarrow [, E \rightarrow], F \rightarrow AC, G \rightarrow AE \}$$

Ableitung von $([]([]))$:

$$\begin{aligned} A &\Rightarrow BF \Rightarrow (F \Rightarrow (AC \Rightarrow (AAC \Rightarrow (DEAC \Rightarrow ([EAC \\ &\Rightarrow ([]AC \Rightarrow ([]BFC \Rightarrow ([](FC \Rightarrow ([](ACC \Rightarrow ([](DECC \\ &\Rightarrow ([]([ECC \Rightarrow ([]([]CC \Rightarrow ([]([])C \Rightarrow ([]([])) \end{aligned}$$

Eigenschaften der Chomsky-Normalform

Der Syntaxbaum zur vorherigen Ableitung:



Herstellen der Chomsky-Normalform

Theorem

Für jede CFG G mit $\varepsilon \notin L(G)$ kann eine CFG G' in Chomsky-Normalform berechnet werden, sodass $L(G') = L(G)$ gilt.

Grundgedanke der Prozedur:

1. Entferne ε -Produktionen (wie in früherer Vorlesung).
2. Entferne Produktionen von der Form $A \rightarrow B$ („Entfernen von Einheitsproduktionen“).
3. Ersetze alle Terminale a in rechten Seiten, die nicht nur aus a bestehen, durch neue Variablen A und füge Produktionen $A \rightarrow a$ hinzu.
4. Zerlege alle Produktionen von der Form $A \rightarrow B_1 \cdots B_m$ mit $m > 2$ in mehrere:
 $A \rightarrow B_1 C_1, C_1 \rightarrow B_2 C_2, \dots, C_{m-2} \rightarrow B_{m-1} B_m$.

Beispiel für die Herstellung der Chomsky-Normalform

0. Eingabe: CFG $G = (\{A\}, \{ (,), [,] \}, \{ A \rightarrow (A) \mid () \mid [A] \mid [] \mid AA \}, A)$.
1. Es gibt keine ε -Produktionen zu entfernen.
2. Es gibt keine Einheitsproduktionen zu entfernen.
3. Das Ersetzen von Terminalen führt zu
 $G_3 = (\{A, B, C, D, E\}, \{ (,), [,] \}, P_3, A)$ mit

$$P_3 := \{ A \rightarrow BAC \mid BC \mid DAE \mid DE \mid AA \} \\ B \rightarrow (, C \rightarrow), D \rightarrow [, E \rightarrow] \}$$

4. Die Zerlegung von langen rechten Seiten führt zu
 $G' = (\{A, B, C, D, E, F, G\}, \{ (,), [,] \}, P', A)$ mit

$$P' := \{ A \rightarrow BF \mid BC \mid DG \mid DE \mid AA, \\ B \rightarrow (, C \rightarrow), D \rightarrow [, E \rightarrow], F \rightarrow AC, G \rightarrow AE \}$$

Entfernen von Einheitsproduktionen

Intuitiv ist klar, dass $A \rightarrow B$ entfernt werden kann:
Wenn erst $A \rightarrow B$, dann $B \rightarrow w$ angewendet wird,
kann man auch gleich $A \rightarrow w$ anwenden.

Algorithmisch zu beachten:

- ▶ Eliminiere in der richtigen Reihenfolge: Wenn $A \rightarrow B$ und $B \rightarrow C$, dann ist Ersetzen von $A \rightarrow B$ durch $A \rightarrow C$ nicht zielführend.
- ▶ Zyklen $A_1 \rightarrow A_2, A_2 \rightarrow A_3, \dots, A_n \rightarrow A_1$ müssen vorher entfernt werden.

Algorithmus 5: Entfernen von Einheitsproduktionen

Eingabe: Eine CFG $G = (V, \Sigma, P, S)$

Beginn

Erzeuge gerichteten Graph $D = (V, E)$, mit $(A, B) \in E$ für jede Einheitsproduktion $A \rightarrow B \in P$;

solange es einen Zyklus $(A_1, A_2), \dots, (A_{n-1}, A_n), (A_n, A_1) \in E$ gibt **tue**

```
   $P := P \setminus \{A_1 \rightarrow A_2, \dots, A_{n-1} \rightarrow A_n, A_n \rightarrow A_1\};$            /* entferne zyklische Regeln */
   $P := P[A_1/A_2, \dots, A_1/A_n];$            /* ersetze alle Vorkommen von  $A_i$  durch  $A_1$  für  $i = 2, \dots, n$  */
   $V := V \setminus \{A_2, A_3, \dots, A_n\};$            /* lösche  $A_2, \dots, A_n$  */
   $S := S[A_1/A_2, \dots, A_1/A_n];$            /* ersetze Startsymbol durch  $A_1$ , falls es  $A_i$ ,  $2 \leq i \leq n$  war */
   $E := E \setminus \{(A_1, A_2), \dots, (A_{n-1}, A_n), (A_n, A_1)\};$            /* entferne Zyklus aus Graph */
   $E := E[A_1/A_2, \dots, A_1/A_n];$            /* ersetze  $A_i$  durch  $A_1$  für  $i = 2, \dots, n$  in anderen Kanten */
```

Sortiere D topologisch und nummeriere die Variablen in V durch (und benenne entsprechend in E, P, S um),
sodass gilt: $A_i \rightarrow A_j$ impliziert $i < j$;

Sei $V = \{A_1, \dots, A_k\}$;

für $i = k$ **bis** 1 **tue**

wenn $A_i \rightarrow A_j \in P$ **dann**

Seien $A_j \rightarrow w_1, \dots, A_j \rightarrow w_m$ alle Produktionen mit A_j als linker Seite;

```
   $P := P \cup \{A_i \rightarrow w_1, \dots, A_i \rightarrow w_m\};$            /* füge  $A_i \rightarrow w_k$  hinzu für  $A_i \rightarrow A_j$  und  $A_j \rightarrow w_k$  */
```

```
   $P := P \setminus \{A_i \rightarrow A_j\};$            /* entferne  $A_i \rightarrow A_j$  */
```

Gib die so entstandene Grammatik als G' aus;

Satz

Algorithmus 5 berechnet bei Eingabe einer CFG G mit $\varepsilon \notin L(G)$ eine CFG G' , die keine Einheitsproduktionen hat, sodass gilt $L(G') = L(G)$.

Wenn G keine ε -Produktionen hat, dann hat auch G' keine ε -Produktionen.

Beweis Das Entfernen eines Zyklus ändert die erzeugte Sprache nicht.

Das Entfernen einer Einheitsproduktion $A_i \rightarrow A_j$ in der rückwärts-laufenden für-Schleife ändert die erzeugte Sprache nicht. Dies ist eine Instanz der Operation „Inlining von Produktionen“, die wir gleich korrekt beweisen werden.

Korrektheit von Algorithmus 5

Beweis (Fortsetzung) Es werden keine Einheitsproduktionen eingeführt, da die Produktionen topologisch sortiert behandelt werden. Wenn $A_i \rightarrow A_j$ entfernt wird, wurden **vorher** alle Einheitsproduktionen $A_j \rightarrow A_k$ entfernt. D.h. zu diesem Zeitpunkt gilt: für alle Produktionen $A_j \rightarrow w$ besteht w nicht nur aus einer Variablen.

Es werden offensichtlich nie ε -Produktionen eingeführt.

Der Algorithmus terminiert:

- ▶ Die solange-Schleife terminiert, da jede Iteration die Anzahl der Zyklen strikt verkleinert.
- ▶ Die für-Schleife terminiert offensichtlich. □

Lemma (Inlining von Produktionen)

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

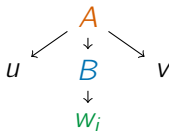
- ▶ $A \rightarrow uBv \in P$
- ▶ $B \rightarrow w_1 \mid \dots \mid w_n$ alle Regeln mit B als linker Seite

und sei $G' = (V, \Sigma, (P \setminus \{A \rightarrow uBv\}) \cup \{A \rightarrow uw_1v \mid \dots \mid uw_nv\}, S)$.

Dann gilt $L(G') = L(G)$.

Inlining von Produktionen

Beweis Das folgt durch Modifizieren der Syntaxbäume zur Ableitung mit G bzw. G' :
Tausche alle Baumabschnitte



im Syntaxbaum mit G durch



im Syntaxbaum mit G' .

□

Algorithmus 6: Herstellen der Chomsky-Normalform

Eingabe: CFG G mit $\varepsilon \notin L(G)$

Ausgabe: CFG G' in Chomsky-Normalform mit $L(G') = L(G)$

Beginn

Entferne die ε -Produktionen in G mit Algorithmus 1 und
entferne anschließend die Einheitsproduktionen mit Algorithmus 5;

Sei $G' = (V', \Sigma, P', S')$ die entstandene Grammatik;

für alle $a \in \Sigma$ **tue**

Sei B eine neue Variable;

/ Führe neue Variable B für a ein, und ersetze Vorkommen von a durch B */*

$G' := (V' \cup \{B\}, \Sigma, \{A \rightarrow w[B/a] \mid A \rightarrow w \in P' \text{ und } |w| > 1\}$
 $\quad \cup \{A \rightarrow w \mid A \rightarrow w \in P' \text{ und } |w| = 1\} \cup \{B \rightarrow a\}, S)$

/ Nun sind alle Regeln von der Form $A \rightarrow a$ oder $A \rightarrow B_1 \cdots B_m$ mit $m \geq 2$ */*

für alle $A \rightarrow B_1 \cdots B_m \in P'$ mit $m > 2$ **tue**

Seien $C_1, \dots, C_{m-2}$ neue Variablen;

$V' := V' \cup \{C_1, \dots, C_{m-2}\};$

/ Ersetze in P' die Produktion $A \rightarrow B_1 \cdots B_m$ durch neue Regeln */*

$P' := (P' \setminus \{A \rightarrow B_1 \cdots B_m\})$
 $\quad \cup \{A \rightarrow B_1 C_1\} \cup \{C_i \rightarrow B_{i+1} C_{i+1} \mid \text{für } i = 1, \dots, m-3\} \cup \{C_{m-2} \rightarrow B_{m-1} B_m\};$

Korrektheit von Algorithmus 6

Theorem

Für jede CFG G mit $\varepsilon \notin L(G)$ kann eine CFG G' in Chomsky-Normalform berechnet werden, sodass $L(G') = L(G)$ gilt.

Beweis Algorithmus 6 berechnet eine solche äquivalente CFG.

Die Schritte „Entfernen von ε -Produktionen“ und „Entfernen von Einheitsproduktionen“ haben wir als korrekt gezeigt.

Die Schritte „Einführen von Produktionen $B \rightarrow a$ “ und „Behandlung von $A \rightarrow B_1 \cdots B_m$ “ sind Varianten der korrekten Operation „Boxing von Satzformen“, die wir gleich korrekt beweisen werden.

Am Ende der Berechnung haben alle Produktionen die gewünschte Form. □

Boxing von Satzformen

Lemma (Boxing von Satzformen)

Sei G eine CFG mit $G = (V, \Sigma, P \cup \{A \rightarrow w_1 \cdots w_n\}, S)$.

Seien $B_1, \dots, B_n$ neue Variablen (d.h. $V \cap \{B_1, \dots, B_n\} = \emptyset$) und

$G' = (V \cup \{B_1, \dots, B_n\}, \Sigma, P \cup \{A \rightarrow B_1 \cdots B_n, B_1 \rightarrow w_1, \dots, B_n \rightarrow w_n\}, S)$.

Dann gilt $L(G') = L(G)$.

Beweis

⊇ Wir zeigen: $w \in L(G)$ impliziert $w \in L(G')$.

Konstruiere aus $S \Rightarrow_G^* w$ die Ableitung $S \Rightarrow_{G'}^* w$: Übersetze jeden Schritt

$uAv \Rightarrow_G uw_1 \cdots w_nv$ in $uAv \Rightarrow_{G'} uB_1 \cdots B_nv \Rightarrow_{G'}^n uw_1 \cdots w_nv$.

Beweis (Fortsetzung)

⊆ Wir zeigen: $w \in L(G')$ impliziert $w \in L(G)$.

Betrachte den Syntaxbaum für $S \Rightarrow_{G'}^* w$.

Identifiziere die Anwendungen der Regeln $A \rightarrow B_1 \cdots B_n$ und $B_i \rightarrow w_i$, und modifiziere den Syntaxbaum durch Anwendung der Regel $A \rightarrow w_1 \cdots w_n$.

Lies Ableitung $S \Rightarrow_G^* w$ ab.



Weiteres Beispiel für die Herstellung der Chomsky-Normalform

Eingabe: $G_0 = (\{A, B, C, D, S\}, \{0, 1\}, P_0, S)$ mit
 $P_0 = \{S \rightarrow 1A, A \rightarrow AB, A \rightarrow DA, A \rightarrow \varepsilon,$
 $B \rightarrow 0, B \rightarrow 1, C \rightarrow AAA, D \rightarrow 1AC\}$

Schritt 1: Entfernen von ε -Produktionen

$G_0 = (\{A, B, C, D, S\}, \{0, 1\}, P_0, S)$ mit
 $P_0 = \{S \rightarrow 1A, A \rightarrow AB, A \rightarrow DA, A \rightarrow \varepsilon, B \rightarrow 0, \\ B \rightarrow 1, C \rightarrow AAA, D \rightarrow 1AC\}$

- Finde Menge W der Variablen, die ε herleiten:
 $W = \{A, C\}$ da $A \rightarrow \varepsilon$ und $C \rightarrow AAA$

- Starte mit

$G_1 = (\{A, B, C, D, S\}, \{0, 1\}, P_1, S)$ mit
 $P_1 = \{S \rightarrow 1A, A \rightarrow AB, A \rightarrow DA, B \rightarrow 0, \\ B \rightarrow 1, C \rightarrow AAA, D \rightarrow 1AC\}$

- Füge Produktionen für Vorkommen von A und C hinzu:

$P_1 = \{S \rightarrow 1A, S \rightarrow 1, A \rightarrow AB, A \rightarrow B, A \rightarrow DA, A \rightarrow D, \\ B \rightarrow 0, B \rightarrow 1, C \rightarrow AAA, C \rightarrow AA, C \rightarrow A, \\ D \rightarrow 1AC, D \rightarrow 1A, D \rightarrow 1C, D \rightarrow 1\}$

Schritt 2: Entfernen von Einheitsproduktionen

$$P_1 = \{S \rightarrow 1A, S \rightarrow 1, A \rightarrow AB, A \rightarrow B, A \rightarrow DA, A \rightarrow D, \\ B \rightarrow 0, B \rightarrow 1, C \rightarrow AAA, C \rightarrow AA, C \rightarrow A, \\ D \rightarrow 1AC, D \rightarrow 1A, D \rightarrow 1C, D \rightarrow 1\}$$

- Der gerichtete Graph ist $D' = (\{S, A, B, C, D\}, \{(A, B), (A, D), (C, A)\})$.
- Es gibt keine Zyklen, wir erhalten $G_2 = G_1$.
- Sortieren und Umbenennen der Variablen, sodass „ $A_i \rightarrow A_j$ impliziert $i < j$ “ gilt, erfordert eine Umbenennung ρ , welche die Beziehungen $A < B$, $A < D$, $C < A$ erzeugt.
- Wir wählen ρ mit $\rho(C) = A_1$, $\rho(A) = A_2$, $\rho(B) = A_3$, $\rho(D) = A_4$, $\rho(S) = A_5$.
- Das liefert uns $G_3 = (\{A_1, A_2, A_3, A_4, A_5\}, \Sigma, P_3, A_5)$ mit
$$P_3 = \{A_5 \rightarrow 1A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2A_3, A_2 \rightarrow A_3, A_2 \rightarrow A_4A_2, \\ A_2 \rightarrow A_4, A_3 \rightarrow 0, A_3 \rightarrow 1, A_1 \rightarrow A_2A_2A_2, A_1 \rightarrow A_2A_2, \\ A_1 \rightarrow A_2, A_4 \rightarrow 1A_2A_1, A_4 \rightarrow 1A_2, A_4 \rightarrow 1A_1, A_4 \rightarrow 1\}$$

Schritt 2: Entfernen von Einheitsproduktionen

$$P_3 = \{A_5 \rightarrow 1A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2A_3, A_2 \rightarrow A_3, A_2 \rightarrow A_4A_2, \\ A_2 \rightarrow A_4, A_3 \rightarrow 0, A_3 \rightarrow 1, A_1 \rightarrow A_2A_2A_2, A_1 \rightarrow A_2A_2, \\ A_1 \rightarrow A_2, A_4 \rightarrow 1A_2A_1, A_4 \rightarrow 1A_2, A_4 \rightarrow 1A_1, A_4 \rightarrow 1\}$$

► Nun läuft die für-Schleife für i von 5 bis 1:

► Für $i = 5, i = 4, i = 3$ gibt es jeweils keine Produktion der Form $A_i \rightarrow A_j$.

► Für $i = 2$ wird $A_2 \rightarrow A_3$ ersetzt durch $A_2 \rightarrow 0, A_2 \rightarrow 1$, und es wird $A_2 \rightarrow A_4$ ersetzt durch $A_2 \rightarrow 1A_2A_1, A_2 \rightarrow 1A_2, A_2 \rightarrow 1A_1$ und $A_2 \rightarrow 1$.
Danach ist

$$P_4 = \{A_5 \rightarrow 1A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2A_3, A_2 \rightarrow 0, A_2 \rightarrow 1, A_2 \rightarrow A_4A_2, \\ A_2 \rightarrow 1A_2A_1, A_2 \rightarrow 1A_2, A_2 \rightarrow 1A_1, A_3 \rightarrow 0, A_3 \rightarrow 1, \\ A_1 \rightarrow A_2A_2A_2, A_1 \rightarrow A_2A_2, A_1 \rightarrow A_2, A_4 \rightarrow 1A_2A_1, \\ A_4 \rightarrow 1A_2, A_4 \rightarrow 1A_1, A_4 \rightarrow 1\}$$

► Für $i = 1$ wird $A_1 \rightarrow A_2$ ersetzt durch $A_1 \rightarrow A_2A_3, A_1 \rightarrow 0, A_1 \rightarrow 1, A_1 \rightarrow A_4A_2, \\ A_1 \rightarrow 1A_2A_1, A_1 \rightarrow 1A_2$ und $A_1 \rightarrow 1A_1$.

Schritt 2: Entfernen von Einheitsproduktionen

- Daher ist die Grammatik nach Entfernen von Einheitsproduktionen:

$G_5 = (V_5, \Sigma, P_5, A_5)$ mit $V_5 = \{A_1, A_2, A_3, A_4, A_5\}$ und

$P_5 = \{A_5 \rightarrow 1A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2A_3, A_2 \rightarrow 0, A_2 \rightarrow 1, A_2 \rightarrow A_4A_2, A_2 \rightarrow 1A_2A_1, A_2 \rightarrow 1A_2, A_2 \rightarrow 1A_1, A_3 \rightarrow 0, A_3 \rightarrow 1, A_1 \rightarrow A_2A_2A_2, A_1 \rightarrow A_2A_2, A_1 \rightarrow A_2A_3, A_1 \rightarrow 0, A_1 \rightarrow 1, A_1 \rightarrow A_4A_2, A_1 \rightarrow 1A_2A_1, A_1 \rightarrow 1A_2, A_1 \rightarrow 1A_1, A_4 \rightarrow 1A_2A_1, A_4 \rightarrow 1A_2, A_4 \rightarrow 1A_1, A_4 \rightarrow 1\}$

Schritt 3: Terminale durch Variablen darstellen

- Füge $B_0 \rightarrow 0$ und $B_1 \rightarrow 1$ hinzu und ersetze in rechten Seiten mit Wortlänge > 1 :

$$P_6 = \{B_0 \rightarrow 0, B_1 \rightarrow 1, A_5 \rightarrow B_1 A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2 A_3, A_2 \rightarrow 0, \\ A_2 \rightarrow 1, A_2 \rightarrow A_4 A_2, A_2 \rightarrow B_1 A_2 A_1, A_2 \rightarrow B_1 A_2, \\ A_2 \rightarrow B_1 A_1, A_3 \rightarrow 0, A_3 \rightarrow 1, A_1 \rightarrow A_2 A_2 A_2, A_1 \rightarrow A_2 A_2, \\ A_1 \rightarrow A_2 A_3, A_1 \rightarrow 0, A_1 \rightarrow 1, A_1 \rightarrow A_4 A_2, A_1 \rightarrow B_1 A_2 A_1, \\ A_1 \rightarrow B_1 A_2, A_1 \rightarrow B_1 A_1, A_4 \rightarrow B_1 A_2 A_1, \\ A_4 \rightarrow B_1 A_2, A_4 \rightarrow B_1 A_1, A_4 \rightarrow 1\}$$

Schritt 4: Rechte Seiten zerlegen

- Zerlege rechte Seiten mit Wortlänge > 2 :

Ergibt $G_7 = (V_7, \Sigma, P_7, A_5)$, wobei

$$V_7 = \{A_1, A_2, A_3, A_4, A_5, B_0, B_1, C_1, C_2, C_3, C_4\}$$

$$P_7 = \{B_0 \rightarrow 0, B_1 \rightarrow 1, A_5 \rightarrow B_1 A_2, A_5 \rightarrow 1, A_2 \rightarrow A_2 A_3, A_2 \rightarrow 0, \\ A_2 \rightarrow 1, A_2 \rightarrow A_4 A_2, A_2 \rightarrow B_1 C_1, C_1 \rightarrow A_2 A_1, \\ A_2 \rightarrow B_1 A_2, A_2 \rightarrow B_1 A_1, A_3 \rightarrow 0, A_3 \rightarrow 1, A_1 \rightarrow A_2 C_2, \\ C_2 \rightarrow A_2 A_2, A_1 \rightarrow A_2 A_2, A_1 \rightarrow A_2 A_3, A_1 \rightarrow 0, A_1 \rightarrow 1, \\ A_1 \rightarrow A_4 A_2, A_1 \rightarrow B_1 C_3, C_3 \rightarrow A_2 A_1, A_1 \rightarrow B_1 A_2, \\ A_1 \rightarrow B_1 A_1, A_4 \rightarrow B_1 C_4, C_4 \rightarrow A_2 A_1, A_4 \rightarrow B_1 A_2, \\ A_4 \rightarrow B_1 A_1, A_4 \rightarrow 1\}$$

G_7 ist in Chomsky-Normalform.