

F2 — Automaten und formale Sprachen

Matthias Jantzen

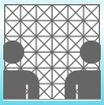
(nach und mit Folienvorlagen von Berndt Farwer)

Fachbereich Informatik

AB „Theoretische Grundlagen der Informatik“ (TGI)

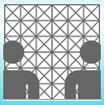
Universität Hamburg

jantzen@informatik.uni-hamburg.de



Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt



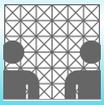
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*),$



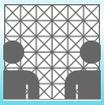
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und



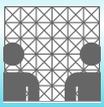
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und
 - **rechtslin**ear, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cup V_T^*)$ ist.



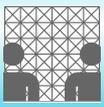
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und
 - **rechtslin**ear, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cup V_T^*)$ ist.
- Eine CFG wird genau dann **einseitig linear** genannt, wenn sie entweder linkslinear oder rechtslinear ist.



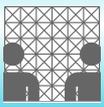
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und
 - **rechtslin**ear, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cup V_T^*)$ ist.
- Eine CFG wird genau dann **einseitig linear** genannt, wenn sie entweder linkslinear oder rechtslinear ist.
- **Beispiel:** $S \longrightarrow aSa \mid \lambda$ ist linear.



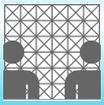
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und
 - **rechtslin**ear, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cup V_T^*)$ ist.
- Eine CFG wird genau dann **einseitig linear** genannt, wenn sie entweder linkslinear oder rechtslinear ist.
- **Beispiel:** $S \longrightarrow aSa \mid \lambda$ ist linear.
- **Beispiel:** $S \longrightarrow aaA \mid aa$ ist rechtslinear.



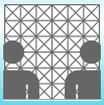
Definition: Lineare Grammatik

- Eine kontextfreie Grammatik $G := (V_N, V_T, P, S)$ heißt
 - **linear**, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cdot V_T^* \cup V_T^*)$,
 - **linkslin**ear, falls $P \subseteq V_N \times (V_N \cdot V_T^* \cup V_T^*)$, und
 - **rechtslin**ear, falls $P \subseteq V_N \times (V_T^* \cdot V_N \cup V_T^*)$ ist.
- Eine CFG wird genau dann **einseitig linear** genannt, wenn sie entweder linkslinear oder rechtslinear ist.
- **Beispiel:** $S \longrightarrow aSa \mid \lambda$ ist linear.
- **Beispiel:** $S \longrightarrow aaA \mid aa$ ist rechtslinear.
- **Beispiel:** $S \longrightarrow Aaa \mid aa$ ist linkslinear.



Rechtslineare Grammatiken

- **Theorem:** $R \subseteq \Sigma^*$ ist regulär gdw. es eine rechtslineare Grammatik G gibt, mit $L(G) = R$.



Rechtslineare Grammatiken

- **Theorem:** $R \subseteq \Sigma^*$ ist regulär gdw. es eine rechtslineare Grammatik G gibt, mit $L(G) = R$.
- **Beweis \subseteq :** Sei $R \in \mathcal{Reg}$ definiert durch einen buchstabierenden NFA $A_1 = (Z, \Sigma, K, \{z_0\}, Z_{\text{end}})$, dann konstruiere eine rechtslineare CFG $G_R := (V_N, V_T, P, S)$:

$$V_N := \{[z] \mid z \in Z\}$$

$$V_T := \Sigma$$

$$S := [z_0]$$

$$P := \{[z] \longrightarrow a[z'] \mid (z, a, z') \in K\} \cup \{[z] \longrightarrow \lambda \mid z \in Z_{\text{end}}\}$$

(etwas einfacher und allgemeiner als im Skript)



Rechtslineare Grammatiken

- **Theorem:** $R \subseteq \Sigma^*$ ist regulär gdw. es eine rechtslineare Grammatik G gibt, mit $L(G) = R$.
- **Beweis \supseteq :** Sei CFG $G := (V_N, V_T, P, S)$ rechtslinear. Wir definieren einen NFA $A_2 := (Z, \Sigma, K, Z_{\text{start}}, Z_{\text{end}})$ durch

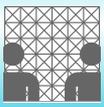
$$Z := \{z_A \mid A \in V_N\} \cup \{z_\lambda\}$$

$$\Sigma := V_T$$

$$Z_{\text{start}} := \{z_S\}$$

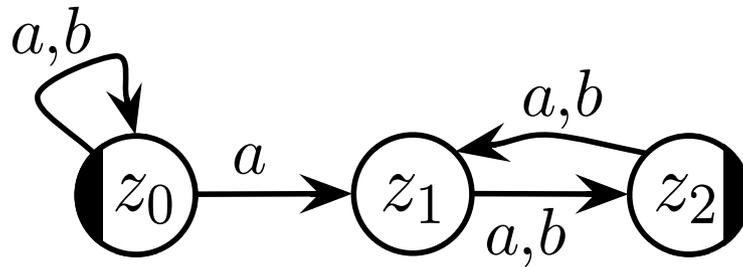
$$Z_{\text{end}} := \{z_\lambda\}$$

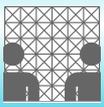
$$K := \{(z_Q, u, z_R) \mid Q, R \in V_N \wedge u \in V_T^* \wedge Q \longrightarrow uR \in P\} \\ \cup \{(z_Q, u, z_\lambda) \mid Q \in V_N \wedge u \in V_T^* \wedge Q \longrightarrow u \in P\}$$



Beispiel: NFA \rightarrow RLG

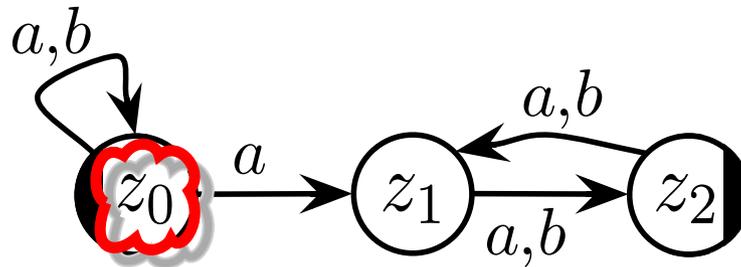
- Wir konstruieren zu einem *endlichen Automaten* eine *rechtslineare Grammatik*:



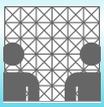


Beispiel: NFA \rightarrow RLG

- Wir konstruieren zu einem *endlichen Automaten* eine *rechtslineare Grammatik*:

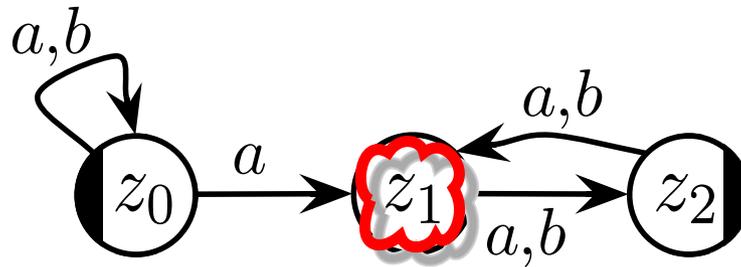


$$[z_0] \longrightarrow a[z_0] \mid b[z_0] \mid a[z_1]$$



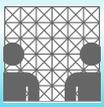
Beispiel: NFA \rightarrow RLG

- Wir konstruieren zu einem *endlichen Automaten* eine *rechtslineare Grammatik*:



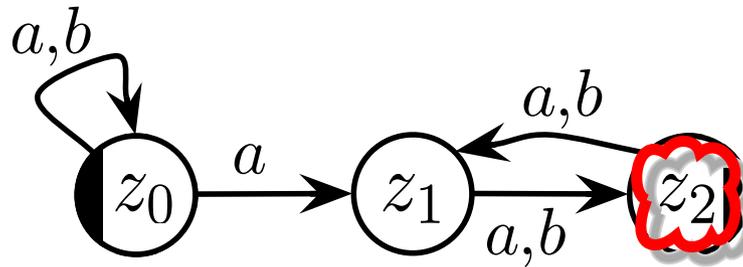
$$[z_0] \longrightarrow a[z_0] \mid b[z_0] \mid a[z_1]$$

$$[z_1] \longrightarrow a[z_2] \mid b[z_2]$$



Beispiel: NFA \rightarrow RLG

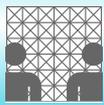
- Wir konstruieren zu einem *endlichen Automaten* eine *rechtslineare Grammatik*:



$$[z_0] \longrightarrow a[z_0] \mid b[z_0] \mid a[z_1]$$

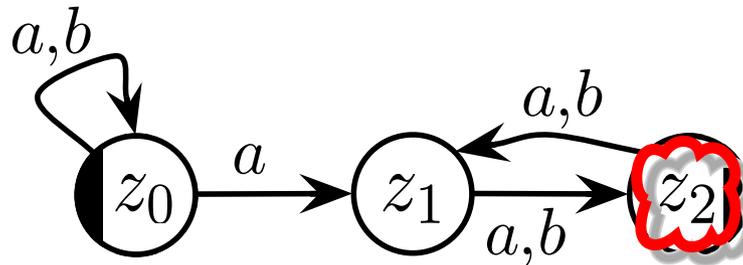
$$[z_1] \longrightarrow a[z_2] \mid b[z_2]$$

$$[z_2] \longrightarrow a[z_1] \mid b[z_1] \mid \lambda$$



Beispiel: NFA \rightarrow RLG

- Wir konstruieren zu einem *endlichen Automaten* eine *rechtslineare Grammatik*:

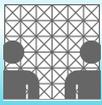


$$[z_0] \longrightarrow a[z_0] \mid b[z_0] \mid a[z_1]$$

$$[z_1] \longrightarrow a[z_2] \mid b[z_2]$$

$$[z_2] \longrightarrow a[z_1] \mid b[z_1] \mid \lambda$$

$$G = (\{[z_0], [z_1], [z_2]\}, \{a, b\}, P, [z_0])$$



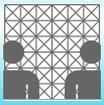
Beispiel: RLG \rightarrow NFA

- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$



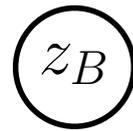
Beispiel: RLG \rightarrow NFA

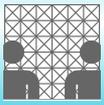
- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$





Beispiel: RLG \rightarrow NFA

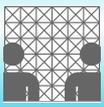
- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$





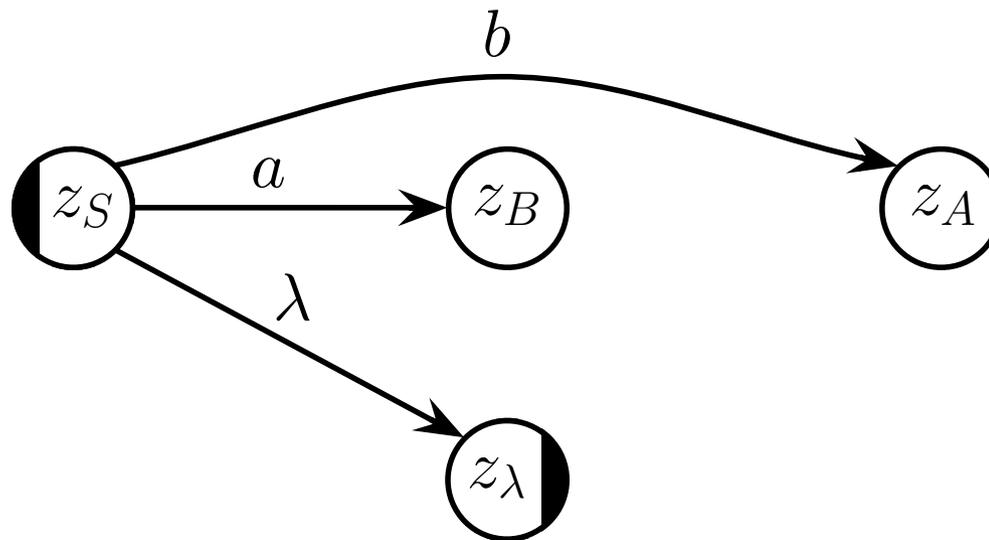
Beispiel: RLG \rightarrow NFA

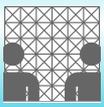
- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$





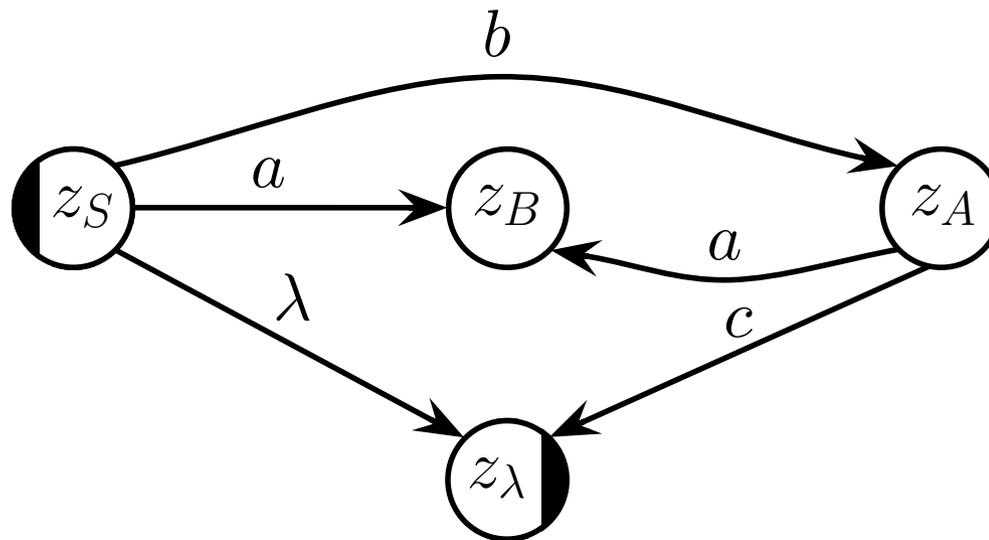
Beispiel: RLG \rightarrow NFA

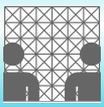
- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$





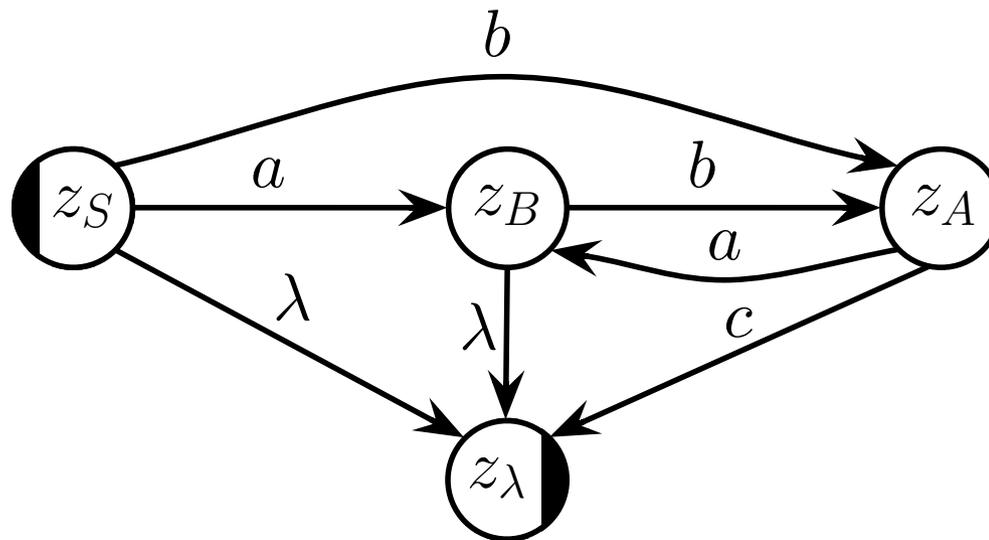
Beispiel: RLG \rightarrow NFA

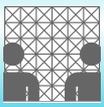
- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$





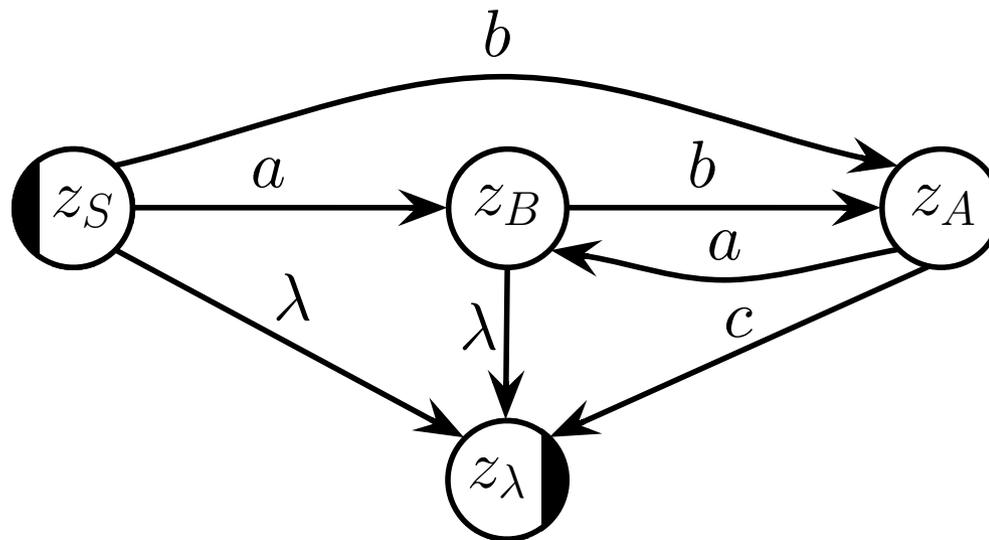
Beispiel: RLG \rightarrow NFA

- Von einer *rechtslinearen Grammatik* zu einem *NFA*:

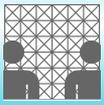
$$S \longrightarrow aB \mid bA \mid \lambda$$

$$A \longrightarrow aB \mid c$$

$$B \longrightarrow bA \mid \lambda$$

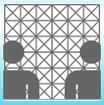


$$A = (\{z_S, z_A, z_B, z_\lambda\}, \{a, b\}, K, \{z_S\}, \{z_\lambda\})$$



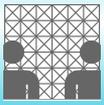
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.



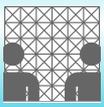
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!



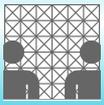
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.



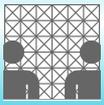
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.



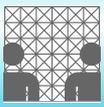
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.
- **Fragen:**



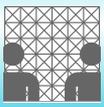
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.
- **Fragen:**
 - Wo sind die Grenzen von *Cf*?



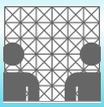
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.
- **Fragen:**
 - Wo sind die Grenzen von *Cf*?
 - Was für Abschlusseigenschaften hat *Cf*?



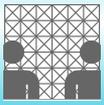
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.
- **Fragen:**
 - Wo sind die Grenzen von *Cf*?
 - Was für Abschlusseigenschaften hat *Cf*?
 - Was für Entscheidbarkeitsresultate kennen wir?



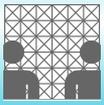
Ergebnisse

- Auch für **linkslinere** Grammatiken lassen sich „äquivalente“ NFAs definieren.
- Allgemein gilt dies nicht für **lineare** Grammatiken, sondern nur für **einseitig lineare** Grammatiken!
- **Beispiel:** $S \longrightarrow aSb \mid \lambda$ erzeugt die nicht-reguläre Sprache *DUP*.
- ... also ist die Familie *Reg* eine echte Teilmenge von *Cf*.
- **Fragen:**
 - Wo sind die Grenzen von *Cf*?
 - Was für Abschlusseigenschaften hat *Cf*?
 - Was für Entscheidbarkeitsresultate kennen wir?
 - Gibt es ein Automatenmodell für *Cf*?



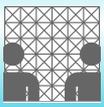
Pumping-Lemma für Cf

- Eine kontextfreie Grammatik besitzt nur endlich viele Nonterminale n .



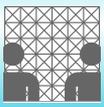
Pumping-Lemma für Cf

- Eine kontextfreie Grammatik besitzt nur endlich viele Nonterminale n .
- Ist in einem Ableitungsbaum ein Pfad länger als n , so kommt ein Nonterminal doppelt vor.



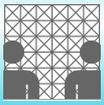
Pumping-Lemma für Cf

- Eine kontextfreie Grammatik besitzt nur endlich viele Nonterminale n .
- Ist in einem Ableitungsbaum ein Pfad länger als n , so kommt ein Nonterminal doppelt vor.
- Daraus ergeben sich weitere Ableitungsbäume, die ebenfalls zu Terminalwörtern führen!



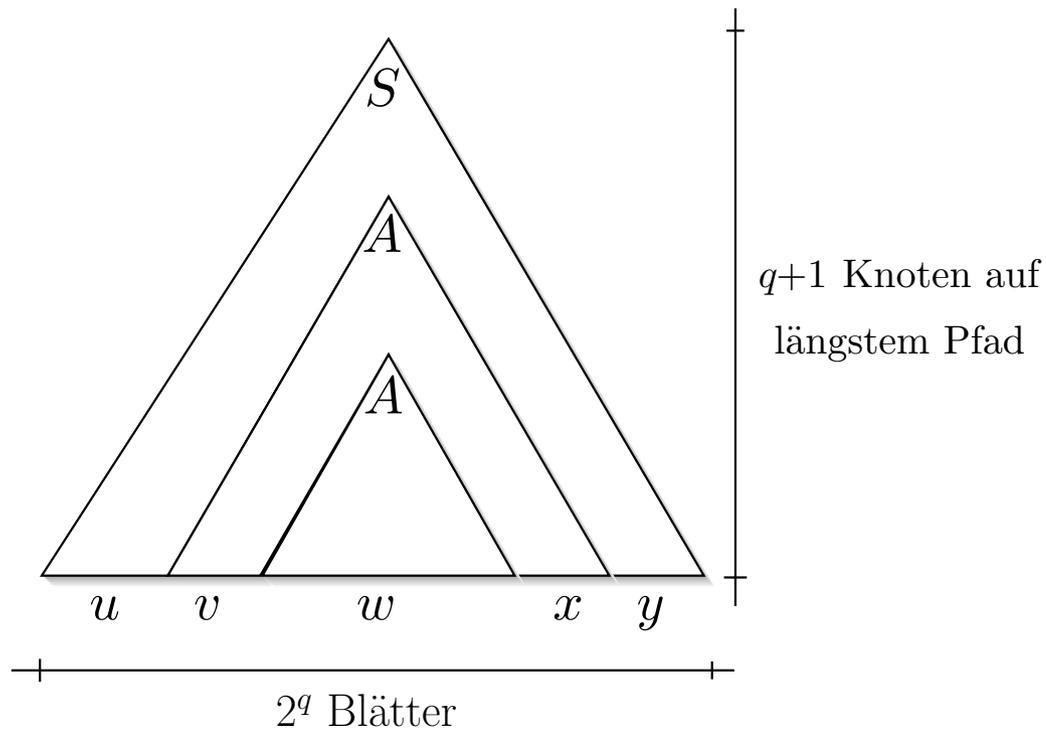
Pumping-Lemma für Cf

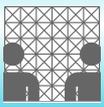
- Eine kontextfreie Grammatik besitzt nur endlich viele Nonterminale n .
- Ist in einem Ableitungsbaum ein Pfad länger als n , so kommt ein Nonterminal doppelt vor.
- Daraus ergeben sich weitere Ableitungsbäume, die ebenfalls zu Terminalwörtern führen!
- Ableitungsbäume von CNF-Grammatiken haben besonders schöne Eigenschaften.



Pumping-Lemma (Beweisidee)

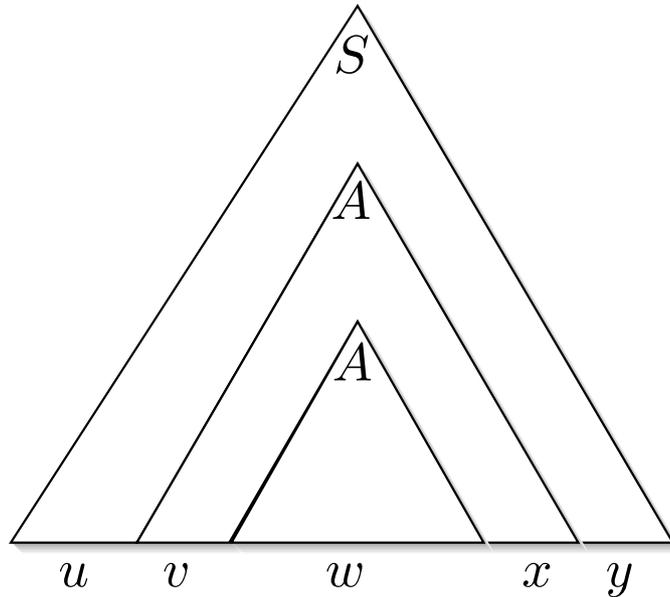
- **Gegeben:** eine CFG G in CNF

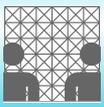




Pumping-Lemma (Beweisidee)

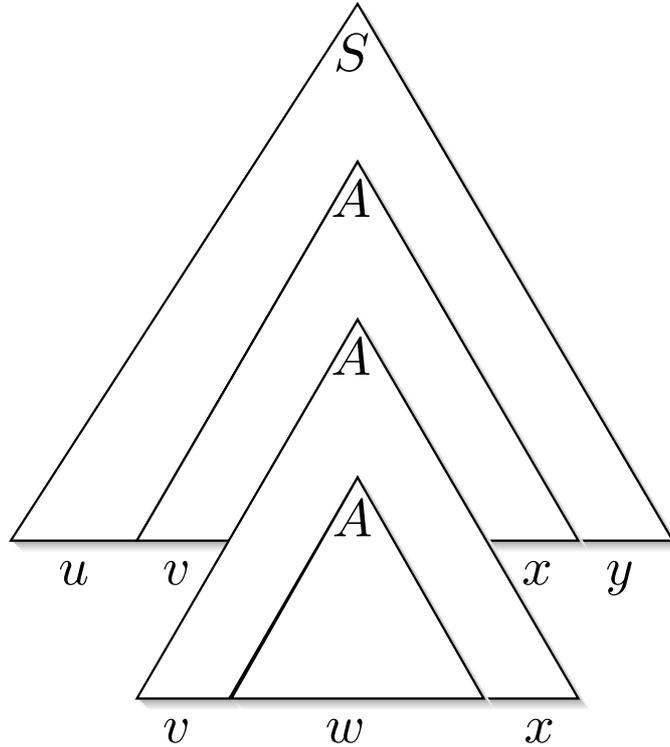
- **Gegeben:** eine CFG G in CNF

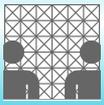




Pumping-Lemma (Beweisidee)

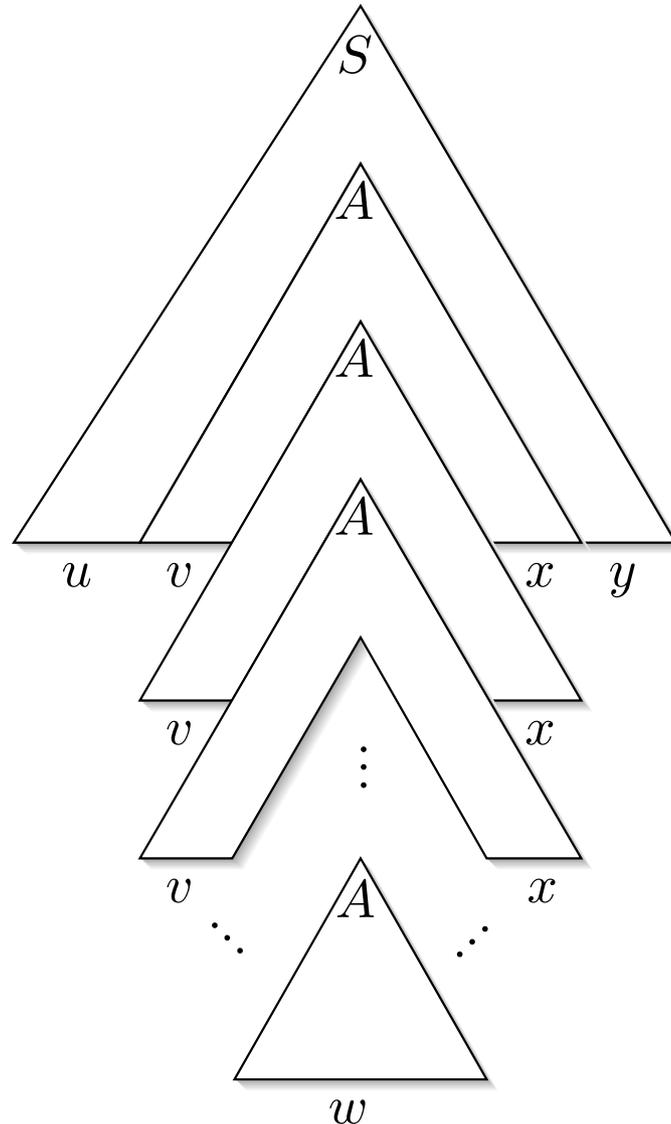
- **Gegeben:** eine CFG G in CNF

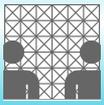




Pumping-Lemma (Beweisidee)

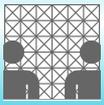
- **Gegeben:** eine CFG G in CNF





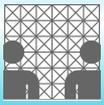
Pumping-Lemma ($uvwxy$ -Theorem)

- **Theorem:** Für jede Sprache $L \in \mathcal{Cf}$ gibt es eine Zahl $n \in \mathbb{N}$, so dass jedes Wort $z \in L$ mit $|z| \geq n$ eine Zerlegung $z = uvwxy$ besitzt, für die folgendes gilt:



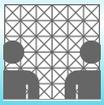
Pumping-Lemma ($uvwxy$ -Theorem)

- **Theorem:** Für jede Sprache $L \in \mathcal{Cf}$ gibt es eine Zahl $n \in \mathbb{N}$, so dass jedes Wort $z \in L$ mit $|z| \geq n$ eine Zerlegung $z = uvwxy$ besitzt, für die folgendes gilt:
 - (i) $|vx| \geq 1$



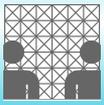
Pumping-Lemma ($uvwxy$ -Theorem)

- **Theorem:** Für jede Sprache $L \in \mathcal{Cf}$ gibt es eine Zahl $n \in \mathbb{N}$, so dass jedes Wort $z \in L$ mit $|z| \geq n$ eine Zerlegung $z = uvwxy$ besitzt, für die folgendes gilt:
 - (i) $|vx| \geq 1$
 - (ii) $|vwx| \leq n$



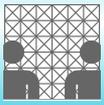
Pumping-Lemma ($uvwxy$ -Theorem)

- **Theorem:** Für jede Sprache $L \in \mathcal{Cf}$ gibt es eine Zahl $n \in \mathbb{N}$, so dass jedes Wort $z \in L$ mit $|z| \geq n$ eine Zerlegung $z = uvwxy$ besitzt, für die folgendes gilt:
 - (i) $|vx| \geq 1$
 - (ii) $|vwx| \leq n$
 - (iii) $\forall i \geq 0 : uv^iwx^iy \in L$



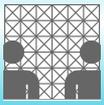
Pumping-Lemma ($uvwxy$ -Theorem)

- **Theorem:** Für jede Sprache $L \in \mathcal{Cf}$ gibt es eine Zahl $n \in \mathbb{N}$, so dass jedes Wort $z \in L$ mit $|z| \geq n$ eine Zerlegung $z = uvwxy$ besitzt, für die folgendes gilt:
 - (i) $|vx| \geq 1$
 - (ii) $|vwx| \leq n$
 - (iii) $\forall i \geq 0 : uv^iwx^iy \in L$
- Wie beim uvw -Theorem, kann hiermit nicht gezeigt werden, dass eine Sprache kontextfrei ist!!!



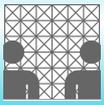
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.



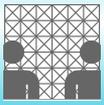
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.



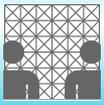
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.



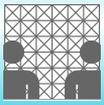
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.
- Da $z \in L$ und $|z| \geq k$ gelten, muss eine Aufspaltung $z = uvwxy$ mit $|vwx| \leq k$ und $|vx| \geq 1$ existieren, so dass $uv^i wx^i y \in L$ für alle $i \in \mathbb{N}$.



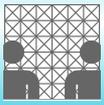
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.
- Da $z \in L$ und $|z| \geq k$ gelten, muss eine Aufspaltung $z = uvwxy$ mit $|vwx| \leq k$ und $|vx| \geq 1$ existieren, so dass $uv^i wx^i y \in L$ für alle $i \in \mathbb{N}$.
- Alle Aufspaltungen führen zum Widerspruch:



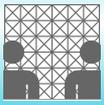
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.
- Da $z \in L$ und $|z| \geq k$ gelten, muss eine Aufspaltung $z = uvwxy$ mit $|vwx| \leq k$ und $|vx| \geq 1$ existieren, so dass $uv^i wx^i y \in L$ für alle $i \in \mathbb{N}$.
- Alle Aufspaltungen führen zum Widerspruch:
 - v oder x enthält a 's und b 's



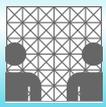
Beispiel

- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.
- Da $z \in L$ und $|z| \geq k$ gelten, muss eine Aufspaltung $z = uvwxy$ mit $|vwx| \leq k$ und $|vx| \geq 1$ existieren, so dass $uv^i wx^i y \in L$ für alle $i \in \mathbb{N}$.
- Alle Aufspaltungen führen zum Widerspruch:
 - v oder x enthält a 's und b 's
 - v oder x enthält b 's und c 's



Beispiel

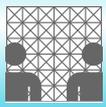
- $L := \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Angenommen L wäre kontextfrei, dann gäbe es ein $k \in \mathbb{N}$ für das die Folgerung aus dem $uvwxy$ -Theorem zutrifft.
- Wähle $z := a^k b^k c^k$.
- Da $z \in L$ und $|z| \geq k$ gelten, muss eine Aufspaltung $z = uvwxy$ mit $|vwx| \leq k$ und $|vx| \geq 1$ existieren, so dass $uv^i wx^i y \in L$ für alle $i \in \mathbb{N}$.
- Alle Aufspaltungen führen zum Widerspruch:
 - v oder x enthält a 's und b 's
 - v oder x enthält b 's und c 's
 - v oder x enthält nur a 's, nur b 's oder nur c 's



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

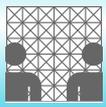
• $\underbrace{a^k}_{\text{red dot}} b^{k+1} c^k d^{k+1}$, d.h. vw besteht nur aus a 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

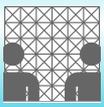
- $\underbrace{a^k}_{\text{red dot}} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

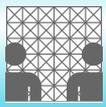
- $\underbrace{a^k}_{vwx} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{vwx} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.
- $a^k \underbrace{b^{k+1}}_{vwx} c^k d^{k+1}$, d.h. vwx besteht nur aus b 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

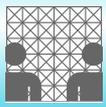
- $\underbrace{a^k}_{vwx} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{vwx} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.
- $a^k \underbrace{b^{k+1}}_{vwx} c^k d^{k+1}$, d.h. vwx besteht nur aus b 's.
- $a^k \underbrace{b^{k+1} c^k}_{vwx} d^{k+1}$, d.h. vwx besteht aus b 's und c 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

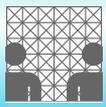
- $\underbrace{a^k}_{\text{red dot}} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.
- $a^k \underbrace{b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht nur aus b 's.
- $a^k b^{k+1} \underbrace{c^k}_{\text{red dot}} d^{k+1}$, d.h. vwx besteht aus b 's und c 's.
- $a^k b^{k+1} \underbrace{c^k d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht nur aus c 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

- $\underbrace{a^k}_{\text{red dot}} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.
- $a^k \underbrace{b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht nur aus b 's.
- $a^k b^{k+1} \underbrace{c^k}_{\text{red dot}} d^{k+1}$, d.h. vwx besteht aus b 's und c 's.
- $a^k b^{k+1} c^k \underbrace{d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht nur aus c 's.
- $a^k b^{k+1} \underbrace{c^k d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht aus c 's und d 's.



Beispiel: $\{a^n b^m c^n d^m \mid n, m \in \mathbb{N}\}$

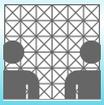
Sei k die Zahl aus dem Pumping-Lemma. Für $z := a^k b^{k+1} c^k d^{k+1}$ gibt es keine Aufspaltung $z = uvwxy$ mit $\forall i \in \mathbb{N} : uv^i wx^i y \in L$.

- $\underbrace{a^k}_{\text{red dot}} b^{k+1} c^k d^{k+1}$, d.h. vwx besteht nur aus a 's.
- $\underbrace{a^k b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht aus a 's und b 's.
- $a^k \underbrace{b^{k+1}}_{\text{red dot}} c^k d^{k+1}$, d.h. vwx besteht nur aus b 's.
- $a^k b^{k+1} \underbrace{c^k}_{\text{red dot}} d^{k+1}$, d.h. vwx besteht aus b 's und c 's.
- $a^k b^{k+1} \underbrace{c^k d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht nur aus c 's.
- $a^k b^{k+1} \underbrace{c^k d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht aus c 's und d 's.
- $a^k b^{k+1} c^k \underbrace{d^{k+1}}_{\text{red dot}}$, d.h. vwx besteht nur aus d 's.



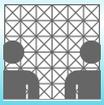
Entscheidbarkeitsresultate

- **Theorem:** Das **spezielle Wortproblem** für kontextfreie Sprachen ist entscheidbar, d.h. es gibt ein Verfahren, das zu einer durch eine CFG G spezifizierten Sprache $L = L(G)$ für jedes w feststellt, ob $w \in L$ gilt.



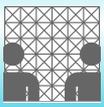
Entscheidbarkeitsresultate

- **Theorem:** Das **spezielle Wortproblem** für kontextfreie Sprachen ist entscheidbar, d.h. es gibt ein Verfahren, das zu einer durch eine CFG G spezifizierten Sprache $L = L(G)$ für jedes w feststellt, ob $w \in L$ gilt.
- **Beweisidee:** O.B.d.A. liege G in GNF vor. Da in einer Ableitung $S \xRightarrow{*} v$ in jedem Schritt ein weiteres Terminal erzeugt wird, brauchen nur die endlich vielen Ableitungen der Länge $|w|$ daraufhin überprüft zu werden, ob eine darunter ist, die das Wort w generiert.



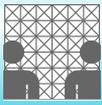
Entscheidbarkeitsresultate

- **Theorem:** Das **spezielle Wortproblem** für kontextfreie Sprachen ist entscheidbar, d.h. es gibt ein Verfahren, das zu einer durch eine CFG G spezifizierten Sprache $L = L(G)$ für jedes w feststellt, ob $w \in L$ gilt.
- **Beweisidee:** O.B.d.A. liege G in GNF vor. Da in einer Ableitung $S \xRightarrow{*} v$ in jedem Schritt ein weiteres Terminal erzeugt wird, brauchen nur die endlich vielen Ableitungen der Länge $|w|$ daraufhin überprüft zu werden, ob eine darunter ist, die das Wort w generiert.
- Das Verfahren benötigt **exponentiellen Aufwand!**



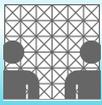
Entscheidbarkeitsresultate

- **Theorem:** Das **spezielle Wortproblem** für kontextfreie Sprachen ist entscheidbar, d.h. es gibt ein Verfahren, das zu einer durch eine CFG G spezifizierten Sprache $L = L(G)$ für jedes w feststellt, ob $w \in L$ gilt.
- **Beweisidee:** O.B.d.A. liege G in GNF vor. Da in einer Ableitung $S \xRightarrow{*} v$ in jedem Schritt ein weiteres Terminal erzeugt wird, brauchen nur die endlich vielen Ableitungen der Länge $|w|$ daraufhin überprüft zu werden, ob eine darunter ist, die das Wort w generiert.
- Das Verfahren benötigt **exponentiellen** Aufwand!
- Entscheidung des **allgemeinen Wortproblems** erfordert zusätzlich die Konstruktion einer äquivalenten GNF.



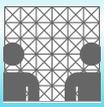
Spezielles Wortproblem

- Das **spezielle Wortproblem** für CNF-Grammatiken kann auch mit polynomiellem Aufwand entschieden werden!



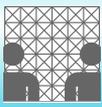
Spezielles Wortproblem

- Das **spezielle Wortproblem** für CNF-Grammatiken kann auch mit polynomiellem Aufwand entschieden werden!
- Sei $G = (V_N, V_T, P, S)$ CFG in CNF und $w \in V_T^*$ mit $w = x_1x_2 \dots x_n$, für $x_i \in V_T$.



Spezielles Wortproblem

- Das **spezielle Wortproblem** für CNF-Grammatiken kann auch mit polynomiellem Aufwand entschieden werden!
- Sei $G = (V_N, V_T, P, S)$ CFG in CNF und $w \in V_T^*$ mit $w = x_1x_2 \dots x_n$, für $x_i \in V_T$.
- Für alle $i, j \in \{0, 1, 2, \dots, n\}$ mit $0 \leq i \leq j$ definieren wir Mengen $V_{i,j} \subseteq V_N$ durch
$$V_{i-1,j} := \{A \in V_N \mid A \xrightarrow{*} x_i x_{i+1} \dots x_j\},$$
 die sukzessive als die Elemente der Felder $V_{i,j}$ einer $(n+1) \times (n+1)$ -Matrix mit Elementen aus V bestimmt werden.



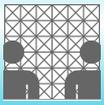
Spezielles Wortproblem

- Das **spezielle Wortproblem** für CNF-Grammatiken kann auch mit polynomiellem Aufwand entschieden werden!
- Sei $G = (V_N, V_T, P, S)$ CFG in CNF und $w \in V_T^*$ mit $w = x_1x_2 \dots x_n$, für $x_i \in V_T$.
- Für alle $i, j \in \{0, 1, 2, \dots, n\}$ mit $0 \leq i \leq j$ definieren wir Mengen $V_{i,j} \subseteq V_N$ durch
$$V_{i-1,j} := \{A \in V_N \mid A \xrightarrow{*} x_i x_{i+1} \dots x_j\},$$
 die sukzessive als die Elemente der Felder $V_{i,j}$ einer $(n+1) \times (n+1)$ -Matrix mit Elementen aus V bestimmt werden.
- Am Ende ist $V_{0,n}$ bestimmt, und es gilt $S \in V_{0,n}$ genau dann, wenn $w \in L(G)$ ist.



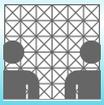
Abschlusseigenschaften

- Die Familie \mathcal{C}_f ist abgeschlossen gegenüber:



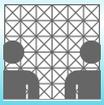
Abschlusseigenschaften

- Die Familie \mathcal{C}_f ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}_f \vee \mathcal{C}_f \subseteq \mathcal{C}_f$



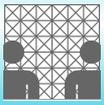
Abschlusseigenschaften

- Die Familie \mathcal{C}_f ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}_f \vee \mathcal{C}_f \subseteq \mathcal{C}_f$
 2. Komplexprodukt, d.h. $\mathcal{C}_f \cdot \mathcal{C}_f \subseteq \mathcal{C}_f$



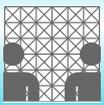
Abschlusseigenschaften

- Die Familie \mathcal{C}_f ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}_f \vee \mathcal{C}_f \subseteq \mathcal{C}_f$
 2. Komplexprodukt, d.h. $\mathcal{C}_f \cdot \mathcal{C}_f \subseteq \mathcal{C}_f$
 3. Kleene'sche Hülle (Sternbildung), d.h. $\mathcal{C}_f^* \subseteq \mathcal{C}_f$



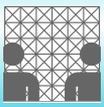
Abschlusseigenschaften

- Die Familie \mathcal{C}_f ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}_f \vee \mathcal{C}_f \subseteq \mathcal{C}_f$
 2. Komplexprodukt, d.h. $\mathcal{C}_f \cdot \mathcal{C}_f \subseteq \mathcal{C}_f$
 3. Kleene'sche Hülle (Sternbildung), d.h. $\mathcal{C}_f^* \subseteq \mathcal{C}_f$
- **Beweis:** Für $i \in \{1, 2\}$ seien L_i gegeben durch $L_i := L(G_i)$ für $G_i := (V_{i,N}, V_{i,T}, P_i, S_i)$.



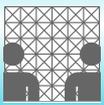
Abschlusseigenschaften

- Die Familie $\mathcal{C}f$ ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}f \vee \mathcal{C}f \subseteq \mathcal{C}f$
 2. Komplexprodukt, d.h. $\mathcal{C}f \cdot \mathcal{C}f \subseteq \mathcal{C}f$
 3. Kleene'sche Hülle (Sternbildung), d.h. $\mathcal{C}f^* \subseteq \mathcal{C}f$
- **Beweis:** Für $i \in \{1, 2\}$ seien L_i gegeben durch $L_i := L(G_i)$ für $G_i := (V_{i,N}, V_{i,T}, P_i, S_i)$.
 1. $L_1 \cup L_2 = L(G_3)$ für
$$G_3 := (V_{1,N} \uplus V_{2,N} \uplus \{S_3\}, V_{1,T} \cup V_{2,T}, P_3, S_3)$$
 mit
$$P_3 := P_1 \cup P_2 \cup \{S_3 \longrightarrow S_1, S_3 \longrightarrow S_2\}$$



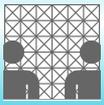
Abschlusseigenschaften

- Die Familie $\mathcal{C}f$ ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}f \vee \mathcal{C}f \subseteq \mathcal{C}f$
 2. Komplexprodukt, d.h. $\mathcal{C}f \cdot \mathcal{C}f \subseteq \mathcal{C}f$
 3. Kleene'sche Hülle (Sternbildung), d.h. $\mathcal{C}f^* \subseteq \mathcal{C}f$
- **Beweis:** Für $i \in \{1, 2\}$ seien L_i gegeben durch $L_i := L(G_i)$ für $G_i := (V_{i,N}, V_{i,T}, P_i, S_i)$.
 1. $L_1 \cup L_2 = L(G_3)$ für
 $G_3 := (V_{1,N} \uplus V_{2,N} \uplus \{S_3\}, V_{1,T} \cup V_{2,T}, P_3, S_3)$ mit
 $P_3 := P_1 \cup P_2 \cup \{S_3 \longrightarrow S_1, S_3 \longrightarrow S_2\}$
 2. $L_1 \cdot L_2 = L(G_4)$ für
 $G_4 := (V_{1,N} \uplus V_{2,N} \uplus \{S_4\}, V_{1,T} \cup V_{2,T}, P_4, S_4)$ mit
 $P_4 := P_1 \cup P_2 \cup \{S_4 \longrightarrow S_1 S_2\}$



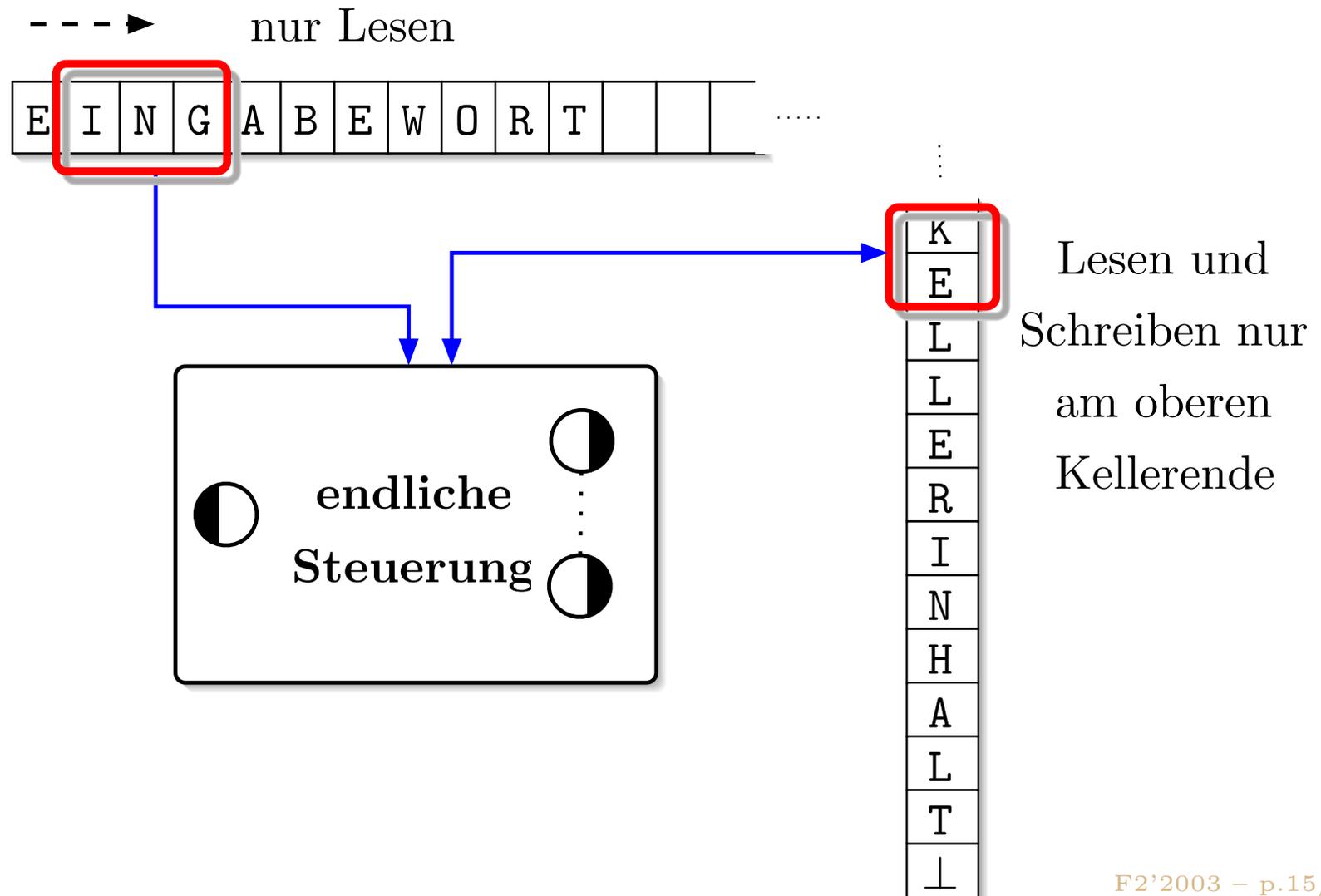
Abschlusseigenschaften

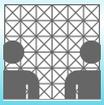
- Die Familie $\mathcal{C}f$ ist abgeschlossen gegenüber:
 1. Vereinigung, d.h. $\mathcal{C}f \vee \mathcal{C}f \subseteq \mathcal{C}f$
 2. Komplexprodukt, d.h. $\mathcal{C}f \cdot \mathcal{C}f \subseteq \mathcal{C}f$
 3. Kleene'sche Hülle (Sternbildung), d.h. $\mathcal{C}f^* \subseteq \mathcal{C}f$
- **Beweis:** Für $i \in \{1, 2\}$ seien L_i gegeben durch $L_i := L(G_i)$ für $G_i := (V_{i,N}, V_{i,T}, P_i, S_i)$.
 1. $L_1 \cup L_2 = L(G_3)$ für
 $G_3 := (V_{1,N} \uplus V_{2,N} \uplus \{S_3\}, V_{1,T} \cup V_{2,T}, P_3, S_3)$ mit
 $P_3 := P_1 \cup P_2 \cup \{S_3 \longrightarrow S_1, S_3 \longrightarrow S_2\}$
 2. $L_1 \cdot L_2 = L(G_4)$ für
 $G_4 := (V_{1,N} \uplus V_{2,N} \uplus \{S_4\}, V_{1,T} \cup V_{2,T}, P_4, S_4)$ mit
 $P_4 := P_1 \cup P_2 \cup \{S_4 \longrightarrow S_1 S_2\}$
 3. $L_1^* = L(G_5)$ für $G_5 := (V_{1,N} \uplus \{S_5\}, V_{1,T}, P_5, S_5)$ mit
 $P_5 := P_1 \cup \{S_5 \longrightarrow S_1 S_5, S_5 \longrightarrow \lambda\}$



Kellerautomaten

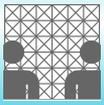
- Eine Kelerautomat ist ein endlicher Automat mit Kellerspeicher:





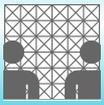
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:



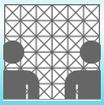
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.



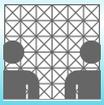
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.



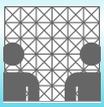
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.
 - Γ ist endliches **Kelleralphabet**.



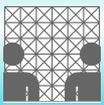
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.
 - Γ ist endliches **Kelleralphabet**.
 - $K \subseteq Z \times \Sigma^* \times \Gamma^* \times \Gamma^* \times Z$ ist die endliche **Zustandsübergangsrelation**.



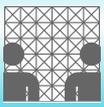
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.
 - Γ ist endliches **Kelleralphabet**.
 - $K \subseteq Z \times \Sigma^* \times \Gamma^* \times \Gamma^* \times Z$ ist die endliche **Zustandsübergangsrelation**.
 - $Z_{\text{start}} \subseteq Z$ ist die Menge der **Startzustände**.



Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.
 - Γ ist endliches **Kelleralphabet**.
 - $K \subseteq Z \times \Sigma^* \times \Gamma^* \times \Gamma^* \times Z$ ist die endliche **Zustandsübergangsrelation**.
 - $Z_{\text{start}} \subseteq Z$ ist die Menge der **Startzustände**.
 - $Z_{\text{end}} \subseteq Z$ ist die Menge der **Endzustände**.

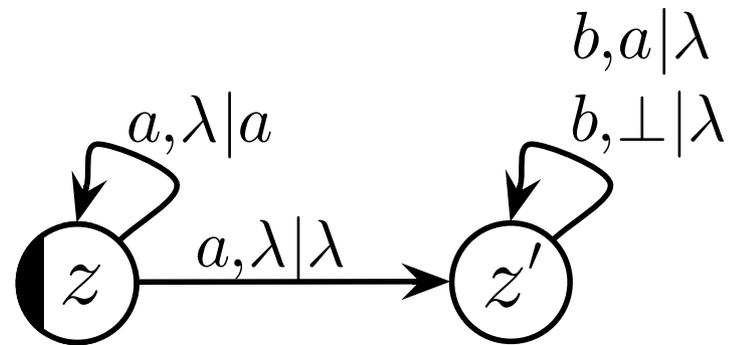


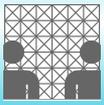
Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$, wobei gilt:
 - Z ist endliche Menge von **Zuständen**.
 - Σ ist endliches **Eingabealphabet**.
 - Γ ist endliches **Kelleralphabet**.
 - $K \subseteq Z \times \Sigma^* \times \Gamma^* \times \Gamma^* \times Z$ ist die endliche **Zustandsübergangsrelation**.
 - $Z_{\text{start}} \subseteq Z$ ist die Menge der **Startzustände**.
 - $Z_{\text{end}} \subseteq Z$ ist die Menge der **Endzustände**.
 - $\perp \in \Gamma$ ist das **Kellerbodenzeichen** oder **Kellerbodensymbol**.

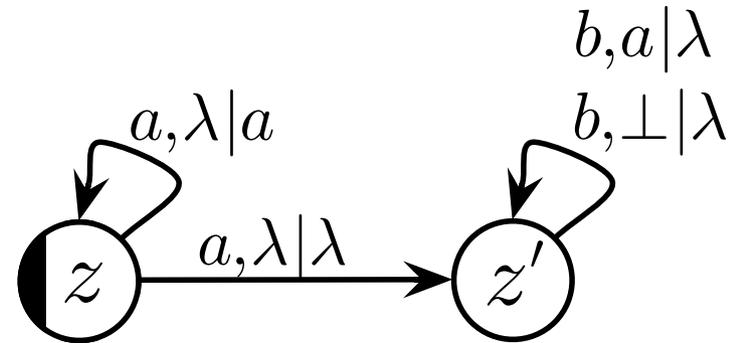


Beispiel

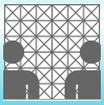




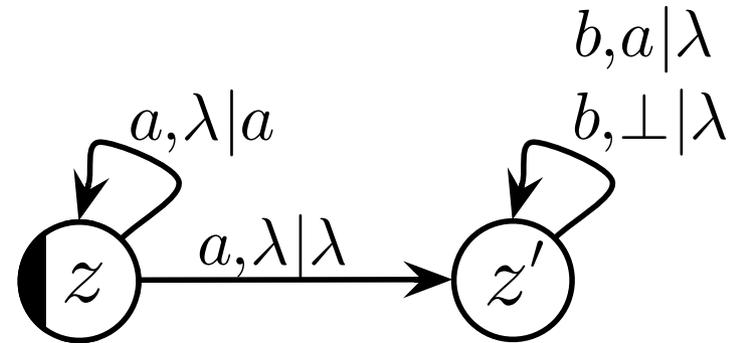
Beispiel



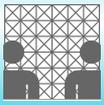
- Was ist die akzeptierte Sprache?



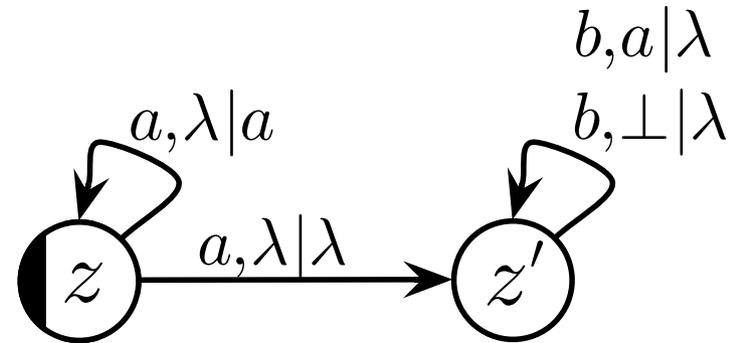
Beispiel



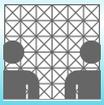
- Was ist die akzeptierte Sprache?
- Was ist eine Rechnung eines PDA?



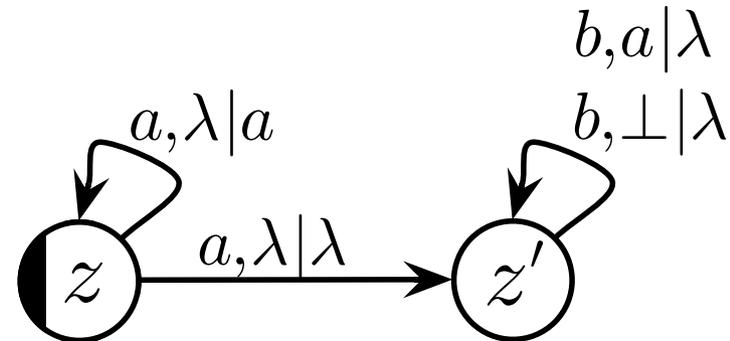
Beispiel



- Was ist die akzeptierte Sprache?
- Was ist eine Rechnung eines PDA?
- Was ist eine Konfiguration eines PDA?



Beispiel



- Was ist die akzeptierte Sprache?
- Was ist eine Rechnung eines PDA?
- Was ist eine Konfiguration eines PDA?

Das wird Gegenstand der Vorlesung in der nächsten Woche sein!