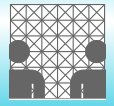


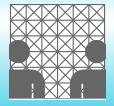
F2 — Automaten und formale Sprachen

Matthias Jantzen
(nach und mit Folienvorlagen von Berndt Farwer)

Fachbereich Informatik
AB „Theoretische Grundlagen der Informatik“ (TGI)
Universität Hamburg
jantzen@informatik.uni-hamburg.de

Reduzierte Grammatik

- **Definition:** Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt genau dann **reduziert**, wenn gilt:



Reduzierte Grammatik

- **Definition:** Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt genau dann **reduziert**, wenn gilt:

  1. Jedes Nonterminal ist **produktiv**, d.h.

$$\forall A \in V_N \text{ gilt } \exists w \in V_T^* : \; A \overset{*}{\underset{G}{\Longrightarrow}} w.$$

Reduzierte Grammatik

- **Definition:** Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt genau dann **reduziert**, wenn gilt:

  1. Jedes Nonterminal ist **produktiv**, d.h.

     $$\forall A \in V_N \text{ gilt } \exists w \in V_T^* : \; A \overset{*}{\underset{G}{\Longrightarrow}} w.$$

  2. Jedes Nonterminal ist **erreichbar**, d.h.

     $$\forall A \in V_N \text{ gilt } \exists u, v \in V^* : \; S \overset{*}{\underset{G}{\Longrightarrow}} uAv.$$

Reduzierte Grammatik

- **Definition:** Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt genau dann **reduziert**, wenn gilt:

  1. Jedes Nonterminal ist **produktiv**, d.h.

     $$\forall A \in V_N \text{ gilt } \exists w \in V_T^* : \; A \overset{*}{\underset{G}{\Longrightarrow}} w.$$

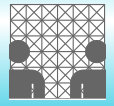
  2. Jedes Nonterminal ist **erreichbar**, d.h.

     $$\forall A \in V_N \text{ gilt } \exists u, v \in V^* : \; S \overset{*}{\underset{G}{\Longrightarrow}} uAv.$$

- Eine reduzierte Grammatik enthält möglicherweise trotzdem noch unnötig viele Symbole!

Konstruktion reduzierter CFG

- Die Konstruktion besteht aus zwei separaten Teilen:



Konstruktion reduzierter CFG

- Die Konstruktion besteht aus zwei separaten Teilen:
- **Teil 1** entfernt Symbole (und Produktionen), so dass danach alle verwendeten Nonterminale auf reine Terminalwörter abgeleitet werden können, d.h. *G enthält danach nur noch produktive Nonterminalsymbole*.

Konstruktion reduzierter CFG

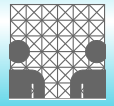
- Die Konstruktion besteht aus zwei separaten Teilen:
- **Teil 1** entfernt Symbole (und Produktionen), so dass danach alle verwendeten Nonterminale auf reine Terminalwörter abgeleitet werden können, d.h. G *enthält danach nur noch produktive Nonterminalsymbole*.
- **Teil 2** entfernt danach alle Symbole, die vom Startsymbol aus *nicht* erreichbar sind.

Konstruktion reduzierter CFG

- Die Konstruktion besteht aus zwei separaten Teilen:
- **Teil 1** entfernt Symbole (und Produktionen), so dass danach alle verwendeten Nonterminale auf reine Terminalwörter abgeleitet werden können, d.h. G *enthält danach nur noch produktive Nonterminalsymbole*.
- **Teil 2** entfernt danach alle Symbole, die vom Startsymbol aus *nicht* erreichbar sind.
- Die Reihenfolge der Anwendung dieser Schritte ist wichtig!

Konstruktion reduzierter CFG

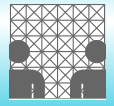
- Die Konstruktion besteht aus zwei separaten Teilen:
- **Teil 1** entfernt Symbole (und Produktionen), so dass danach alle verwendeten Nonterminale auf reine Terminalwörter abgeleitet werden können, d.h. G *enthält danach nur noch produktive Nonterminalsymbole*.
- **Teil 2** entfernt danach alle Symbole, die vom Startsymbol aus *nicht* erreichbar sind.
- Die Reihenfolge der Anwendung dieser Schritte ist wichtig!
- . . . im Wesentlichen dasselbe Verfahren, wie bei der initialen Zusammenhangskomponente eines DFA.



produktive Nonterminale

$$
\begin{aligned}
S &\longrightarrow ACS \mid ABBS \mid \lambda \\
A &\longrightarrow Ca \mid CDb \mid aB \\
B &\longrightarrow bb \mid bB \\
C &\longrightarrow bCE \mid aD \\
D &\longrightarrow aDE \mid bC \\
E &\longrightarrow aAb
\end{aligned}
$$

$$M_0 := V_T = \{a, b\}$$



produktive Nonterminale

$$
\begin{aligned}
S &\longrightarrow ACS \mid ABBS \mid \lambda \\
A &\longrightarrow Ca \mid CDb \mid aB \\
B &\longrightarrow bb \mid bB \\
C &\longrightarrow bCE \mid aD \\
D &\longrightarrow aDE \mid bC \\
E &\longrightarrow aAb
\end{aligned}
$$

$$M_0 := V_T = \{a, b\}$$

$$M_1 := M_0 \cup \{S, B\} = \{a, b, S, B\}$$

produktive Nonterminale

$$S \longrightarrow ACS \mid ABBS \mid \lambda$$
$$A \longrightarrow Ca \mid CDb \mid aB$$
$$B \longrightarrow bb \mid bB$$
$$C \longrightarrow bCE \mid aD$$
$$D \longrightarrow aDE \mid bC$$
$$E \longrightarrow aAb$$

$$M_0 := V_T = \{a, b\}$$

$$M_1 := M_0 \cup \{S, B\} = \{a, b, S, B\}$$

$$M_2 := M_1 \cup \{A\} = \{a, b, S, A, B\}$$

produktive Nonterminale

$$
\begin{aligned}
S &\longrightarrow ACS \mid ABBS \mid \lambda \\
A &\longrightarrow Ca \mid CDb \mid aB \\
B &\longrightarrow bb \mid bB \\
C &\longrightarrow bCE \mid aD \\
D &\longrightarrow aDE \mid bC \\
E &\longrightarrow aAb
\end{aligned}
$$

$$M_0 := V_T = \{a, b\}$$

$$M_1 := M_0 \cup \{S, B\} = \{a, b, S, B\}$$

$$M_2 := M_1 \cup \{A\} = \{a, b, S, A, B\}$$

$$M_3 := M_2 \cup \{E\} = \{a, b, S, A, B, E\}$$

produktive Nonterminale

$$S \longrightarrow ACS \mid ABBS \mid \lambda$$
$$A \longrightarrow Ca \mid CDb \mid aB$$
$$B \longrightarrow bb \mid bB$$
$$C \longrightarrow bCE \mid aD$$
$$D \longrightarrow aDE \mid bC$$
$$E \longrightarrow aAb$$

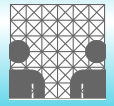
$$M_0 := V_T = \{a, b\}$$

$$M_1 := M_0 \cup \{S, B\} = \{a, b, S, B\}$$

$$M_2 := M_1 \cup \{A\} = \{a, b, S, A, B\}$$

$$M_3 := M_2 \cup \{E\} = \{a, b, S, A, B, E\}$$

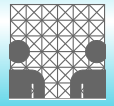
$$M_4 := M_3 \cup \emptyset$$



erreichbare Nonterminale

$$S \longrightarrow ABBS \mid \lambda$$
$$A \longrightarrow aB$$
$$B \longrightarrow bb \mid bB$$
$$E \longrightarrow aAb$$

$$M_0 := \{S\}$$



erreichbare Nonterminale

$$S \longrightarrow ABBS \mid \lambda$$
$$A \longrightarrow aB$$
$$B \longrightarrow bb \mid bB$$
$$E \longrightarrow aAb$$

$$M_0 := \{S\}$$

$$M_1 := M_0 \cup \{A, B, S\} = \{S, A, B\}$$

erreichbare Nonterminale

$$S \longrightarrow ABBS \mid \lambda$$
$$A \longrightarrow aB$$
$$B \longrightarrow bb \mid bB$$
$$E \longrightarrow aAb$$

$$M_0 := \{S\}$$

$$M_1 := M_0 \cup \{A, B, S\} = \{S, A, B\}$$

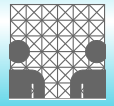
$$M_2 := M_1 \cup \{B\} = \{S, A, B\}$$

Normalformen für CFGs

- Es ist manchmal unhandlich, viele mögliche Gestalten *rechter Seiten von Regeln* haben zu können.

Normalformen für CFGs

- Es ist manchmal unhandlich, viele mögliche Gestalten *rechter Seiten von Regeln* haben zu können.
- Auf den *linken Seiten* steht bei CFGs sowieso immer *genau ein* Nonterminal.



Normalformen für CFGs

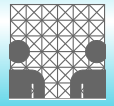
- Es ist manchmal unhandlich, viele mögliche Gestalten *rechter Seiten von Regeln* haben zu können.
- Auf den *linken Seiten* steht bei CFGs sowieso immer *genau ein* Nonterminal.
- **Normalformen** helfen bei Konstruktionen und Beweisen.

Normalformen für CFGs

- Es ist manchmal unhandlich, viele mögliche Gestalten *rechter Seiten von Regeln* haben zu können.
- Auf den *linken Seiten* steht bei CFGs sowieso immer *genau ein* Nonterminal.
- **Normalformen** helfen bei Konstruktionen und Beweisen.
- **Definition:** Eine CFG $G := (V_N, V_T, P, S)$ ist in **Chomsky-Normalform** (CNF) gdw. alle Produktionen in P von der Form $A \longrightarrow BC$ oder $A \longrightarrow a$ für $A, B, C \in V_N$ und $a \in V_T$ sind.

Normalformen für CFGs

- Es ist manchmal unhandlich, viele mögliche Gestalten *rechter Seiten von Regeln* haben zu können.
- Auf den *linken Seiten* steht bei CFGs sowieso immer *genau ein* Nonterminal.
- **Normalformen** helfen bei Konstruktionen und Beweisen.
- **Definition:** Eine CFG $G := (V_N, V_T, P, S)$ ist in **Chomsky-Normalform** (CNF) gdw. alle Produktionen in P von der Form $A \longrightarrow BC$ oder $A \longrightarrow a$ für $A, B, C \in V_N$ und $a \in V_T$ sind.
- **Definition:** Eine CFG $G := (V_N, V_T, P, S)$ ist in **Greibach-Normalform** gdw. $P \subseteq V_N \times V_T \cdot V_N^*$.

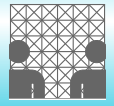


Beispiele

- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.

Beispiele

- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.
- $S \longrightarrow aA$, $A \longrightarrow aA \mid aB$, $B \longrightarrow bB \mid b$ ist in Greibach-Normalform, jedoch nicht in Chomsky-Normalform.

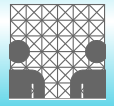


Beispiele

- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.
- $S \longrightarrow aA$, $A \longrightarrow aA \mid aB$, $B \longrightarrow bB \mid b$ ist in Greibach-Normalform, jedoch nicht in Chomsky-Normalform.
- $S \longrightarrow aAbB$, $A \longrightarrow aA \mid \lambda$, $B \longrightarrow bB \mid \lambda$ ist weder in Greibach- noch in Chomsky-Normalform.

Beispiele

- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.
- $S \longrightarrow aA$, $A \longrightarrow aA \mid aB$, $B \longrightarrow bB \mid b$ ist in Greibach-Normalform, jedoch nicht in Chomsky-Normalform.
- $S \longrightarrow aAbB$, $A \longrightarrow aA \mid \lambda$, $B \longrightarrow bB \mid \lambda$ ist weder in Greibach- noch in Chomsky-Normalform.
- Wie sieht es mit $S \longrightarrow aA$, $aA \longrightarrow aaA \mid abB$, $bB \longrightarrow bbB \mid b$ aus?



Beispiele

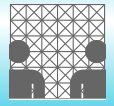
- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.
- $S \longrightarrow aA$, $A \longrightarrow aA \mid aB$, $B \longrightarrow bB \mid b$ ist in Greibach-Normalform, jedoch nicht in Chomsky-Normalform.
- $S \longrightarrow aAbB$, $A \longrightarrow aA \mid \lambda$, $B \longrightarrow bB \mid \lambda$ ist weder in Greibach- noch in Chomsky-Normalform.
- Wie sieht es mit $S \longrightarrow aA$, $aA \longrightarrow aaA \mid abB$, $bB \longrightarrow bbB \mid b$ aus?

  Diese Grammatik ist nicht einmal kontextfrei!

Beispiele

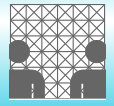
- $S \longrightarrow SS \mid AB$, $A \longrightarrow AA \mid a$, $B \longrightarrow BB \mid b$ ist in Chomsky-Normalform, aber nicht in Greibach-Normalform.
- $S \longrightarrow aA$, $A \longrightarrow aA \mid aB$, $B \longrightarrow bB \mid b$ ist in Greibach-Normalform, jedoch nicht in Chomsky-Normalform.
- $S \longrightarrow aAbB$, $A \longrightarrow aA \mid \lambda$, $B \longrightarrow bB \mid \lambda$ ist weder in Greibach- noch in Chomsky-Normalform.
- Wie sieht es mit $S \longrightarrow aA$, $aA \longrightarrow aaA \mid abB$, $bB \longrightarrow bbB \mid b$ aus?

  Diese Grammatik ist nicht einmal kontextfrei!
- $S \longrightarrow a \mid b$ ist sowohl in CNF, als auch in GNF!



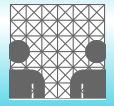
Chomsky-Normalform

- Normalformen sind nur sinnvoll, wenn sie für eine genügend große Menge von Grammatiken existieren.



Chomsky-Normalform

- Normalformen sind nur sinnvoll, wenn sie für eine genügend große Menge von Grammatiken existieren.
- **Theorem:** Zu jeder CFG G kann effektiv eine äquivalente CFG $G' := (V'_N, V'_T, P', S')$ konstruiert werden, für die folgendes gilt:



Chomsky-Normalform

- Normalformen sind nur sinnvoll, wenn sie für eine genügend große Menge von Grammatiken existieren.
- **Theorem:** Zu jeder CFG G kann effektiv eine äquivalente CFG $G' := (V'_N, V'_T, P', S')$ konstruiert werden, für die folgendes gilt:
  - Falls $\lambda \notin L(G)$, so ist G' in Chomsky-Normalform.

Chomsky-Normalform

- Normalformen sind nur sinnvoll, wenn sie für eine genügend große Menge von Grammatiken existieren.
- **Theorem:** Zu jeder CFG G kann effektiv eine äquivalente CFG $G' := (V'_N, V'_T, P', S')$ konstruiert werden, für die folgendes gilt:
  - Falls $\lambda \notin L(G)$, so ist G' in Chomsky-Normalform.
  - Gilt $\lambda \in L(G)$, so ist $S' \longrightarrow \lambda$ die einzige λ-Produktion in P' und die Produktionen in $P' \setminus \{S' \longrightarrow \lambda\} \subseteq V'_N \times (V'_N \cdot V'_N \cup V'_T)$ sind in CNF. (S' nicht auf rechter Seite!)

Chomsky-Normalform

- Normalformen sind nur sinnvoll, wenn sie für eine genügend große Menge von Grammatiken existieren.
- **Theorem:** Zu jeder CFG G kann effektiv eine äquivalente CFG $G' := (V'_N, V'_T, P', S')$ konstruiert werden, für die folgendes gilt:
  - Falls $\lambda \notin L(G)$, so ist G' in Chomsky-Normalform.
  - Gilt $\lambda \in L(G)$, so ist $S' \longrightarrow \lambda$ die einzige λ-Produktion in P' und die Produktionen in $P' \setminus \{S' \longrightarrow \lambda\} \subseteq V'_N \times (V'_N \cdot V'_N \cup V'_T)$ sind in CNF. (S' nicht auf rechter Seite!)
- **Beweis:** In mehreren Schritten werden äquivalente Grammatiken erstellt ...

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren
  3. Kettenregeln entfernen

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren
  3. Kettenregeln entfernen
  4. Ersetzen langer Terminalregeln

Konstruktion einer CNF

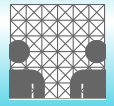
- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren
  3. Kettenregeln entfernen
  4. Ersetzen langer Terminalregeln
  5. Verkürzen zu langer Regeln

Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren
  3. Kettenregeln entfernen
  4. Ersetzen langer Terminalregeln
  5. Verkürzen zu langer Regeln
  6. Wiederherstellen der ursprünglichen Sprache durch evtl. Hinzunahme einer λ-Regel

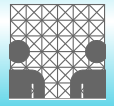
Konstruktion einer CNF

- Die Konstruktion erfolgt in sechs Schritten:
  1. λ-frei-machen
  2. Reduzieren
  3. Kettenregeln entfernen
  4. Ersetzen langer Terminalregeln
  5. Verkürzen zu langer Regeln
  6. Wiederherstellen der ursprünglichen Sprache durch evtl. Hinzunahme einer λ-Regel
- **Beispielgrammatik:**
  $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid \lambda,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow BC \mid AC$



Skript-Verfahren hier aufgeteilt!

- Zunächst die Konstruktion der CFG ohne λ-Produktionen. Dazu im korrigierten Skript – anders als im Verteilten – ein extra Theorem (1. Schritt zur Konstruktion der Chomsky- Normalform im verteilten F2-Skript)



Skript-Verfahren hier aufgeteilt!

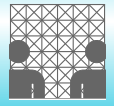
- Zunächst die Konstruktion der CFG ohne λ-Produktionen. Dazu im korrigierten Skript – anders als im Verteilten – ein extra Theorem (1. Schritt zur Konstruktion der Chomsky- Normalform im verteilten F2-Skript)
- Die wichtigen 3 Schritte der Chomsky-Normalform Konstruktion (Schritte 3,4, und 5 im verteilten F2-Skript)

Skript-Verfahren hier aufgeteilt!

- Zunächst die Konstruktion der CFG ohne λ-Produktionen. Dazu im korrigierten Skript – anders als im Verteilten – ein extra Theorem (1. Schritt zur Konstruktion der Chomsky- Normalform im verteilten F2-Skript)
- Die wichtigen 3 Schritte der Chomsky-Normalform Konstruktion (Schritte 3,4, und 5 im verteilten F2-Skript)
- Zusammenbringen der vorigen Teile (6. Schritt in der Konstruktion der Chomsky-Normalform entsprechend dem verteilten F2-Skript)

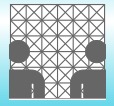
Skript-Verfahren hier aufgeteilt!

- Zunächst die Konstruktion der CFG ohne λ-Produktionen. Dazu im korrigierten Skript – anders als im Verteilten – ein extra Theorem (1. Schritt zur Konstruktion der Chomsky- Normalform im verteilten F2-Skript)
- Die wichtigen 3 Schritte der Chomsky-Normalform Konstruktion (Schritte 3,4, und 5 im verteilten F2-Skript)
- Zusammenbringen der vorigen Teile (6. Schritt in der Konstruktion der Chomsky-Normalform entsprechend dem verteilten F2-Skript)
- Eine entsprechend modifizierte Skript-Version ist ins Web gestellt!



1. λ-frei-machen

- Für jedes $A \in V_N$ entscheiden wir, ob $A \overset{*}{\underset{G}{\Longrightarrow}} \lambda$ gilt. Bestimmen der Menge $V_\lambda \subseteq V_N$:



1. λ-frei-machen

- Für jedes $A \in V_N$ entscheiden wir, ob $A \overset{*}{\underset{G}{\Longrightarrow}} \lambda$ gilt. Bestimmen der Menge $V_\lambda \subseteq V_N$:
- $M_0 := \{A \in V_N \mid A \longrightarrow \lambda \in P\}$ und

$$M_{i+1} := \{A \in V_N \mid \exists w \in M_i^* : \ A \longrightarrow w \in P\} \cup M_i$$

1. λ-frei-machen

- Für jedes $A \in V_N$ entscheiden wir, ob $A \overset{*}{\underset{G}{\Longrightarrow}} \lambda$ gilt. Bestimmen der Menge $V_\lambda \subseteq V_N$:
- $M_0 := \{A \in V_N \mid A \longrightarrow \lambda \in P\}$ und
$M_{i+1} := \{A \in V_N \mid \exists w \in M_i^* : \ A \longrightarrow w \in P\} \cup M_i$
- Es existiert ein Index $k \in I\!N$ mit $M_k = M_{k+1}$ und es ist $V_\lambda := M_k$ die gesuchte Menge.

λ-frei-machen (Forts.)

- Wir definieren nun die endliche Substitution σ:

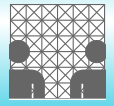
$$\sigma(A) := \begin{cases} \{A\} & \text{falls } A \in (V \setminus V_\lambda) \\ \{\lambda, A\} & \text{falls } A \in V_\lambda \end{cases}$$

λ-frei-machen (Forts.)

- Wir definieren nun die endliche Substitution σ:

$$\sigma(A) := \begin{cases} \{A\} & \text{falls } A \in (V \setminus V_\lambda) \\ \{\lambda, A\} & \text{falls } A \in V_\lambda \end{cases}$$

- $\sigma(u) = \{v \mid v$ entsteht aus u durch Streichen einiger (aller, keiner) der Symbole aus $V_\lambda\}$



λ-frei-machen (Forts.)

- Wir definieren nun die endliche Substitution σ:

$$\sigma(A) := \begin{cases} \{A\} & \text{falls } A \in (V \setminus V_\lambda) \\ \{\lambda, A\} & \text{falls } A \in V_\lambda \end{cases}$$

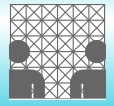
- $\sigma(u) = \{v \mid v$ entsteht aus u durch Streichen einiger (aller, keiner) der Symbole aus $V_\lambda\}$
- **Beispiel:**
  $V_\lambda := \{A, B\}$ $\quad \sigma(aABAbC) = \{aABAbC, aBAbC, aABbC, aAAbC, aAbC, aBbC, abC\}$.

λ-frei-machen (Forts.)

- Wir definieren nun die endliche Substitution σ:

$$\sigma(A) := \begin{cases} \{A\} & \text{falls } A \in (V \setminus V_\lambda) \\ \{\lambda, A\} & \text{falls } A \in V_\lambda \end{cases}$$

- $\sigma(u) = \{v \mid v$ entsteht aus u durch Streichen einiger (aller, keiner) der Symbole aus $V_\lambda\}$
- **Beispiel:**
  $V_\lambda := \{A, B\}$ $\quad \sigma(aABAbC) = \{aABAbC, aBAbC, aABbC, aAAbC, aAbC, aBbC, abC\}$.
- $G_1 := (V_N, V_T, P_1, S)$ mit $L(G_1) = L(G) \setminus \{\lambda\}$:
  $P_1 := \{A \longrightarrow v \mid v \neq \lambda \wedge v \in \sigma(w)$ für $A \longrightarrow w \in P\}$.

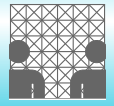


λ-frei-machen (Forts.)

- Wir definieren nun die endliche Substitution σ:

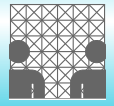
$$\sigma(A) := \begin{cases} \{A\} & \text{falls } A \in (V \setminus V_\lambda) \\ \{\lambda, A\} & \text{falls } A \in V_\lambda \end{cases}$$

- $\sigma(u) = \{v \mid v$ entsteht aus u durch Streichen einiger (aller, keiner) der Symbole aus $V_\lambda\}$
- **Beispiel:**
  $V_\lambda := \{A, B\}$ $\quad \sigma(aABAbC) = \{aABAbC, aBAbC, aABbC, aAAbC, aAbC, aBbC, abC\}$.
- $G_1 := (V_N, V_T, P_1, S)$ mit $L(G_1) = L(G) \setminus \{\lambda\}$:
  $P_1 := \{A \longrightarrow v \mid v \neq \lambda \wedge v \in \sigma(w)$ für $A \longrightarrow w \in P\}$.
- Induktionsbeweis im Skript!



Beispiel: λ-frei-machen

- $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid B,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow AC \mid BC$



Beispiel: λ-frei-machen

- $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid B,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow AC \mid BC$
- $M_0 = \{B\}$, $M_1 = \{B, A\}$, und $M_3 = \{B, A, S\}$ ergibt $V_\lambda = \{S, A, B\}$.

Beispiel: λ-frei-machen

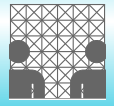
- $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid B,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow AC \mid BC$
- $M_0 = \{B\}$, $M_1 = \{B, A\}$, und $M_3 = \{B, A, S\}$ ergibt $V_\lambda = \{S, A, B\}$.
- Anwenden der Substitution aus der Konstruktion:
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B \mid \lambda,$
  $A \longrightarrow aaaA \mid B \mid \lambda \mid aaa,$
  $B \longrightarrow bB \mid \lambda \mid b,$
  $C \longrightarrow AC \mid BC \mid C$

Beispiel: λ-frei-machen

- $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid B,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow AC \mid BC$
- $M_0 = \{B\}$, $M_1 = \{B, A\}$, und $M_3 = \{B, A, S\}$ ergibt $V_\lambda = \{S, A, B\}$.
- Löschen der λ-Regeln:
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b,$
  $C \longrightarrow AC \mid BC \mid C$

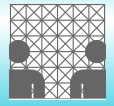
Beispiel: λ-frei-machen

- $S \longrightarrow ABC \mid AB,$
  $A \longrightarrow aaaA \mid B,$
  $B \longrightarrow bB \mid \lambda,$
  $C \longrightarrow AC \mid BC$
- $M_0 = \{B\}$, $M_1 = \{B, A\}$, und $M_3 = \{B, A, S\}$ ergibt $V_\lambda = \{S, A, B\}$.
- Löschen der λ-Regeln:
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b,$
  $C \longrightarrow AC \mid BC \mid C$
- Es gilt $L(G_{neu}) = L(G_{alt}) \setminus \{\lambda\}$.



2. Reduzieren (auch vor 1. gut)

- Wir konstruieren aus G_1 die äquivalente reduzierte CFG $G_2 := (V_{N_2}, V_{T_2}, P_2, S)$.



2. Reduzieren (auch vor 1. gut)

- Wir konstruieren aus G_1 die äquivalente reduzierte CFG $G_2 := (V_{N_2}, V_{T_2}, P_2, S)$.
- G_1 enthielt die Regeln
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B$,
  $A \longrightarrow aaaA \mid B \mid aaa$,
  $B \longrightarrow bB \mid b$,
  $C \longrightarrow AC \mid BC \mid C$

2. Reduzieren (auch vor 1. gut)

- Wir konstruieren aus G_1 die äquivalente reduzierte CFG $G_2 := (V_{N_2}, V_{T_2}, P_2, S)$.
- G_1 enthielt die Regeln
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B$,
  $A \longrightarrow aaaA \mid B \mid aaa$,
  $B \longrightarrow bB \mid b$,
  $C \longrightarrow AC \mid BC \mid C$
- A, B, und S sind produktiv:
  $S \longrightarrow AB \mid A \mid B$,
  $A \longrightarrow aaaA \mid B \mid aaa$,
  $B \longrightarrow bB \mid b$

2. Reduzieren (auch vor 1. gut)

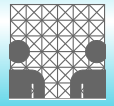
- Wir konstruieren aus G_1 die äquivalente reduzierte CFG $G_2 := (V_{N_2}, V_{T_2}, P_2, S)$.
- G_1 enthielt die Regeln
  $S \longrightarrow ABC \mid AB \mid AC \mid BC \mid C \mid A \mid B$,
  $A \longrightarrow aaaA \mid B \mid aaa$,
  $B \longrightarrow bB \mid b$,
  $C \longrightarrow AC \mid BC \mid C$
- A, B, und S sind produktiv:
  $S \longrightarrow AB \mid A \mid B$,
  $A \longrightarrow aaaA \mid B \mid aaa$,
  $B \longrightarrow bB \mid b$
- Nun sind alle Nonterminale auch noch erreichbar. Also sind wir mit dem 2. Schritt fertig.

3. Kettenregeln entfernen

- In G_2 kann es noch Produktionen der Form $A \longrightarrow B \in V_N \times V_N$ geben.

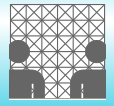
3. Kettenregeln entfernen

- In G_2 kann es noch Produktionen der Form $A \longrightarrow B \in V_N \times V_N$ geben.
- Wir definieren die Relation $\ll \subseteq V_N \times V_N$ mit: $A \ll B$ gdw. $A \longrightarrow B \in P_2$.



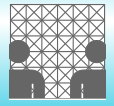
3. Kettenregeln entfernen

- In G_2 kann es noch Produktionen der Form $A \longrightarrow B \in V_N \times V_N$ geben.
- Wir definieren die Relation $\ll \subseteq V_N \times V_N$ mit: $A \ll B$ gdw. $A \longrightarrow B \in P_2$.
- $G_3 := (V_{N_3}, V_{T_3}, P_3, S)$ mit $V_{N_3} := V_{N_2}$, $V_{T_3} := V_{T_2}$ und $P_3 := \{A \longrightarrow w \mid w \notin V_{N_3} \ \wedge \ \exists B \longrightarrow w \in P_2 \ \wedge \ A \overset{*}{\ll} B\}$



3. Kettenregeln entfernen

- In G_2 kann es noch Produktionen der Form $A \longrightarrow B \in V_N \times V_N$ geben.
- Wir definieren die Relation $\ll \subseteq V_N \times V_N$ mit: $A \ll B$ gdw. $A \longrightarrow B \in P_2$.
- $G_3 := (V_{N_3}, V_{T_3}, P_3, S)$ mit $V_{N_3} := V_{N_2}$, $V_{T_3} := V_{T_2}$ und $P_3 := \{A \longrightarrow w \mid w \notin V_{N_3} \ \wedge \ \exists B \longrightarrow w \in P_2 \ \wedge \ A \overset{*}{\ll} B\}$
- Das Startsymbol bleibt S, denn sollte $L(G) = \emptyset$ gelten, so wurde bereits $P_2 = \emptyset$.

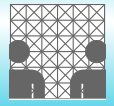


3. Kettenregeln entfernen

- In G_2 kann es noch Produktionen der Form $A \longrightarrow B \in V_N \times V_N$ geben.
- Wir definieren die Relation $\ll \subseteq V_N \times V_N$ mit: $A \ll B$ gdw. $A \longrightarrow B \in P_2$.
- $G_3 := (V_{N_3}, V_{T_3}, P_3, S)$ mit $V_{N_3} := V_{N_2}$, $V_{T_3} := V_{T_2}$ und $P_3 := \{A \longrightarrow w \mid w \notin V_{N_3} \ \wedge \ \exists B \longrightarrow w \in P_2 \ \wedge \ A \overset{*}{\ll} B\}$
- Das Startsymbol bleibt S, denn sollte $L(G) = \emptyset$ gelten, so wurde bereits $P_2 = \emptyset$.
- Die Gleichheit von $L(G_3)$ und $L(G_2)$ zeigt man auch leicht formal.

Beispiel

- $S \longrightarrow AB \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b$

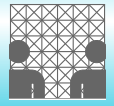


Beispiel

- $S \longrightarrow AB \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b$
- $S \ll A$, $S \ll B$ und $A \ll B$

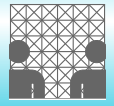
Beispiel

- $S \longrightarrow AB \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b$
- $S \ll A$, $S \ll B$ und $A \ll B$
- $S \longrightarrow AB \mid aaaA \mid aaa \mid bB \mid b,$
  $A \longrightarrow aaaA \mid bB \mid b \mid aaa,$
  $B \longrightarrow bB \mid b$



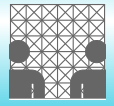
Beispiel

- $S \longrightarrow AB \mid A \mid B,$
  $A \longrightarrow aaaA \mid B \mid aaa,$
  $B \longrightarrow bB \mid b$
- $S \ll A$, $S \ll B$ und $A \ll B$
- $S \longrightarrow AB \mid aaaA \mid aaa \mid bB \mid b,$
  $A \longrightarrow aaaA \mid bB \mid b \mid aaa,$
  $B \longrightarrow bB \mid b$
- Somit sind alle Kettenregeln beseitigt!



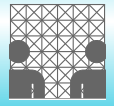
4. Ersetzen langer Terminalregeln

- Terminalsymbole dürfen in Chomsky-Normalform nur durch Produktionen der Form $A \longrightarrow a$ generiert werden.



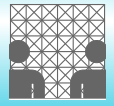
4. Ersetzen langer Terminalregeln

- Terminalsymbole dürfen in Chomsky-Normalform nur durch Produktionen der Form $A \longrightarrow a$ generiert werden.
- In allen rechten Seiten, deren Länge größer 1 ist, ersetzen wir jedes Terminal a durch ein neues Nonterminal $\langle a \rangle$.



4. Ersetzen langer Terminalregeln

- Terminalsymbole dürfen in Chomsky-Normalform nur durch Produktionen der Form $A \longrightarrow a$ generiert werden.
- In allen rechten Seiten, deren Länge größer 1 ist, ersetzen wir jedes Terminal a durch ein neues Nonterminal $\langle a \rangle$.
- Wir erweitern P_3 zu P_4 durch Hinzunahme der Produktionen $\langle a \rangle \longrightarrow a$.



4. Ersetzen langer Terminalregeln

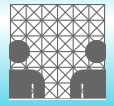
- Terminalsymbole dürfen in Chomsky-Normalform nur durch Produktionen der Form $A \longrightarrow a$ generiert werden.
- In allen rechten Seiten, deren Länge größer 1 ist, ersetzen wir jedes Terminal a durch ein neues Nonterminal $\langle a \rangle$.
- Wir erweitern P_3 zu P_4 durch Hinzunahme der Produktionen $\langle a \rangle \longrightarrow a$.
- Wir erhalten so im vierten Schritt $G_4 := (V_{N_4}, V_{T_3}, P_4, S)$.

4. Ersetzen langer Terminalregeln

- Terminalsymbole dürfen in Chomsky-Normalform nur durch Produktionen der Form $A \longrightarrow a$ generiert werden.
- In allen rechten Seiten, deren Länge größer 1 ist, ersetzen wir jedes Terminal a durch ein neues Nonterminal $\langle a \rangle$.
- Wir erweitern P_3 zu P_4 durch Hinzunahme der Produktionen $\langle a \rangle \longrightarrow a$.
- Wir erhalten so im vierten Schritt $G_4 := (V_{N_4}, V_{T_3}, P_4, S)$.
- $S \longrightarrow AB \mid \langle a \rangle\langle a \rangle\langle a \rangle A \mid \langle a \rangle\langle a \rangle\langle a \rangle \mid \langle b \rangle B \mid b,$
  $A \longrightarrow \langle a \rangle\langle a \rangle\langle a \rangle A \mid \langle b \rangle B \mid b \mid \langle a \rangle\langle a \rangle\langle a \rangle,$
  $B \longrightarrow \langle b \rangle B \mid b,$
  $\langle a \rangle \longrightarrow a, \qquad \langle b \rangle \longrightarrow b$

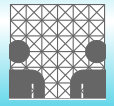
5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.



5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.
- Neue Nonterminale $\langle v \rangle$ für jeden echten Präfix v der rechten Seiten in P_4 mit $|v| \geq 2$:



5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.
- Neue Nonterminale $\langle v \rangle$ für jeden echten Präfix v der rechten Seiten in P_4 mit $|v| \geq 2$:

  $G_5 := (V_{N_5}, V_{T_3}, P_5, S)$ mit

5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.
- Neue Nonterminale $\langle v \rangle$ für jeden echten Präfix v der rechten Seiten in P_4 mit $|v| \geq 2$:

  $G_5 := (V_{N_5}, V_{T_3}, P_5, S)$ mit

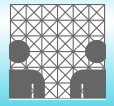
  $V_{N_5} := \{\langle v \rangle \mid \exists u \neq \lambda : \exists A \longrightarrow w \in P_4 : \ w = vu \wedge |v| \geq 2\} \cup V_{N_4}$ und $P_5 :=$

$$\{A \longrightarrow \langle v \rangle x \mid A \longrightarrow w \in P_4 : \ |w| \geq 3 \wedge w = vx \wedge x \in V_{N_4}\} \cup$$

$$\{\langle v \rangle \longrightarrow \langle u \rangle y \mid \langle u \rangle, \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge y \in V_{N_4} \wedge v = uy\} \cup$$

$$\{\langle v \rangle \longrightarrow xy \mid \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge x, y \in V_{N_4} \wedge v = xy\} \cup$$

$$\{A \longrightarrow w \mid A \longrightarrow w \in P_4 \wedge |w| \leq 2\}$$



5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.
- Neue Nonterminale $\langle v \rangle$ für jeden echten Präfix v der rechten Seiten in P_4 mit $|v| \geq 2$:

  $G_5 := (V_{N_5}, V_{T_3}, P_5, S)$ mit

  $V_{N_5} := \{\langle v \rangle \mid \exists u \neq \lambda : \exists A \longrightarrow w \in P_4 : \ w = vu \wedge |v| \geq 2\} \cup V_{N_4}$ und $P_5 :=$

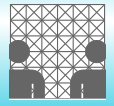
$\{A \longrightarrow \langle v \rangle x \mid A \longrightarrow w \in P_4 : \ |w| \geq 3 \wedge w = vx \wedge x \in V_{N_4}\} \cup$

$\{\langle v \rangle \longrightarrow \langle u \rangle y \mid \langle u \rangle, \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge y \in V_{N_4} \wedge v = uy\} \cup$

$\{\langle v \rangle \longrightarrow xy \mid \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge x, y \in V_{N_4} \wedge v = xy\} \cup$

$\{A \longrightarrow w \mid A \longrightarrow w \in P_4 \wedge |w| \leq 2\}$

- G_5 ist in Chomsky-Normalform!!



5. Verkürzen zu langer Regeln

- Es kann noch Regeln $A \longrightarrow w$ mit $|w| > 2$ geben.
- Neue Nonterminale $\langle v \rangle$ für jeden echten Präfix v der rechten Seiten in P_4 mit $|v| \geq 2$:

  $G_5 := (V_{N_5}, V_{T_3}, P_5, S)$ mit

  $V_{N_5} := \{\langle v \rangle \mid \exists u \neq \lambda : \exists A \longrightarrow w \in P_4 : \ w = vu \wedge |v| \geq 2\} \cup V_{N_4}$ und $P_5 :=$

$\{A \longrightarrow \langle v \rangle x \mid A \longrightarrow w \in P_4 : \ |w| \geq 3 \wedge w = vx \wedge x \in V_{N_4}\} \cup$

$\{\langle v \rangle \longrightarrow \langle u \rangle y \mid \langle u \rangle, \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge y \in V_{N_4} \wedge v = uy\} \cup$

$\{\langle v \rangle \longrightarrow xy \mid \langle v \rangle \in (V_{N_5} \setminus V_{N_4}) \wedge x, y \in V_{N_4} \wedge v = xy\} \cup$

$\{A \longrightarrow w \mid A \longrightarrow w \in P_4 \wedge |w| \leq 2\}$

- G_5 ist in Chomsky-Normalform!!
- $L(G_5) = L(G)$, falls $\lambda \notin L(G)$ gilt.

Beispiel

- $S \longrightarrow AB \mid \langle a\rangle\langle a\rangle\langle a\rangle A \mid \langle a\rangle\langle a\rangle\langle a\rangle \mid \langle b\rangle B \mid b,$
  $A \longrightarrow \langle a\rangle\langle a\rangle\langle a\rangle A \mid \langle b\rangle B \mid b \mid \langle a\rangle\langle a\rangle\langle a\rangle,$
  $B \longrightarrow \langle b\rangle B \mid b,$
  $\langle a\rangle \longrightarrow a,$
  $\langle b\rangle \longrightarrow b$

Beispiel

- $S \longrightarrow AB \mid \langle a\rangle\langle a\rangle\langle a\rangle A \mid \langle a\rangle\langle a\rangle\langle a\rangle \mid \langle b\rangle B \mid b,$
  $A \longrightarrow \langle a\rangle\langle a\rangle\langle a\rangle A \mid \langle b\rangle B \mid b \mid \langle a\rangle\langle a\rangle\langle a\rangle,$
  $B \longrightarrow \langle b\rangle B \mid b,$
  $\langle a\rangle \longrightarrow a,$
  $\langle b\rangle \longrightarrow b$
- . . . wird zu:
  $S \longrightarrow AB \mid \langle\langle a\rangle\langle a\rangle\langle a\rangle\rangle A \mid \langle\langle a\rangle\langle a\rangle\rangle\langle a\rangle \mid \langle b\rangle B \mid b,$
  $A \longrightarrow \langle\langle a\rangle\langle a\rangle\langle a\rangle\rangle A \mid \langle b\rangle B \mid b \mid \langle\langle a\rangle\langle a\rangle\rangle\langle a\rangle,$
  $B \longrightarrow \langle b\rangle B \mid b,$
  $\langle a\rangle \longrightarrow a,$
  $\langle b\rangle \longrightarrow b,$
  $\langle\langle a\rangle\langle a\rangle\langle a\rangle\rangle \longrightarrow \langle\langle a\rangle\langle a\rangle\rangle\langle a\rangle,$
  $\langle\langle a\rangle\langle a\rangle\rangle \longrightarrow \langle a\rangle\langle a\rangle$

6. Wiederherstellen von λ

- Im ersten Schritt wurde die Menge V_λ zur Ausgangsgrammatik G bestimmt.

6. Wiederherstellen von λ

- Im ersten Schritt wurde die Menge V_λ zur Ausgangsgrammatik G bestimmt.
- Dort konnte also die Frage „$\lambda \in L(G)$?“ auf die Frage „$S \in V_\lambda$?“ reduziert, und damit entschieden werden.

6. Wiederherstellen von λ

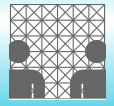
- Im ersten Schritt wurde die Menge V_λ zur Ausgangsgrammatik G bestimmt.
- Dort konnte also die Frage „$\lambda \in L(G)$?“ auf die Frage „$S \in V_\lambda$?“ reduziert, und damit entschieden werden.
- Gilt also $\lambda \in L(G)$, so konstruieren wir G_6, indem wir zu den Nonterminalen von G_5 ein neues Startsymbol S_{neu}, sowie die neuen Produktionen $S_{\text{neu}} \longrightarrow \lambda$ und $\{S_{\text{neu}} \longrightarrow w \mid \exists S \longrightarrow w \in P_5\}$ zu P_5 hinzufügen. (Nur erforderlich, wenn S in rechten Seiten von Produktionen vorkommt!)

6. Wiederherstellen von λ

- Im ersten Schritt wurde die Menge V_λ zur Ausgangsgrammatik G bestimmt.
- Dort konnte also die Frage „$\lambda \in L(G)$?“ auf die Frage „$S \in V_\lambda$?“ reduziert, und damit entschieden werden.
- Gilt also $\lambda \in L(G)$, so konstruieren wir G_6, indem wir zu den Nonterminalen von G_5 ein neues Startsymbol S_{neu}, sowie die neuen Produktionen $S_{\text{neu}} \longrightarrow \lambda$ und $\{S_{\text{neu}} \longrightarrow w \mid \exists S \longrightarrow w \in P_5\}$ zu P_5 hinzufügen. (Nur erforderlich, wenn S in rechten Seiten von Produktionen vorkommt!)
- Es werden die Regeln $S_{\text{neu}} \longrightarrow \lambda$, $S_{\text{neu}} \longrightarrow AB \mid \langle\langle a\rangle\langle a\rangle\langle a\rangle\rangle A \mid \langle\langle a\rangle\langle a\rangle\rangle\langle a\rangle \mid \langle b\rangle B \mid b$ hinzugefügt.

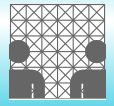
Linksrekursion

- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.



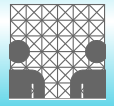
Linksrekursion

- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.
- $A \longrightarrow Av$ heißt **linksrekursiv**.



Linksrekursion

- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.
- $A \longrightarrow Av$ heißt **linksrekursiv**.
- **Theorem:** Zu jeder CFG G gibt es eine äquivalente CFG G' ohne Linksrekursion.



Linksrekursion

- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.
- $A \longrightarrow Av$ heißt **linksrekursiv**.
- **Theorem:** Zu jeder CFG G gibt es eine äquivalente CFG G' ohne Linksrekursion.
- **Beweisidee:** Habe $G := (V_N, V_T, P, S)$ die linksrekursiven A-Produktionen $A \longrightarrow Au_1 \mid Au_2 \mid \ldots \mid Au_r$ und die verbleibenden A-Produktionen $A \longrightarrow v_1 \mid v_2 \mid \ldots \mid v_s$.

Linksrekursion

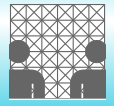
- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.
- $A \longrightarrow Av$ heißt **linksrekursiv**.
- **Theorem:** Zu jeder CFG G gibt es eine äquivalente CFG G' ohne Linksrekursion.
- **Beweisidee:** Habe $G := (V_N, V_T, P, S)$ die linksrekursiven A-Produktionen $A \longrightarrow Au_1 \mid Au_2 \mid \ldots \mid Au_r$ und die verbleibenden A-Produktionen $A \longrightarrow v_1 \mid v_2 \mid \ldots \mid v_s$.
  - Definiere CFG $G' := (V'_N, V'_T, P', S')$ mit neuem Nonterminal $\overline{A}$ und ersetze alle linksrekursiven A-Produktionen von G durch:

Linksrekursion

- Jede Regel $A \longrightarrow w$ heißt A-**Produktion**.
- $A \longrightarrow Av$ heißt **linksrekursiv**.
- **Theorem:** Zu jeder CFG G gibt es eine äquivalente CFG G' ohne Linksrekursion.
- **Beweisidee:** Habe $G := (V_N, V_T, P, S)$ die linksrekursiven A-Produktionen $A \longrightarrow Au_1 \mid Au_2 \mid \ldots \mid Au_r$ und die verbleibenden A-Produktionen $A \longrightarrow v_1 \mid v_2 \mid \ldots \mid v_s$.
  - Definiere CFG $G' := (V'_N, V'_T, P', S')$ mit neuem Nonterminal $\overline{A}$ und ersetze alle linksrekursiven A-Produktionen von G durch:
  - $A \longrightarrow v_1\overline{A} \mid v_2\overline{A} \mid \ldots \mid v_s\overline{A}$ und $\overline{A} \longrightarrow u_1\overline{A} \mid u_2\overline{A} \mid \ldots \mid u_r\overline{A} \mid u_1 \mid u_2 \mid \ldots \mid u_r$.

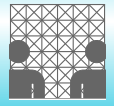
Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.



Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:



Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:
  1. Zunächst wird G in eine CFG G_1 umgeformt, die in erweiterter Chomsky-Normalform ist.

Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:
  1. Zunächst wird G in eine CFG G_1 umgeformt, die in erweiterter Chomsky-Normalform ist.
  2. Nehmen wir nun an, dass o.B.d.A. $V_{N_1} := \{A_1, A_2, \ldots, A_n\}$ und $S_1 := A_1$ ist.

Greibach-Normalform

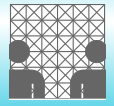
- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:
  1. Zunächst wird G in eine CFG G_1 umgeformt, die in erweiterter Chomsky-Normalform ist.
  2. Nehmen wir nun an, dass o.B.d.A. $V_{N_1} := \{A_1, A_2, \ldots, A_n\}$ und $S_1 := A_1$ ist.
  3. Durch sukkzessive Veränderung der Regelmenge P_1 wird $G_2 := (V_{N_2}, V_T, P_2, S_2)$ konstruiert, für die aus $A_i \longrightarrow A_j w \in P_2$ stets $j > i$ folgt.

Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:
  1. Zunächst wird G in eine CFG G_1 umgeformt, die in erweiterter Chomsky-Normalform ist.
  2. Nehmen wir nun an, dass o.B.d.A. $V_{N_1} := \{A_1, A_2, \ldots, A_n\}$ und $S_1 := A_1$ ist.
  3. Durch sukkzessive Veränderung der Regelmenge P_1 wird $G_2 := (V_{N_2}, V_T, P_2, S_2)$ konstruiert, für die aus $A_i \longrightarrow A_j w \in P_2$ stets $j > i$ folgt.
  4. Dann sukzessives Ersetzen der A_j beginnend mit dem größten j.

Greibach-Normalform

- **Theorem:** Zu jeder CFG G kann effektiv eine CFG G' in Greibach-Normalform konstruiert werden, für die $L(G') = L(G) \setminus \{\lambda\}$ gilt.
- **Beweis:** Konstruiere G' in mehreren Schritten:
  1. Zunächst wird G in eine CFG G_1 umgeformt, die in erweiterter Chomsky-Normalform ist.
  2. Nehmen wir nun an, dass o.B.d.A. $V_{N_1} := \{A_1, A_2, \ldots, A_n\}$ und $S_1 := A_1$ ist.
  3. Durch sukkzessive Veränderung der Regelmenge P_1 wird $G_2 := (V_{N_2}, V_T, P_2, S_2)$ konstruiert, für die aus $A_i \longrightarrow A_j w \in P_2$ stets $j > i$ folgt.
  4. Dann sukzessives Ersetzen der A_j beginnend mit dem größten j.
  5. Beweis im Skript, hier dennoch ein Hinweis zum



Algorithmus A

begin

for $k := 1$ **to** n **do**

for $j := 1$ **to** $k - 1$ **do** (* *j ist stets kleiner k !!* *)

for jede Produktion der Form $A_k \longrightarrow A_j u$ **do**

begin

for alle Produktionen $A_j \longrightarrow v$ **do**

füge neue Produktion $A_k \longrightarrow vu$ hinzu

und entferne die Produktion $A_k \longrightarrow A_j u$;

end

end $(*$ *for* Produktionen $A_k \longrightarrow A_j u$ $*)$

end $(*$ *for j* $*)$

for jede Produktion der Form $A_k \longrightarrow A_k u$ **do**

begin

definiere neues Nonterminal B_k und ergänze neue Produktionen

$B_k \longrightarrow u$ und $B_k \longrightarrow uB_k$ und

entferne die Produktion $A_k \longrightarrow A_k u$

end;

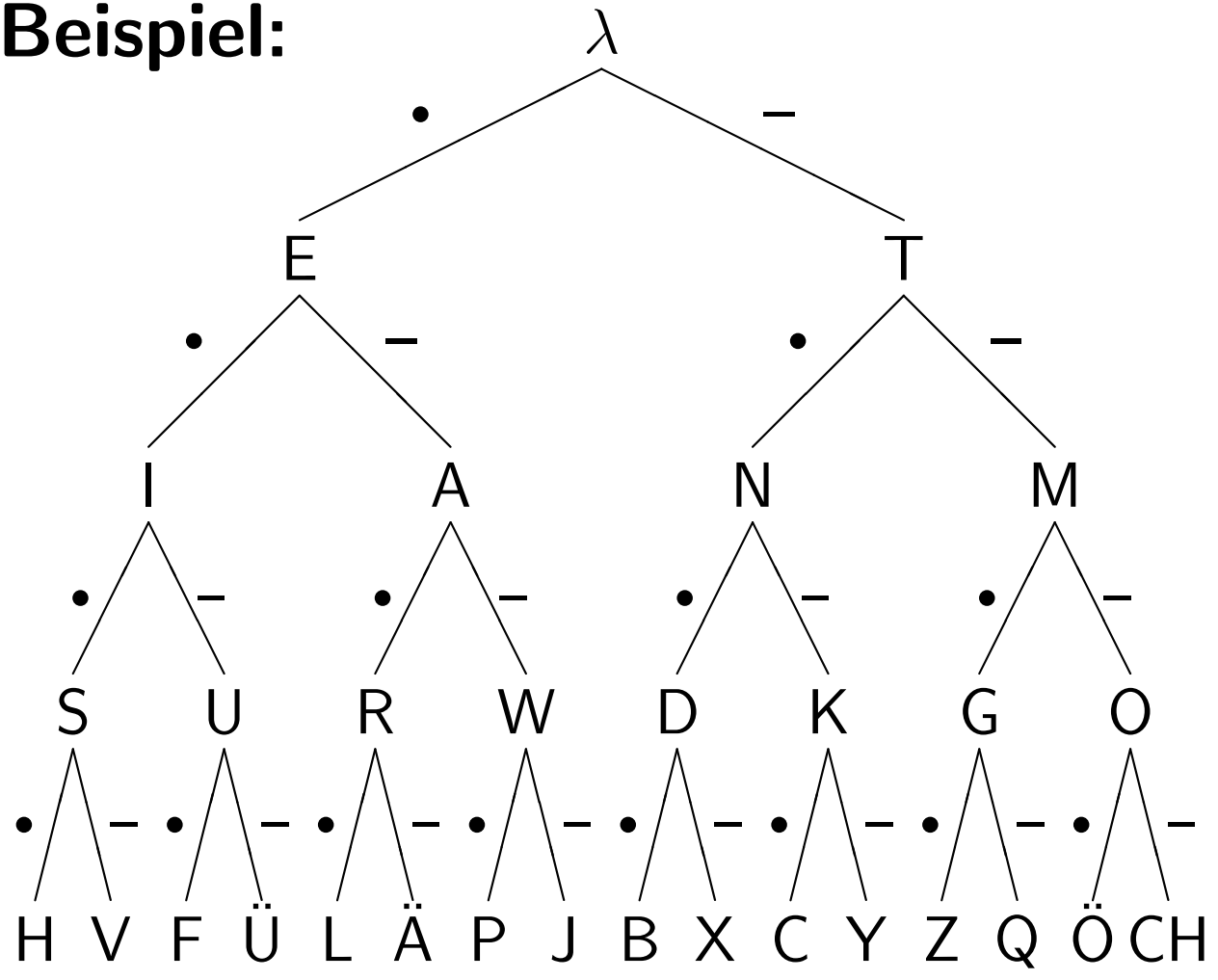
for jede Produktion der Form $A_k \longrightarrow u$,

Bäume

- Ein **Baum** ist ein gerichteter, geordneter, zyklenfreier und knotenmarkierter Graph mit einer **Wurzel**, die keinen Vorgängerknoten besitzt.

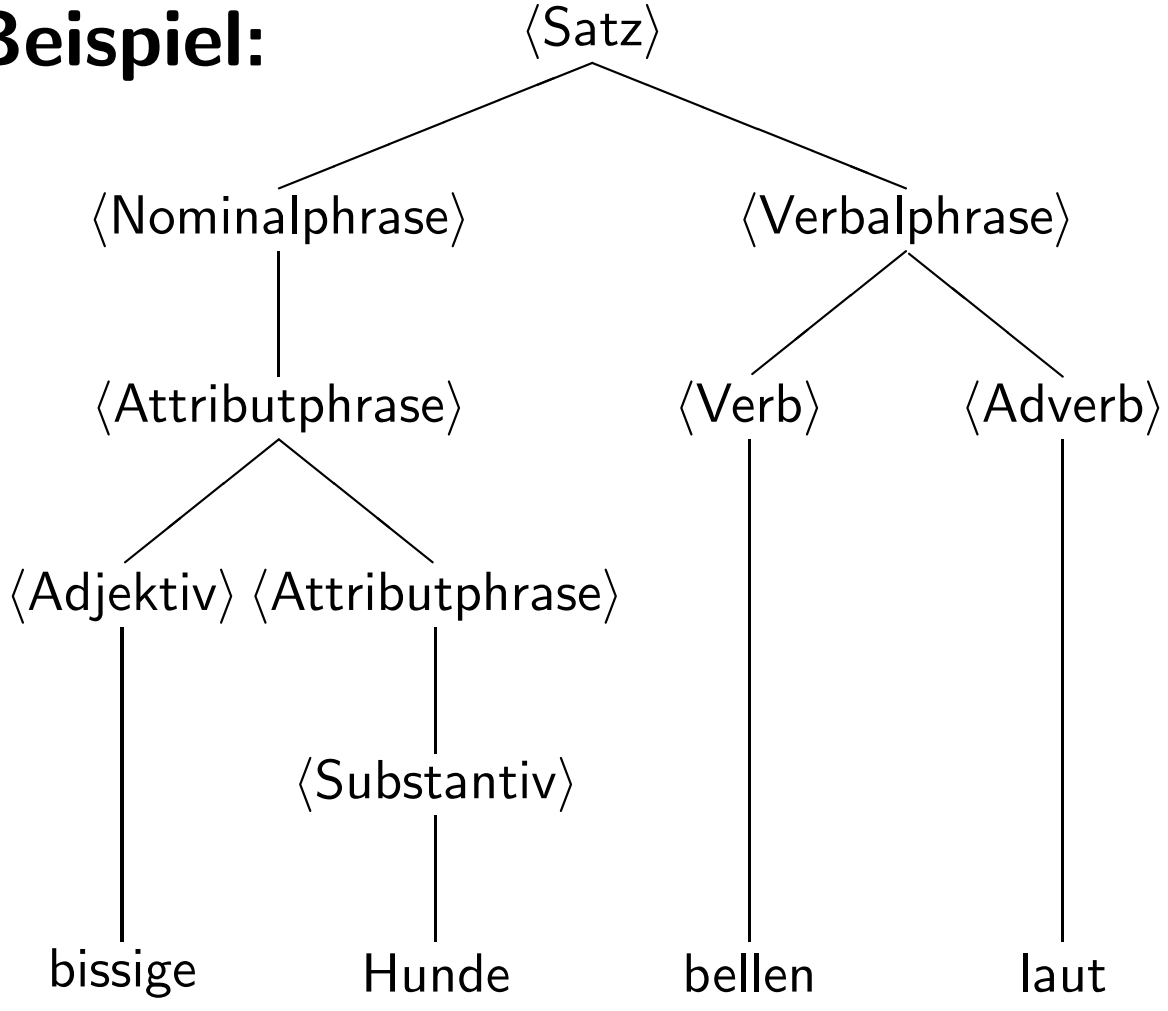
Bäume

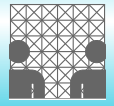
- Ein **Baum** ist ein gerichteter, geordneter, zyklenfreier und knotenmarkierter Graph mit einer **Wurzel**, die keinen Vorgängerknoten besitzt.
- **Beispiel:**



Bäume

- Ein **Baum** ist ein gerichteter, geordneter, zyklenfreier und knotenmarkierter Graph mit einer **Wurzel**, die keinen Vorgängerknoten besitzt.
- **Beispiel:**





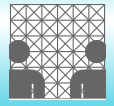
Ableitungsbaum

- Sei $G := (V_N, V_T, P, S)$ eine beliebige CFG, dann wird für jedes Symbol $A \in V$ ein A-**Ableitungsbaum** $B(A)$ wie folgt erklärt:

Ableitungsbaum

- Sei $G := (V_N, V_T, P, S)$ eine beliebige CFG, dann wird für jedes Symbol $A \in V$ ein A-**Ableitungsbaum** $B(A)$ wie folgt erklärt:

  1. Jedes Blatt von $B(A)$ ist mit einem Symbol $a \in V_T \cup \{\lambda\}$ beschriftet.



Ableitungsbaum

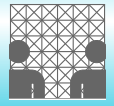
- Sei $G := (V_N, V_T, P, S)$ eine beliebige CFG, dann wird für jedes Symbol $A \in V$ ein A-**Ableitungsbaum** $B(A)$ wie folgt erklärt:

  1. Jedes Blatt von $B(A)$ ist mit einem Symbol $a \in V_T \cup \{\lambda\}$ beschriftet.

  2. Jeder Knoten von $B(A)$, der kein Blatt ist, ist mit einem Symbol $\Sigma \in V_N$, die Wurzel mit dem Symbol A beschriftet.

Ableitungsbaum

- Sei $G := (V_N, V_T, P, S)$ eine beliebige CFG, dann wird für jedes Symbol $A \in V$ ein A-**Ableitungsbaum** $B(A)$ wie folgt erklärt:
  1. Jedes Blatt von $B(A)$ ist mit einem Symbol $a \in V_T \cup \{\lambda\}$ beschriftet.
  2. Jeder Knoten von $B(A)$, der kein Blatt ist, ist mit einem Symbol $\Sigma \in V_N$, die Wurzel mit dem Symbol A beschriftet.
  3. Ein (innerer) Knoten ist mit dem Symbol $\Sigma \in V_N$ beschriftet gdw. seine Nachfolgerknoten von links nach rechts mit $\Sigma_1, \Sigma_2, \ldots, \Sigma_k$ beschriftet sind und $\Sigma \longrightarrow \Sigma_1\Sigma_2 \ldots \Sigma_k$ eine Produktion von G ist.



Beispiel: Dyck-Sprache

- Betrachte die CFG G_1 mit folgenden Produktionen:

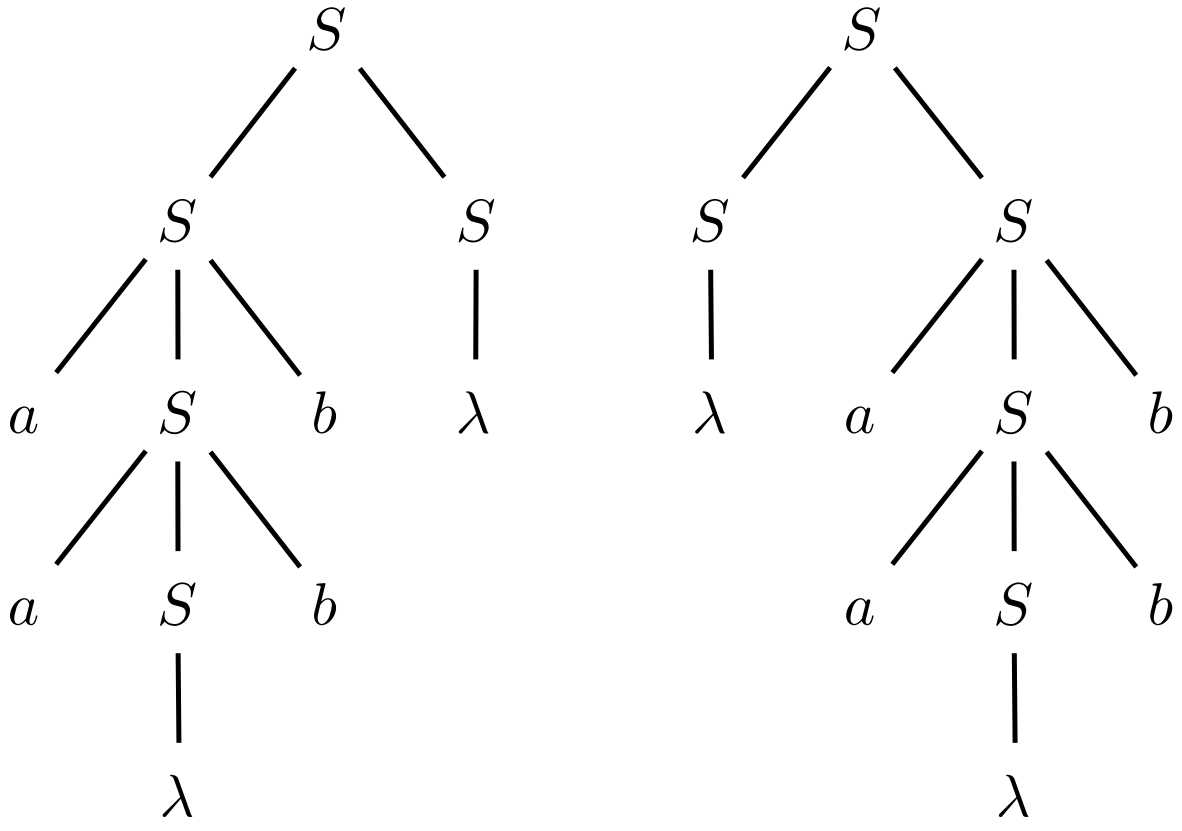
$$S \longrightarrow SS \mid aSb \mid \lambda$$

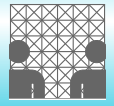
Beispiel: Dyck-Sprache

- Betrachte die CFG G_1 mit folgenden Produktionen:

$$S \longrightarrow SS \mid aSb \mid \lambda$$

- Zwei unterschiedliche Ableitungsbäume für $aabb$:



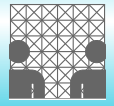


Beispiel: Dyck-Sprache

- Betrachte die CFG G_1 mit folgenden Produktionen:

$$S \longrightarrow SS \mid aSb \mid \lambda$$

- Zwei unterschiedliche Ableitungsbäume für $aabb$:
- $L(G_1)$ ist die **Dyck-Sprache** D_1 aller korrekt geklammerten Ausdrücke über einem Klammerpaar.



Beispiel: Dyck-Sprache

- Betrachte die CFG G_1 mit folgenden Produktionen:

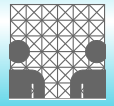
$$S \longrightarrow SS \mid aSb \mid \lambda$$

- Zwei unterschiedliche Ableitungsbäume für $aabb$:
- $L(G_1)$ ist die **Dyck-Sprache** D_1 aller korrekt geklammerten Ausdrücke über einem Klammerpaar.
- Eine äquivalente **eindeutige** Grammatik:

$$S \longrightarrow aSbS \mid \lambda$$

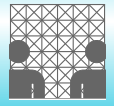
Definition: Mehrdeutigkeit

- Eine CFG G heißt **mehrdeutig** (*ambiguous*) gdw. es für mindestens ein Wort $w \in L(G)$ zwei verschiedene Ableitungsbäume gibt. Andernfalls heißt G **eindeutig**.



Definition: Mehrdeutigkeit

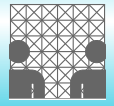
- Eine CFG G heißt **mehrdeutig** (*ambiguous*) gdw. es für mindestens ein Wort $w \in L(G)$ zwei verschiedene Ableitungsbäume gibt. Andernfalls heißt G **eindeutig**.
- Eine kontextfreie Sprache $L \in \mathcal{C}f$ heißt **eindeutig** (*unambiguous*) gdw. es eine eindeutige CFG G mit $L = L(G)$ gibt. Andernfalls heißt L **mehrdeutig**.



Definition: Mehrdeutigkeit

- Eine CFG G heißt **mehrdeutig** (*ambiguous*) gdw. es für mindestens ein Wort $w \in L(G)$ zwei verschiedene Ableitungsbäume gibt. Andernfalls heißt G **eindeutig**.
- Eine kontextfreie Sprache $L \in \mathcal{C}f$ heißt **eindeutig** (*unambiguous*) gdw. es eine eindeutige CFG G mit $L = L(G)$ gibt. Andernfalls heißt L **mehrdeutig**.
- Eine Ableitung in einer CFG heißt **Linksableitung** (bzw. **Rechtsableitung**) gdw. in jedem Schritt der Ableitung das ersetzte Zeichen das am weitesten links (bzw. rechts) stehende Nonterminal ist.

  (Ableitungsrelationen $\underset{\text{li}}{\Longrightarrow}$ und $\underset{\text{re}}{\Longrightarrow}$)



Definition: Mehrdeutigkeit

- Eine CFG G heißt **mehrdeutig** (*ambiguous*) gdw. es für mindestens ein Wort $w \in L(G)$ zwei verschiedene Ableitungsbäume gibt. Andernfalls heißt G **eindeutig**.
- Eine kontextfreie Sprache $L \in \mathcal{C}f$ heißt **eindeutig** (*unambiguous*) gdw. es eine eindeutige CFG G mit $L = L(G)$ gibt. Andernfalls heißt L **mehrdeutig**.
- Eine Ableitung in einer CFG heißt **Linksableitung** (bzw. **Rechtsableitung**) gdw. in jedem Schritt der Ableitung das ersetzte Zeichen das am weitesten links (bzw. rechts) stehende Nonterminal ist.

  (Ableitungsrelationen $\underset{\text{li}}{\Longrightarrow}$ und $\underset{\text{re}}{\Longrightarrow}$)
- $L_M := \{a^r b^s c^t \mid \exists r, s, t \in I\!N : \ r = s \vee s = t\}$ ist *nicht* eindeutig.

Ausblick

- Ist jede reguläre Sprache auch durch eine kontextfreie Grammatik erzeugbar?

Ausblick

- Ist jede reguläre Sprache auch durch eine kontextfreie Grammatik erzeugbar?
- Eigenschaften der kontextfreien Sprachen

Ausblick

- Ist jede reguläre Sprache auch durch eine kontextfreie Grammatik erzeugbar?
- Eigenschaften der kontextfreien Sprachen
- Grenzen der kontextfreien Sprachen

Ausblick

- Ist jede reguläre Sprache auch durch eine kontextfreie Grammatik erzeugbar?
- Eigenschaften der kontextfreien Sprachen
- Grenzen der kontextfreien Sprachen
- Kellerautomaten