Theoretische Informatik
WS 2024/2025

Kapitel 3

Kontextfreie Sprachen

3.2

Chomsky Normalform (CNF)

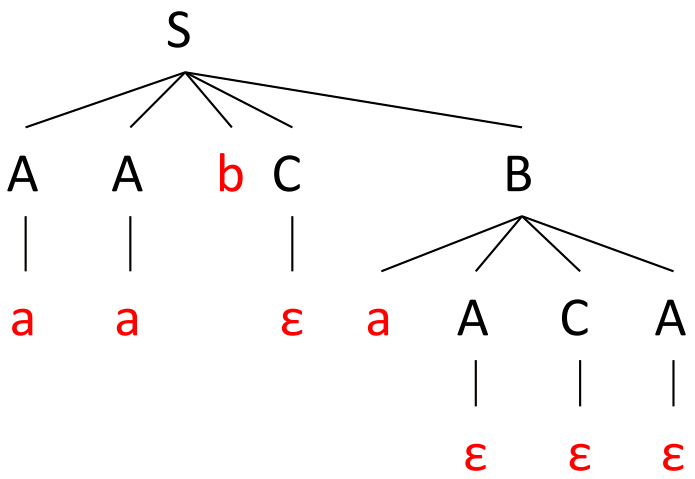
Prof. Dr. Robert Preis
Fachbereich Informatik
Fachhochschule Dortmund
Robert.Preis@fh-dortmund.de

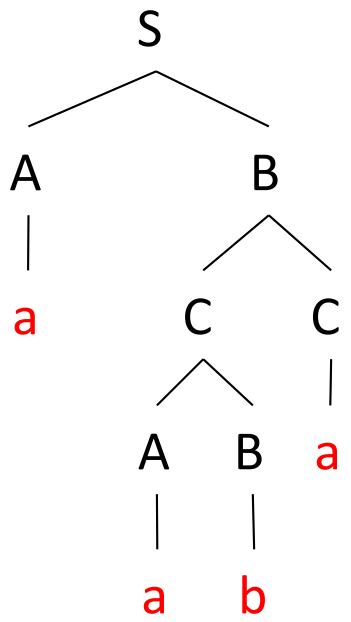
Ableitungsbäume von kontextfreien Grammatiken

G=(V,T,P,S) mit P = {
S → AAbCB | ABBc,
A → BACS | a | ε,
B → CC | aACA | ab,
C → AB | a | ε,
D → BcCa | BAaC}

H=(V,T,P,S) mit P = {
S → AB | BC,
A → BA | a,
B → CC | b,
C → AB | a }

Wie sehen die Ableitungsbäume für das Wort w = aaba aus?





Bei G ist der Ableitungsbaum recht chaotisch, bei H ist es ein Binärbaum.

Grammatik in Chomsky-Normalform

Ein Grammatik $G_{CNF} = (V,T,P,S)$ ist in

Chomsky-Normalform (CNF)

wenn G keine unnützen Variablen enthält und jede Produktion eine der drei Formen hat:

1. $A \rightarrow BC$ mit $A,B,C \in V$
2. $A \rightarrow a$ mit $A \in V$ und $a \in T$
3. $S \rightarrow \varepsilon$ nur falls S Startvariable und S nirgends auf der rechten Seite enthalten ist. Dies ist wichtig für $\varepsilon \in L(G_{CNF})$.

Jede kontextfreie Grammatik G kann man in eine Grammatik G_{CNF} in CNF umwandeln mit $L(G_{CNF}) = L(G)$.

Umformung in CNF

Die Umformung einer allgemeinen kontextfreien Grammatik in eine kontextfreie Grammatik in Chomsky Normalform erfolgt in mehreren Schritten:

1. Eliminierung von ε-Produktionen $A \rightarrow \varepsilon$ falls A kein Startzustand
   (die Produktionen müssen rechts mindestens die Länge 1 haben)
2. Eliminierung von Einheitsproduktionen $A \rightarrow B$
   (Produktionen der Länge 1 dürfen nicht aus einer Variablen bestehen)
3. Eliminierung unnützer Symbole
   (keine unnütze Variablen erlaubt)
4. Separieren von Terminalen und Variablen in Produktionen (X-Regel)
   (Produktionen haben rechts entweder Variablen oder Terminale)
5. Aufspalten von Produktionen $A \rightarrow \alpha$ mit $|\alpha|>2$ (Y-Regel)
   (Produktionen haben rechts höchstens die Länge 2)

Schritt 1: Elimination von ε-Übergängen

ε-Produktionen sind überflüssig (Ausnahme: S→ε).

Eine Variable A∈V ist eliminierbar, falls A →* ε.

(Sonderfall: falls S eliminierbar UND S ist irgendwo auf der rechten Seite:
erzeuge neue Startvariable S‘ und S‘ → ε | S.)

P = {

S → **AB** | CD | abc, S ist eliminierbar, weil A und B eliminierbar sind

A → D | a**AA**b | **ε**, A ist eliminierbar

B → b**BB** | **ε** , B ist eliminierbar

C → c,

D → cD}

Neue Grammatik:

P_1 = {

S → AB | CD | abc | ε | A | B, *Die rechten Seiten*

A → D | aAAb | aAb | ab, *haben alle die*

B → bBB | bB | b, *Länge >= 1*

C → c, *(außer S → ε) !*

D → cD}

Schritt 1: Elimination von ε-Übergängen: Verfahren

Bestimmung aller eliminierbaren Variablen:

1. Ist $A \rightarrow \varepsilon \in P$ dann ist A eliminierbar.
2. Ist $A \rightarrow X_1 \ldots X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar.

Verfahren zur Erzeugung einer Grammatik ohne ε-Regeln (ohne eliminierbaren Variablen):

1. Bestimme alle eliminierbaren Variablen.
2. Falls S eliminierbar ist:
   - falls S nirgendwo auf der rechten Seite vorkommt: erzeuge $S \rightarrow \varepsilon$
   - sonst erzeuge neue Startvariable S‘ und $S' \rightarrow \varepsilon \mid S$.
3. Für $A \rightarrow \alpha \in P$ mit eliminierbaren Symbolen $X_1, \ldots, X_m$ in α erzeuge neue Regeln (maximal 2^m-1): Streiche dabei jeweils alle möglichen Teilmengen von $\{X_1, \ldots, X_m\}$ aus α.
4. Entferne alle Regeln der Form $A \rightarrow \varepsilon$ (nur falls A keine Startvariable).

Die neue Grammatik erzeugt die selben Wörter!
Die neue Grammatik enthält keine ε-Regeln mehr außer $S \rightarrow \varepsilon$!

Schritt 2: Elimination von Einheitsproduktionen

Einheitsproduktionen A→B dürfen nicht vorkommen und sind auch unnötig.

(A, B) ist ein Einheitspaar, falls A →* B.

P_1 = {

S → AB | CD | abc | ε | A | B,	Einheitspaare (S,S),(S,A),(S,B),(S,D)
A → D | aAAb | aAb | ab,	Einheitspaare (A,A),(A,D)
B → bBB | bB | b,	Einheitspaar (B,B)
C → c,	Einheitspaar (C,C)
D → cD}	Einheitspaar (D,D)

Neue Grammatik:

P_2 = {

S → AB | CD | abc | ε | aAAb | aAb | ab | bBB | bB | b | cD,
A → aAAb | aAb | ab | cD,
B → bBB | bB | b,
C → c,
D → cD}

Auf den rechten Seiten steht keine Variable mehr alleine!

Schritt 2: Elimination von Einheitsproduktionen: Verfahren

Bestimmung aller Einheitspaare:

1. Alle Paare (A,A) für A∈V sind Einheitspaare.
2. Ist (A, B) ein Einheitspaar und B→C ∈ P dann ist (A,C) Einheitspaar.

Verfahren zur Erzeugung einer Grammatik ohne Einheitsproduktionen:

1. Bestimme alle Einheitspaare in G.
2. Für jedes Einheitspaar (A, B) erzeuge neue Produktionen
   { A→α | B→α ∈ P ist keine Einheitsproduktion}.
3. Lösche alle Einheitsproduktionen.

Die neue Grammatik erzeugt die selben Wörter!

Die neue Grammatik enthält keine Einheitsproduktionen mehr!

Schritt 3: Elimination von unnützen Variablen

Nur nützliche Variablen machen Sinn.

Eine Symbol X ist nützlich, falls es eine Ableitung $S \rightarrow^* \alpha X \beta \rightarrow^* w \in T^*$ gibt, d.h. es ist erreichbar ($S \rightarrow^* \alpha X \beta$) und erzeugend ($X \rightarrow^* v \in T^*$).

P_2 = {

S → AB|CD|abc|ε|aAAb|aAb|ab|bBB|bB|b|cD, erzeugend / erreichbar

A → aAAb | aAb | ab | cD, erzeugend / erreichbar

B → bBB | bB | b, erzeugend / erreichbar

C → c, erz./err., nach Löschung von S→CD nicht erreichbar

D → cD} nicht erzeugend / erreichbar

Neue Grammatik:

P_3 = {

S → AB | abc | ε | aAAb | aAb | ab | bBB | bB | b, *Alle Variablen*

A → aAAb | aAb | ab, *sind erreichbar*

B → bBB | bB | b} *und erzeugend!*

Schritt 3: Elimination von unnützen Variablen: Verfahren

Verfahren zur Bestimmung aller erzeugender Symbole:

1. Alle Terminalsymbole a ∈ T sind erzeugend.
2. Ist $A \rightarrow X_1 \ldots X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend.

Verfahren zur Bestimmung aller erreichbaren Symbole:

1. S ist erreichbar.
2. Ist $A \rightarrow X_1 \ldots X_n \in P$ und A erreichbar dann sind alle X_i erreichbar.

Verfahren zur Elimination von unnützen Variablen:

1. Lösche alle nicht erzeugenden Variablen (mit Produktionen).
2. Lösche alle nicht erreichbaren Variablen (mit Produktionen).

Die neue Grammatik erzeugt die selben Wörter!

Die neue Grammatik enthält keine unnützen Variablen mehr!

Schritt 4: Separieren von Terminalen und Variablen (X-Regel)

Auf der rechten Seite der Regeln müssen entweder
NUR mind. 2 Variablen oder NUR ein Terminal stehen.

P_3 = {
S → AB | abc | ε | aAAb | aAb | ab | bBB | bB | b,
A → aAAb | aAb | ab,
B → bBB | bB | b}

Neue Grammatik:

P_4 = {
S → AB | $X_aX_bX_c$ | ε | X_aAAX_b | X_aAX_b | X_aX_b | X_bBB | X_bB | b,
A → X_aAAX_b | X_aAX_b | X_aX_b,
B → X_bBB | X_bB | b,
X_a → a,
X_b → b,
X_c → c}

Auf den rechten Seiten ist entweder genau ein Terminal oder sind mindestens 2 Variablen!

Schritt 4: Separieren von Terminalen und Variablen: Verfahren

Verfahren zur Separation von Terminalsymbolen und Variablen:

1. Für jedes Terminalsymbol a ∈ T erzeuge neue Variable X_a.
2. In jeder Produktion $A \rightarrow \alpha$ mit $|\alpha| \geq 2$, in der $a \in T$ auftaucht, ersetze a durch X_a.
3. Ergänze Produktionen $X_a \rightarrow a$ für alle $a \in T$.
4. Lösche Variable X_a, falls nicht verwendet.

Die neue Grammatik erzeugt die selben Wörter!

Die neue Grammatik separiert die Terminale und Variablen!

Schritt 5: Aufspalten von A→α mit |α|>2 (Y-Regel)

Auf der rechten Seite müssen alle Regeln mit Variablen aus **genau zwei** Variablen bestehen.

P_4 = {

S → AB | $X_aX_bX_c$ | ε | X_aAAX_b | X_aAX_b | X_aX_b | X_bBB | X_bB | b,

A → X_aAAX_b | X_aAX_b | X_aX_b,

B → X_bBB | X_bB | b,

X_a → a, X_b → b, X_c → c}

Neue Grammatik:

P_5 = {

S → AB | X_aY_1 | ε | X_aY_2 | X_aY_4 | X_aX_b | X_bY_5 | X_bB | b,

Y_1→X_bX_c, Y_2→AY_3, Y_3→AX_b, Y_4→AX_b, Y_5→BB,

A → X_aY_6 | X_aY_8 | X_aX_b, Y_6→AY_7, Y_7→AX_b, Y_8→AX_b,

B → X_bY_9 | X_bB | b, Y_9→BB,

X_a → a, X_b → b, X_c → c}

Alle Regeln sind in CNF !

Schritt 5: Aufspalten von A→α mit |α|>2: Verfahren

Verfahren zur Aufspaltung von Produktionen A → α mit |α|>2:

Ersetze jede Produktion

$$A \rightarrow X_1 \ldots X_k$$

durch k−1 Produktionen mit k-2 neuen Variablen $Y_1, \ldots Y_{k-2}$

$$A \rightarrow X_1 Y_1, \quad Y_1 \rightarrow X_2 Y_2, \quad \ldots \quad Y_{k-2} \rightarrow X_{k-1} X_k$$

Die neue Grammatik erzeugt die selben Wörter!

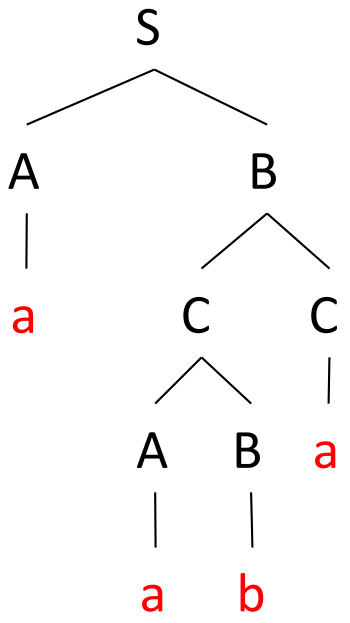
Die neue Grammatik ist in CNF!

Ableitung bei CNF-Grammatiken: Binärbäume

Wie sehen die Ableitungsbäume von CNF-Grammatiken aus?

Sei G=(V,T,P,S) mit

P = { S → AB | BC,
A → BA | a,
B → CC | b,
C → AB | a }



Wie sieht ein Ableitungsbaum für w = „aaba“ aus?

Beobachtungen für einen Ableitungsbaum für ein Wort w≠ε:

- Es gibt genau |w| Blätter.
- Die Variablenknoten bilden einen Binärbaum (d.h. immer entweder 0 oder 2 Kinder als Variablen).
- Es gibt genau 2|w|−1 Variablenknoten.

Jede Ableitung eines Wortes w hat die Länge 2|w|-1.

Zusammenfassung

- Man kann jede kontextfreie Grammatik in eine Grammatik in **Chomsky-Normalform** umwandeln.
- CNF-Grammatiken erzeugen **binäre Ableitungsbäume** (max. 2 Kinder pro Knoten). Analysen und Beweise von Eigenschaften werden einfacher.