



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



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung



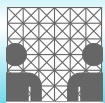
Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz
- Den **Beweis** führen wir indirekt:



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz
- Den **Beweis** führen wir indirekt:
 - $L_1 := \{a^n b^n \mid n \in \mathbb{N}\}$ wie auch $L_2 := \{b^m c^m \mid m \in \mathbb{N}\}$ sind kontextfreie Sprachen.



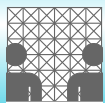
Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz
- Den **Beweis** führen wir indirekt:
 - $L_1 := \{a^n b^n \mid n \in \mathbb{N}\}$ wie auch $L_2 := \{b^m c^m \mid m \in \mathbb{N}\}$ sind kontextfreie Sprachen.
 - $L_3 := L_1 \cdot \{c\}^*$ und $L_4 := \{a\}^* \cdot L_2$ ist kontextfrei.



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz
- Den **Beweis** führen wir indirekt:
 - $L_1 := \{a^n b^n \mid n \in \mathbb{N}\}$ wie auch $L_2 := \{b^m c^m \mid m \in \mathbb{N}\}$ sind kontextfreie Sprachen.
 - $L_3 := L_1 \cdot \{c\}^*$ und $L_4 := \{a\}^* \cdot L_2$ ist kontextfrei.
 - $L_5 := L_3 \cap L_4 = \{a^n b^n c^n \mid n \in \mathbb{N}\} \in \mathcal{Cf}$. L_5 ist aber nicht kontextfrei!



Abschlusseigenschaften (2)

- Die Familie der kontextfreien Sprachen ist **nicht** abgeschlossen gegenüber folgenden Operatoren:
 1. Durchschnittsbildung
 2. Komplementbildung
 3. Bildung einer Mengendifferenz
- Den **Beweis** führen wir indirekt:
 - $L_1 := \{a^n b^n \mid n \in \mathbb{N}\}$ wie auch $L_2 := \{b^m c^m \mid m \in \mathbb{N}\}$ sind kontextfreie Sprachen.
 - $L_3 := L_1 \cdot \{c\}^*$ und $L_4 := \{a\}^* \cdot L_2$ ist kontextfrei.
 - $L_5 := L_3 \cap L_4 = \{a^n b^n c^n \mid n \in \mathbb{N}\} \in \mathcal{Cf}$. L_5 ist aber nicht kontextfrei!
- 2. und 3. folgen somit direkt!



Schnitt mit regulären Mengen

- Obwohl $\mathcal{C}f$ nicht allgemein gegen Durchschnittsbildung abgeschlossen ist, so doch für einen speziellen Fall:



Schnitt mit regulären Mengen

- Obwohl \mathcal{Cf} nicht allgemein gegen Durchschnittsbildung abgeschlossen ist, so doch für einen speziellen Fall:
- **Theorem:** Für $L \in \mathcal{Cf}$ und $R \in \mathcal{Reg}$ gilt $L \cap R \in \mathcal{Cf}$, kurz $\mathcal{Cf} \wedge \mathcal{Reg} \subseteq \mathcal{Cf}$.



Schnitt mit regulären Mengen

- Obwohl Cf nicht allgemein gegen Durchschnittsbildung abgeschlossen ist, so doch für einen speziellen Fall:
- **Theorem:** Für $L \in Cf$ und $R \in Reg$ gilt $L \cap R \in Cf$, kurz $Cf \wedge Reg \subseteq Cf$.
- **Beweisidee:** Bildung eines Produktautomaten aus dem NFA für R und der endlichen Steuerung des Kellerautomaten für L ... aber was ist eigentlich ein Kellerautomat?



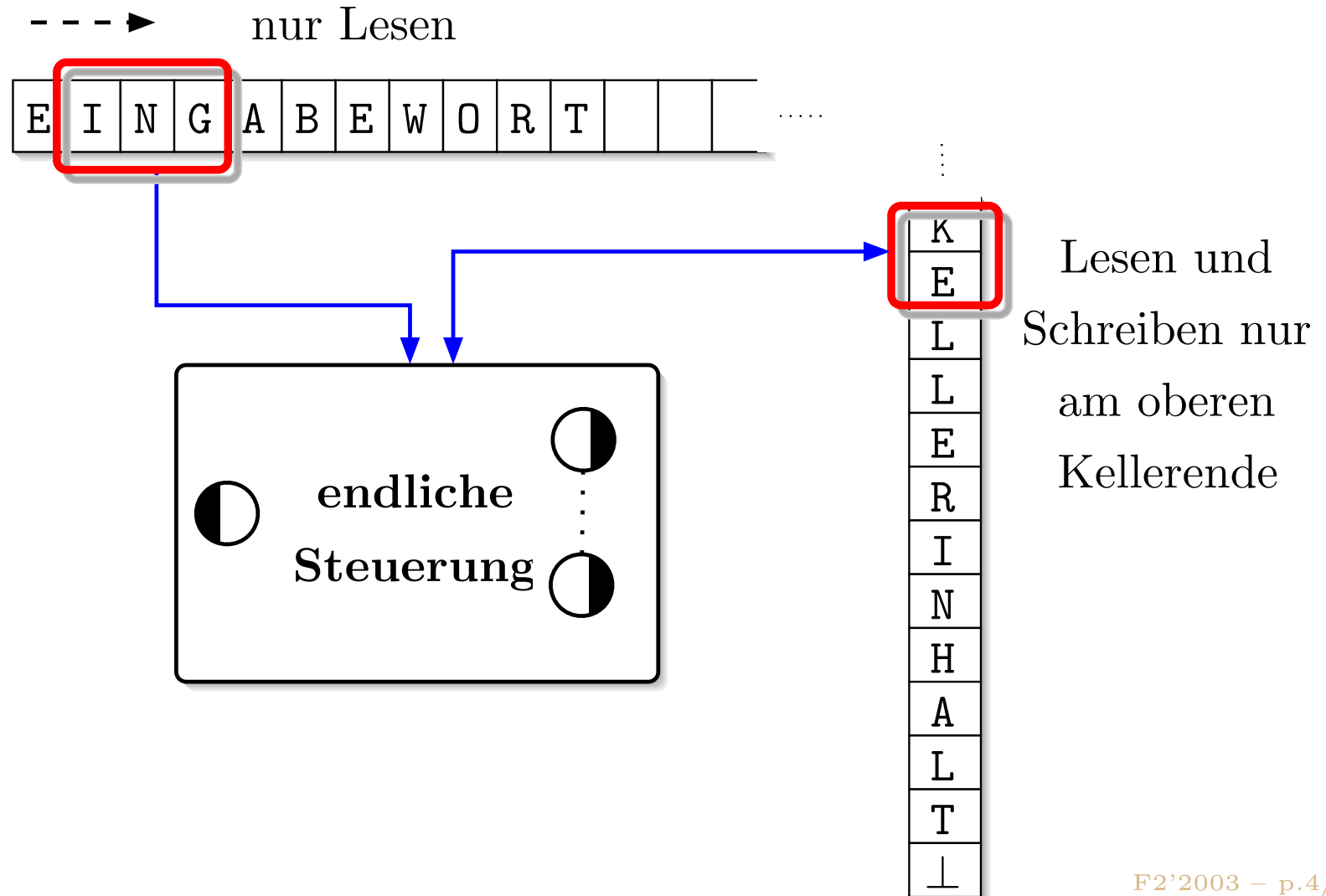
Schnitt mit regulären Mengen

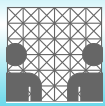
- Obwohl \mathcal{Cf} nicht allgemein gegen Durchschnittsbildung abgeschlossen ist, so doch für einen speziellen Fall:
- **Theorem:** Für $L \in \mathcal{Cf}$ und $R \in \mathcal{Reg}$ gilt $L \cap R \in \mathcal{Cf}$, kurz $\mathcal{Cf} \wedge \mathcal{Reg} \subseteq \mathcal{Cf}$.
- **Beweisidee:** Bildung eines Produktautomaten aus dem NFA für R und der endlichen Steuerung des Kellerautomaten für L ... aber was ist eigentlich ein Kellerautomat?
- Im Skript ist dies auch direkt mit Grammatiken bewiesen!



Kellerautomaten

- Eine Kellerautomat 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:



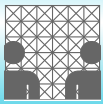
Kellerautomat (formal)

- 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**.



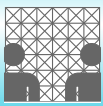
Kellerautomat (formal)

- 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**.



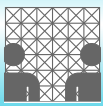
Kellerautomat (formal)

- 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 **Zustandsüberführungsrelation**.



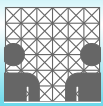
Kellerautomat (formal)

- 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 **Zustandsüberführungsrelation**.
 - $Z_{\text{start}} \subseteq Z$ 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 **Zustandsüberführungsrelation**.
 - $Z_{\text{start}} \subseteq Z$ 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**.



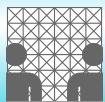
Einige Notationen

- **Wichtige Notation:** Der Kellerinhalt wird durch ein Wort $v \in \Gamma^*$ so beschrieben, dass das oberste Zeichen des Kellerinhaltes in v ganz am Anfang, d.h. links, steht.



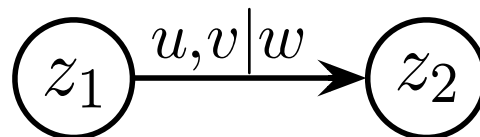
Einige Notationen

- **Wichtige Notation:** Der Kellerinhalt wird durch ein Wort $v \in \Gamma^*$ so beschrieben, dass das oberste Zeichen des Kellerinhaltes in v ganz am Anfang, d.h. links, steht.
- **Beispiel:** $abab\perp$ bedeutet, dass a das oberste und \perp das unterste Zeichen im Keller ist.



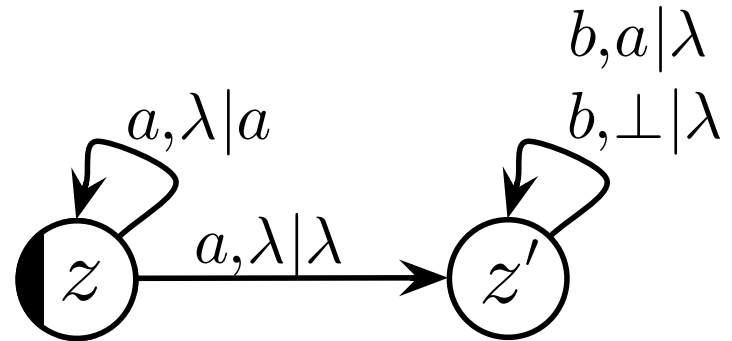
Einige Notationen

- **Wichtige Notation:** Der Kellerinhalt wird durch ein Wort $v \in \Gamma^*$ so beschrieben, dass das oberste Zeichen des Kellerinhaltes in v ganz am Anfang, d.h. links, steht.
- **Beispiel:** $abab\perp$ bedeutet, dass a das oberste und \perp das unterste Zeichen im Keller ist.
- Zunächst eine weitere **Notation:**
Zustandsüberführungen werden als beschriftete Kanten gezeichnet. Für $(z_1, u, v, w, z_2) \in K$:



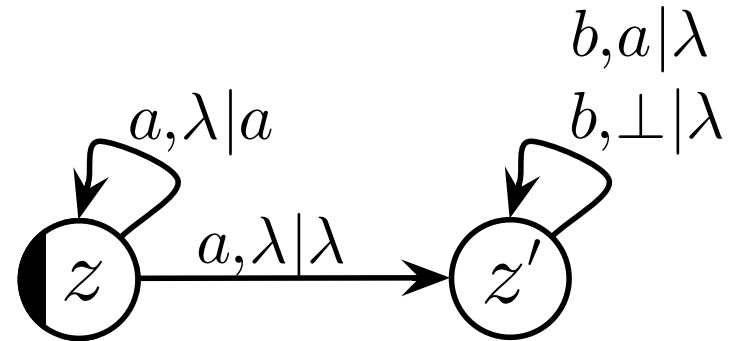


Beispiel





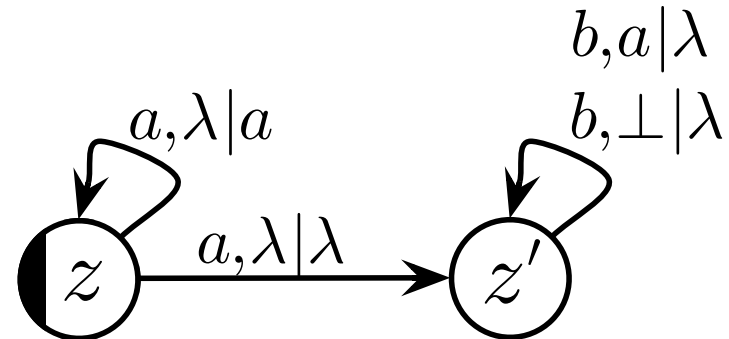
Beispiel



● Offensichtliche Frage:



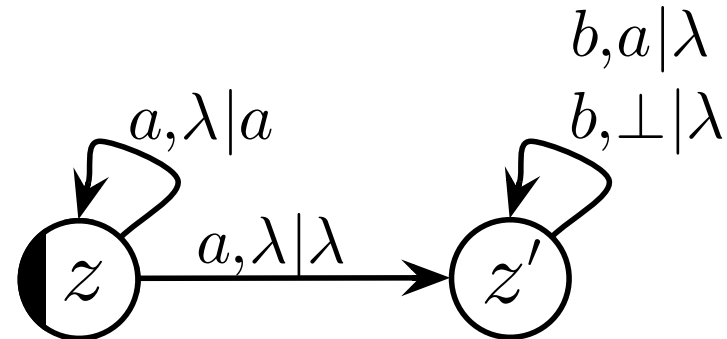
Beispiel



- **Offensichtliche Frage:**
 - Was ist die akzeptierte Sprache eines PDA?



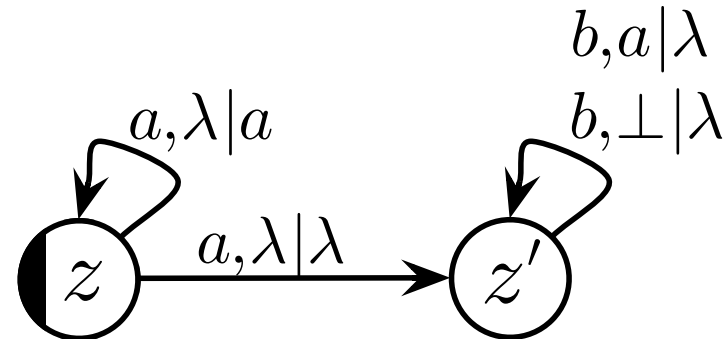
Beispiel



- **Offensichtliche Frage:**
 - Was ist die akzeptierte Sprache eines PDA?
 - ... dazu benötigen wir folgende Begriffe:



Beispiel

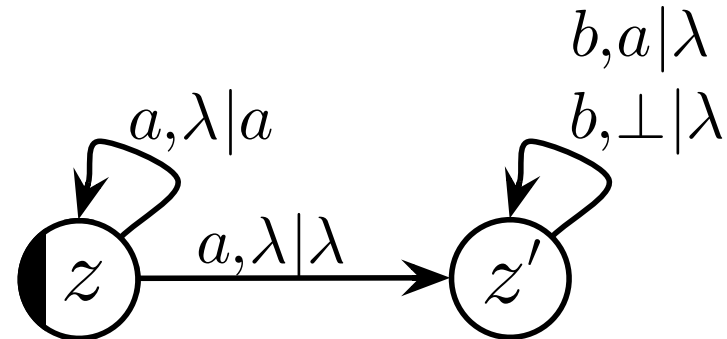


● Offensichtliche Frage:

- Was ist die akzeptierte Sprache eines PDA?
- ... dazu benötigen wir folgende Begriffe:
 - Rechnung/Erfolgsrechnung eines PDA



Beispiel

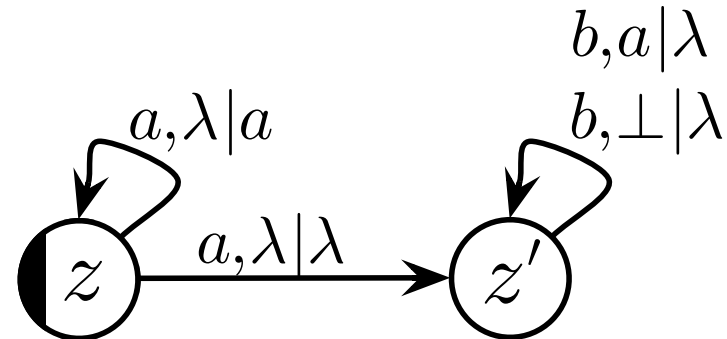


● Offensichtliche Frage:

- Was ist die akzeptierte Sprache eines PDA?
- ... dazu benötigen wir folgende Begriffe:
 - Rechnung/Erfolgsrechnung eines PDA
 - Konfiguration eines PDA



Beispiel



● Offensichtliche Frage:

- Was ist die akzeptierte Sprache eines PDA?
- ... dazu benötigen wir folgende Begriffe:
 - Rechnung/Erfolgsrechnung eines PDA
 - Konfiguration eines PDA
 - Überführungs- oder Transitionsrelation eines PDA



Konfiguration

- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.



Konfiguration

- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.
- Dabei zu beachten: Konfigurationen



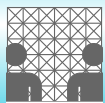
Konfiguration

- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.
- Dabei zu beachten: Konfigurationen
 - sind **relativ** zu einer festen Struktur.



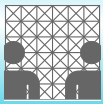
Konfiguration

- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.
- Dabei zu beachten: Konfigurationen
 - sind **relativ** zu einer festen Struktur.
 - beinhalten eine Beschreibung des **Speicherinhaltes** und des **Systemzustands**

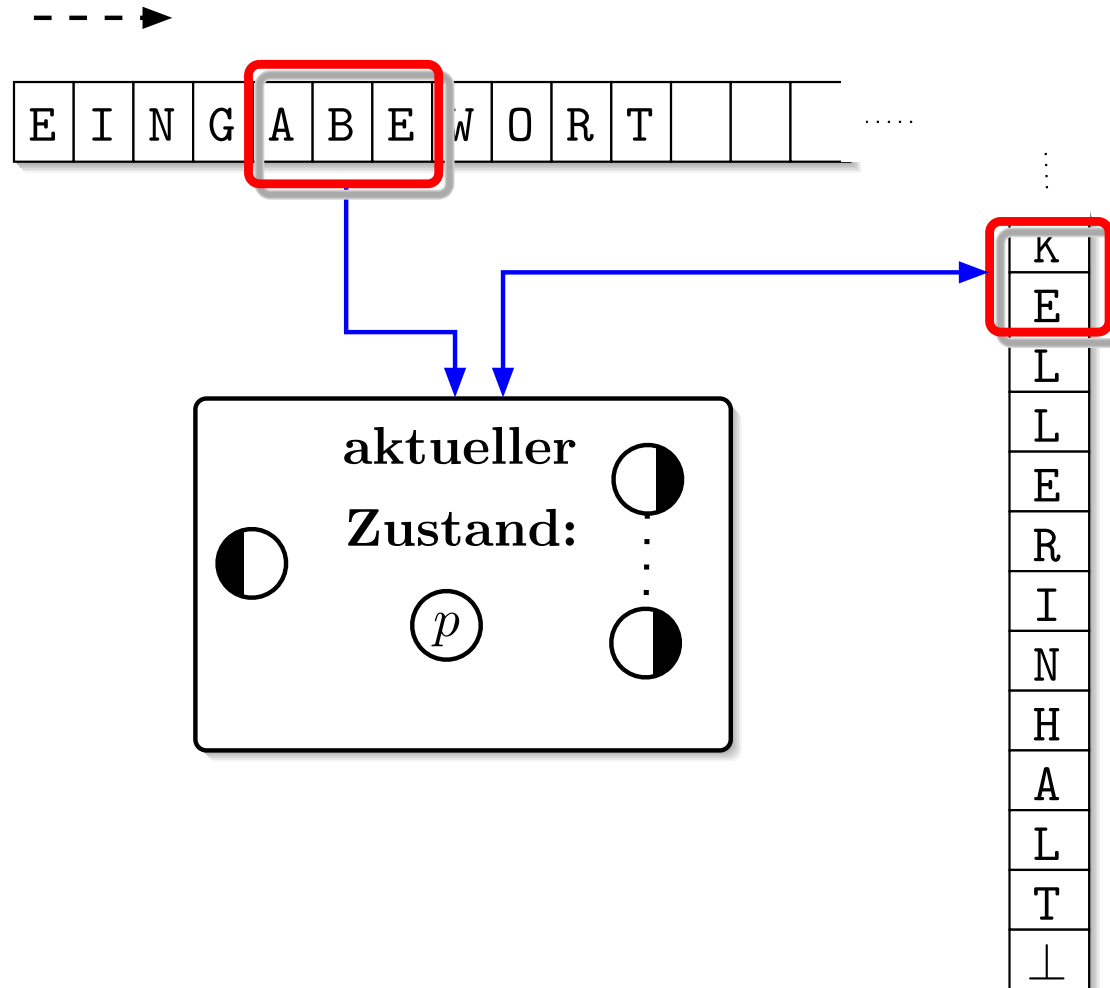


Konfiguration

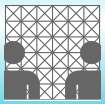
- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.
- Dabei zu beachten: Konfigurationen
 - sind **relativ** zu einer festen Struktur.
 - beinhalten eine Beschreibung des **Speicherinhaltes** und des **Systemzustands**
- **Definition:**
Eine **Konfiguration** (*instantaneous description*, *ID*) des PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ wird notiert als Element $k \in \Gamma^* \times Z \times \Sigma^*$.



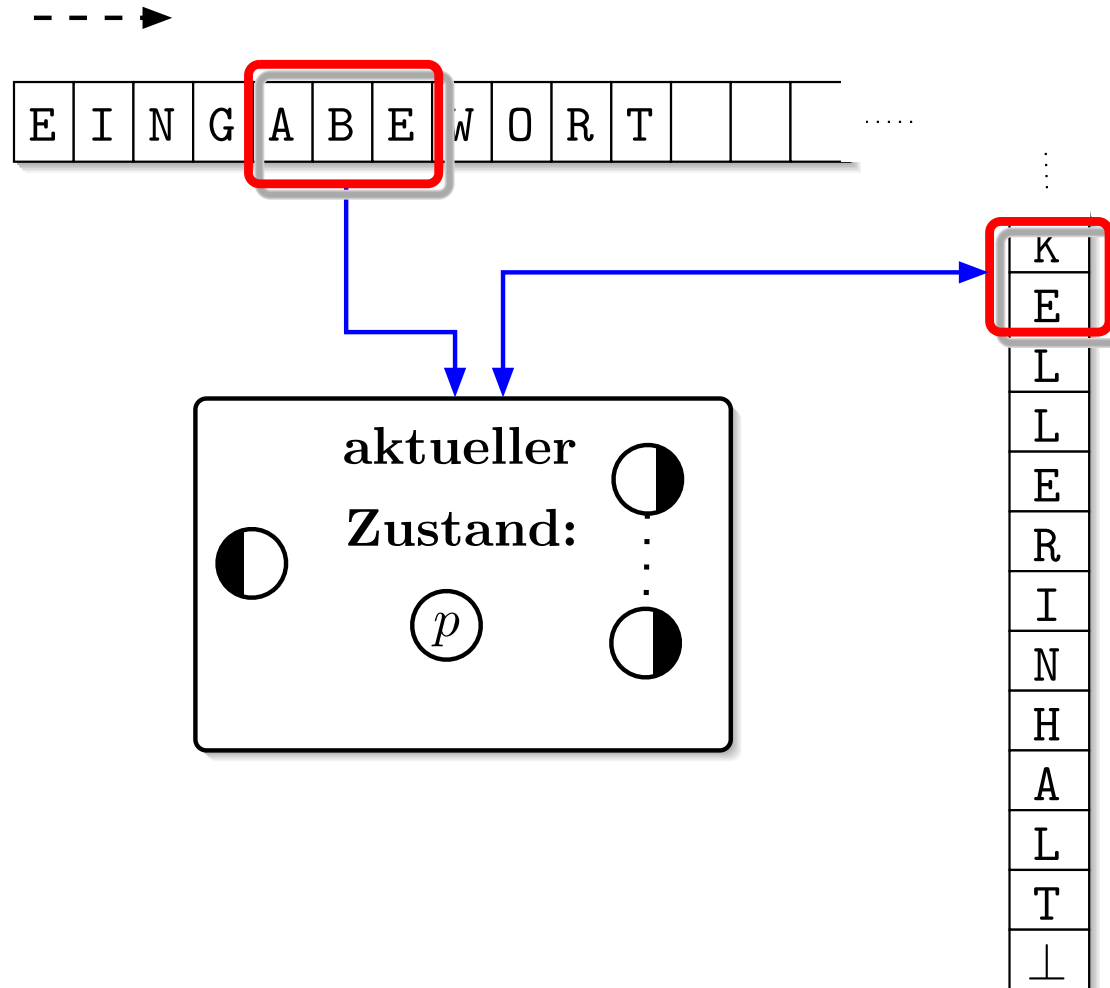
Beispiel: Konfigurationen



• Ein PDA ...



Beispiel: Konfigurationen



- Ein PDA ...
- ... und seine aktuelle Konfiguration:
 $\text{KELLERINHALT} \perp_p \text{ABEWORT}$

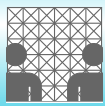


Überführungsrelation

- **Definition:** Seien $u, u', v \in \Gamma^*$, $w, w' \in \Sigma^*$ und $z, z' \in Z$.

Zwischen Konfigurationen eines Kellerautomaten A wird die **Überführungsrelation** (Rechenschritt-, Transitionsrelation) $\vdash \subseteq \Gamma^* \cdot Z \cdot \Sigma^* \times \Gamma^* \cdot Z \cdot \Sigma^*$ definiert durch:

$$uvzww' \vdash u'vz'w' \text{ gdw. } \exists(z, w, u, u', z') \in K$$



Überführungsrelation

- **Definition:** Seien $u, u', v \in \Gamma^*$, $w, w' \in \Sigma^*$ und $z, z' \in Z$.
Zwischen Konfigurationen eines Kellerautomaten A wird die **Überführungsrelation** (Rechenschritt-, Transitionsrelation) $\vdash \subseteq \Gamma^* \cdot Z \cdot \Sigma^* \times \Gamma^* \cdot Z \cdot \Sigma^*$ definiert durch:

$$uvzww' \vdash u'vz'w' \text{ gdw. } \exists(z, w, u, u', z') \in K$$

- Wie üblich bezeichnet \vdash^* die reflexive, transitive Hülle von \vdash und wir schreiben \vdash_A , wenn andernfalls unklar ist, zu welchem PDA diese Überführungsrelation gehört.



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.
- **Zur Erinnerung:** Bei **endlichen Automaten** musste ein *Wort bis zum Ende gelesen* werden und damit ein *Endzustand* erreicht werden. Z.B. für einen DFA:

$$\delta(z_0, w) = z \text{ mit } z \in Z_{\text{end}}$$



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.
- **Zur Erinnerung:** Bei **endlichen Automaten** musste ein *Wort bis zum Ende gelesen* werden und damit ein *Endzustand* erreicht werden. Z.B. für einen DFA:

$$\delta(z_0, w) = z \text{ mit } z \in Z_{\text{end}}$$

- Beim **Kellerautomaten** haben wir zwei Möglichkeiten:



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.
- **Zur Erinnerung:** Bei **endlichen Automaten** musste ein *Wort bis zum Ende gelesen* werden und damit ein *Endzustand* erreicht werden. Z.B. für einen DFA:

$$\delta(z_0, w) = z \text{ mit } z \in Z_{\text{end}}$$

- Beim **Kellerautomaten** haben wir zwei Möglichkeiten:
 - Zustand der Steuerungskomponente



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.
- **Zur Erinnerung:** Bei **endlichen Automaten** musste ein *Wort bis zum Ende gelesen* werden und damit ein *Endzustand* erreicht werden. Z.B. für einen DFA:

$$\delta(z_0, w) = z \text{ mit } z \in Z_{\text{end}}$$

- Beim **Kellerautomaten** haben wir zwei Möglichkeiten:
 - Zustand der Steuerungskomponente
 - Speicherzustand



Akzeptierungsbedingungen

- Verwendet werden i.A. Bedingungen an die Konfiguration eines Automaten.
- **Zur Erinnerung:** Bei **endlichen Automaten** musste ein *Wort bis zum Ende gelesen* werden und damit ein *Endzustand* erreicht werden. Z.B. für einen DFA:

$$\delta(z_0, w) = z \text{ mit } z \in Z_{\text{end}}$$

- Beim **Kellerautomaten** haben wir zwei Möglichkeiten:
 - Zustand der Steuerungskomponente
 - Speicherzustand
- Dies führt zur **Akzeptierung mit Endzustand** und zur **Akzeptierung mit leerem Keller**.



$L(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit **Endzustand** **akzeptierte Sprache** $L(A)$ ist

$$L(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z_e \in Z_{\text{end}} : \\ \exists v \in \Gamma^* : \perp z_0 w \xrightarrow{*} v z_e\}$$

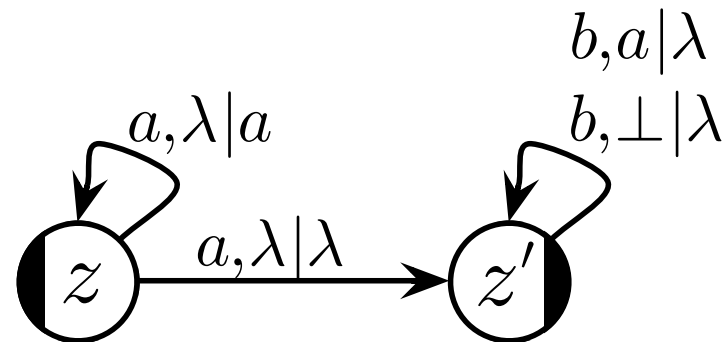


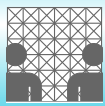
$L(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit **Endzustand** akzeptierte **Sprache** $L(A)$ ist

$$L(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z_e \in Z_{\text{end}} : \\ \exists v \in \Gamma^* : \perp z_0 w \vdash^* v z_e\}$$

- **Beispiel:**

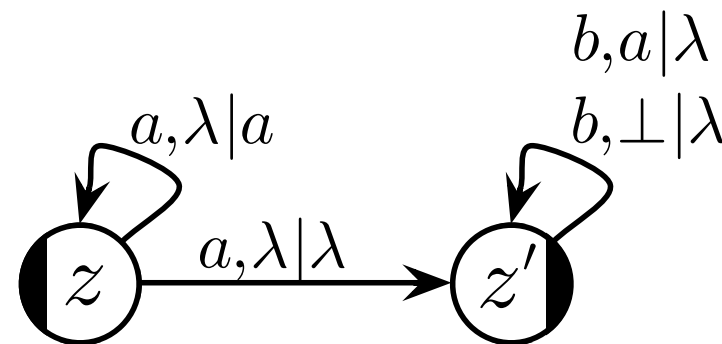




$L(A)$ für Kellerautomaten

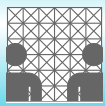
- **Definition:** Die vom nichtdeterministischen PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit **Endzustand** akzeptierte **Sprache** $L(A)$ ist

$$L(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z_e \in Z_{\text{end}} : \\ \exists v \in \Gamma^* : \perp z_0 w \xrightarrow{*} v z_e\}$$



- **Beispiel:**

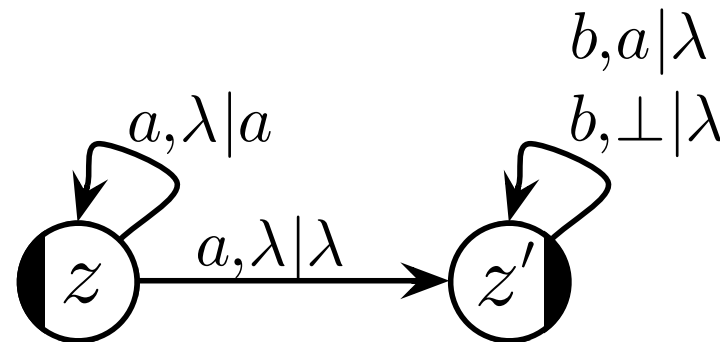
$$L(A) = \{a^n b^m \mid n, m \in \mathbb{N} \wedge n \geq 1 \wedge n \geq m\}$$



$L(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit **Endzustand** **akzeptierte Sprache** $L(A)$ ist

$$L(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z_e \in Z_{\text{end}} : \\ \exists v \in \Gamma^* : \perp z_0 w \vdash^* v z_e\}$$



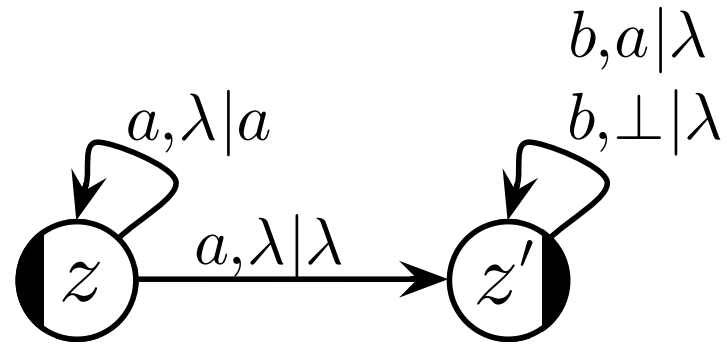
- **Beispiel:**

$$L(A) = \{a^n b^m \mid n, m \in \mathbb{N} \wedge n \geq 1 \wedge n \geq m\}$$

... für jedes gelesene b muss vorher ein a gelesen worden sein!



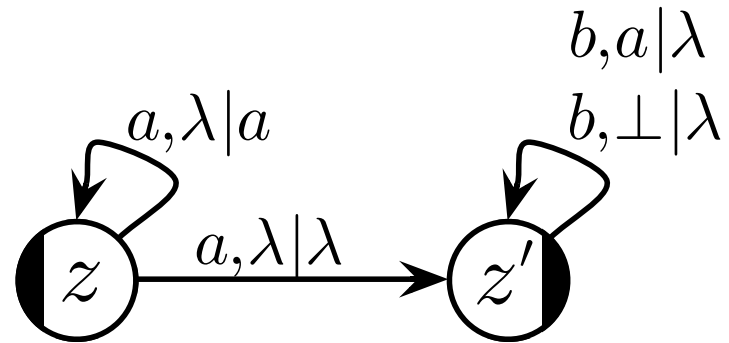
Beispielrechnungen



• $\perp z a a a b b$



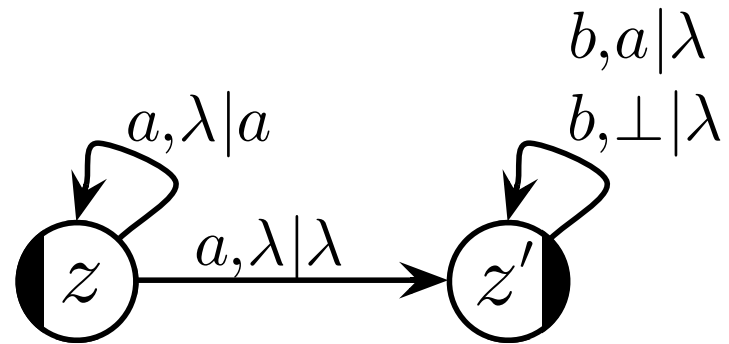
Beispielrechnungen



- $\perp z a a a b b$ ist Startkonfiguration.



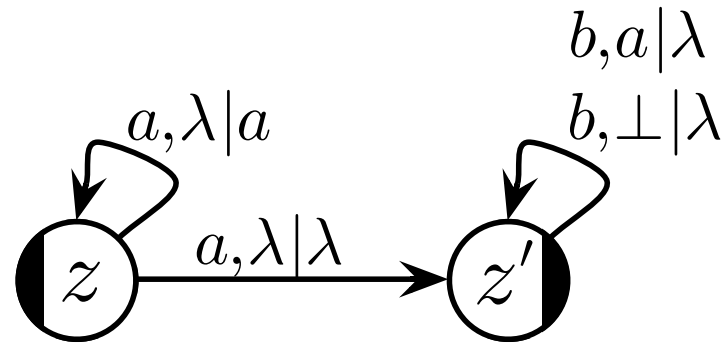
Beispielrechnungen



• $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a a b b$



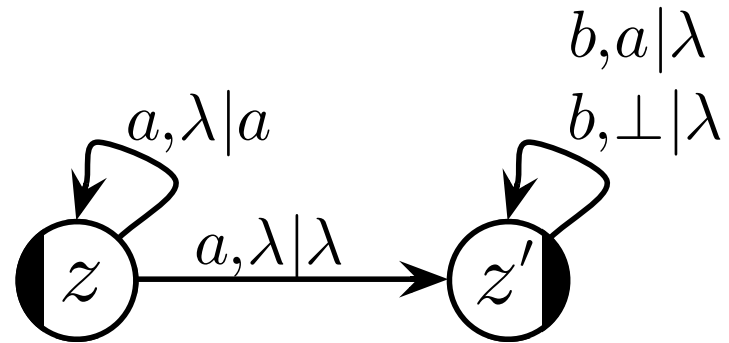
Beispielrechnungen



• $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a a b b \vdash a a \perp \mathbf{z} a b b$



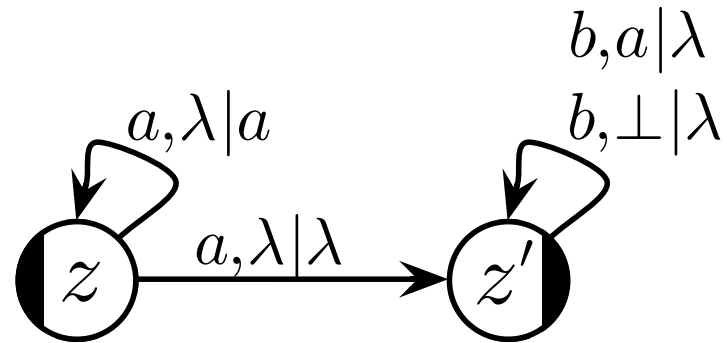
Beispielrechnungen



● $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a a b b \vdash a a \perp \mathbf{z} a b b$
 $\vdash a a a \perp \mathbf{z} b b$



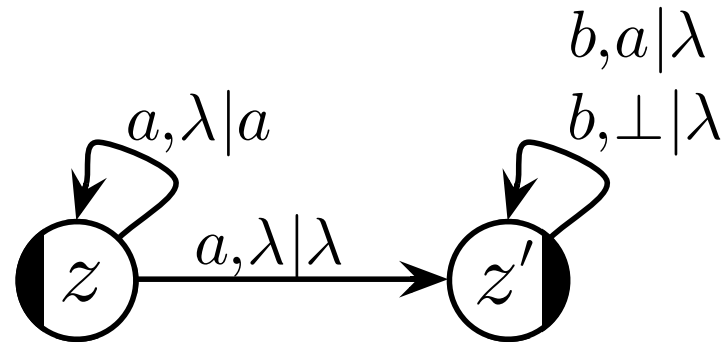
Beispielrechnungen



- $\perp \mathbf{z} a a a b b \vdash a \perp \mathbf{z} a a b b \vdash a a \perp \mathbf{z} a b b$
 $\vdash a a a \perp \mathbf{z} b b$ und hier blockiert der PDA.



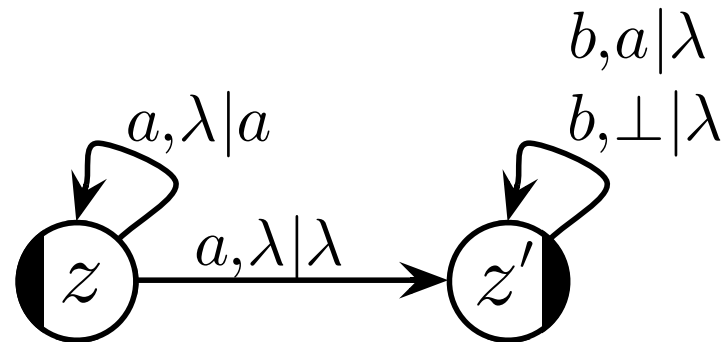
Beispielrechnungen



- $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a a b b \vdash a a \perp \mathbf{z} a b b$
 $\vdash a a a \perp \mathbf{z} b b$ und hier blockiert der PDA.
- Eine weitere Rechnung für $a a a b b$:
 $\perp \mathbf{z} a a a b b \vdash a \perp \mathbf{z} a a a b b \vdash a a \perp \mathbf{z} a a b b$
 $\vdash a a \perp \mathbf{z}' b b \vdash a \perp \mathbf{z}' b \vdash \perp \mathbf{z}'$



Beispielrechnungen



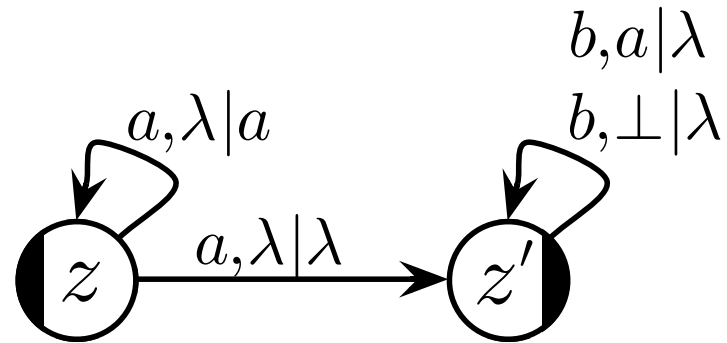
• $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a b b \vdash a a \perp \mathbf{z} a b b$
 $\vdash a a a \perp \mathbf{z} b b$ und hier blockiert der PDA.

• Eine weitere Rechnung für $a a b b$:
 $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a b b \vdash a a \perp \mathbf{z} a b b \vdash$
 $a a \perp \mathbf{z}' b b \vdash a \perp \mathbf{z}' b \vdash \perp \mathbf{z}'$

Zwar ist der Keller *nicht* leer (er enthält noch \perp),
aber es ist ein Endzustand erreicht \Rightarrow akzeptieren!



Beispielrechnungen



- $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a b b \vdash a a \perp \mathbf{z} a b b$
 $\vdash a a a \perp \mathbf{z} b b$ und hier blockiert der PDA.

- Eine weitere Rechnung für $a a a b b$:
 $\perp \mathbf{z} a a b b \vdash a \perp \mathbf{z} a a b b \vdash a a \perp \mathbf{z} a b b \vdash$
 $a a \perp \mathbf{z}' b b \vdash a \perp \mathbf{z}' b \vdash \perp \mathbf{z}'$

Zwar ist der Keller *nicht* leer (er enthält noch \perp),
aber es ist ein Endzustand erreicht \Rightarrow akzeptieren!

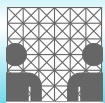
- Für $a a b b b$ existiert *keine* Erfolgsrechnung.



$N(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA A mit leerem Keller akzeptierte Sprache $N(A)$ ist

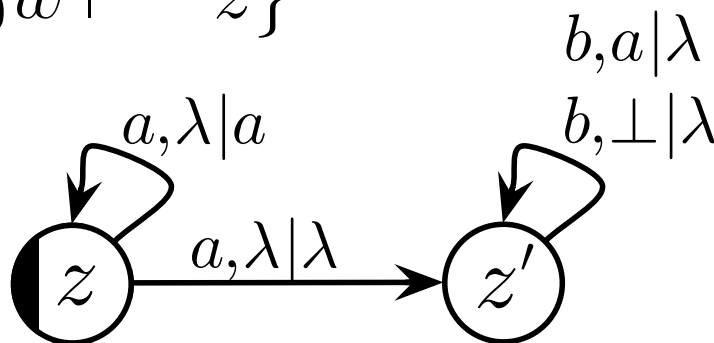
$$N(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z \in Z : \perp z_0 w \xrightarrow{*} z\}$$



$N(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA A mit leerem Keller akzeptierte Sprache $N(A)$ ist

$$N(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z \in Z : \perp z_0 w \xrightarrow{*} z\}$$



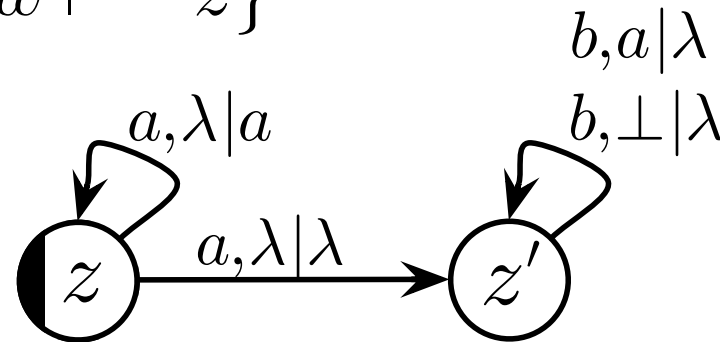
- **Beispiel:**



$N(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA A mit leerem Keller akzeptierte Sprache $N(A)$ ist

$$N(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z \in Z : \perp z_0 w \vdash^* z\}$$



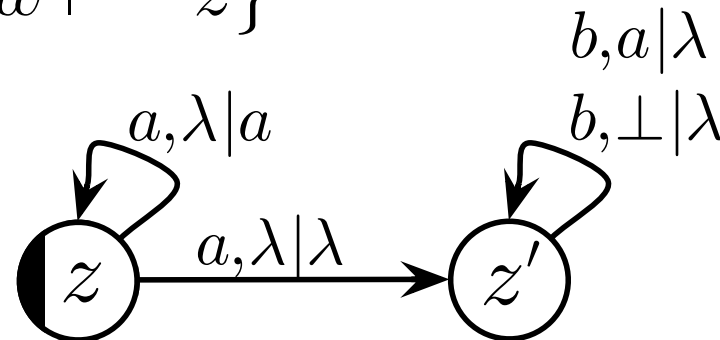
- **Beispiel:**
- $N(A) = \{a^n b^n \mid n \in \mathbb{N}, n \geq 1\}$



$N(A)$ für Kellerautomaten

- **Definition:** Die vom nichtdeterministischen PDA A mit leerem Keller akzeptierte Sprache $N(A)$ ist

$$N(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\text{start}} : \exists z \in Z : \perp z_0 w \vdash^* z\}$$



- **Beispiel:**
- $N(A) = \{a^n b^n \mid n \in \mathbb{N}, n \geq 1\}$
- ... für jedes gelesene b muss vorher ein a gelesen worden sein und für jedes a muss später ein b folgen!



Äquivalenzbegriff

- **Definition:** Zwei Kellerautomaten A und B heißen **äquivalent** genau dann, wenn ihre mit Endzustand akzeptierten Sprachen gleich sind, d.h. $L(A) = L(B)$ gilt.



Äquivalenzbegriff

- **Definition:** Zwei Kellerautomaten A und B heißen **äquivalent** genau dann, wenn ihre mit Endzustand akzeptierten Sprachen gleich sind, d.h. $L(A) = L(B)$ gilt.
- **Merke:** Dieser Äquivalenzbegriff vergleicht nur PDAs, die im Endzustand akzeptieren!



Äquivalenzbegriff

- **Definition:** Zwei Kellerautomaten A und B heißen **äquivalent** genau dann, wenn ihre mit Endzustand akzeptierten Sprachen gleich sind, d.h. $L(A) = L(B)$ gilt.
- **Merke:** Dieser Äquivalenzbegriff vergleicht nur PDAs, die im Endzustand akzeptieren!
- Eine Verallgemeinerung wäre aber natürlich möglich:



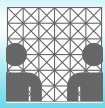
Äquivalenzbegriff

- **Definition:** Zwei Kellerautomaten A und B heißen **äquivalent** genau dann, wenn ihre mit Endzustand akzeptierten Sprachen gleich sind, d.h. $L(A) = L(B)$ gilt.
- **Merke:** Dieser Äquivalenzbegriff vergleicht nur PDAs, die im Endzustand akzeptieren!
- Eine Verallgemeinerung wäre aber natürlich möglich:
 - bezüglich der mit leerem Keller akzeptierten Sprachen ($N(A) = N(B)$)



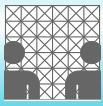
Äquivalenzbegriff

- **Definition:** Zwei Kellerautomaten A und B heißen **äquivalent** genau dann, wenn ihre mit Endzustand akzeptierten Sprachen gleich sind, d.h. $L(A) = L(B)$ gilt.
- **Merke:** Dieser Äquivalenzbegriff vergleicht nur PDAs, die im Endzustand akzeptieren!
- Eine Verallgemeinerung wäre aber natürlich möglich:
 - bezüglich der mit leerem Keller akzeptierten Sprachen ($N(A) = N(B)$)
 - „interdisziplinär“ ($L(A) = N(B)$)



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z$$
 gilt.



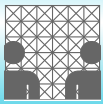
Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$
 - **deterministisch**, (Abk.: DPDA) genau dann, wenn die folgenden Bedingungen erfüllt sind:



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$
 - **deterministisch**, (Abk.: DPDA) genau dann, wenn die folgenden Bedingungen erfüllt sind:
 1. $|Z_{\text{start}}| = 1$



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$
 - **deterministisch**, (Abk.: DPDA) genau dann, wenn die folgenden Bedingungen erfüllt sind:
 1. $|Z_{\text{start}}| = 1$
 2. A ist buchstabierend, d.h.
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$$



Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$
 - **deterministisch**, (Abk.: DPDA) genau dann, wenn die folgenden Bedingungen erfüllt sind:
 1. $|Z_{\text{start}}| = 1$
 2. A ist buchstabierend, d.h.
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$$
 3. Zu jedem $(z, x, X) \in Z \times (\Sigma \cup \{\lambda\}) \times \Gamma$ gibt es max. ein $(w, z') \in \Gamma^* \times Z$ mit $(z, x, X, w, z') \in K$.



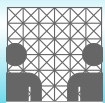
Eigenschaften von PDAs

- Ein Kellerautomat $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ heißt
 - **fast-buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z \text{ gilt.}$$
 - **buchstabierend** genau dann, wenn
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z \text{ gilt.}$$
 - **deterministisch**, (Abk.: DPDA) genau dann, wenn die folgenden Bedingungen erfüllt sind:
 1. $|Z_{\text{start}}| = 1$
 2. A ist buchstabierend, d.h.
$$K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$$
 3. Zu jedem $(z, x, X) \in Z \times (\Sigma \cup \{\lambda\}) \times \Gamma$ gibt es max. ein $(w, z') \in \Gamma^* \times Z$ mit $(z, x, X, w, z') \in K$.
 4. Falls $(z, \lambda, X, w, z') \in K$ ist, so gilt $\forall x \in \Sigma :$
$$\forall z'' \in Z : (z, x, X, w', z'') \notin K.$$



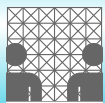
Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$.



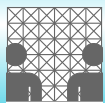
Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$.
- **Beweisidee:** Konstruktion eines äquivalenten PDA, der je Schritt max. ein Eingabesymbol liest.



Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$.
- **Beweisidee:** Konstruktion eines äquivalenten PDA, der je Schritt max. ein Eingabesymbol liest.
 - Speichern der jeweils gelesenen Symbolkette in einem neuen Zustand:
$$Z'' := Z \cup \{[z, u'] \mid (z, u, v, w, z') \in K \wedge u' \in \Sigma^* \text{ ist echter Präfix von } u \text{ mit } |u'| \geq 1\}$$



Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$.
- **Beweisidee:** Konstruktion eines äquivalenten PDA, der je Schritt max. ein Eingabesymbol liest.
 - Speichern der jeweils gelesenen Symbolkette in einem neuen Zustand:
$$Z'' := Z \cup \{[z, u'] \mid (z, u, v, w, z') \in K \wedge u' \in \Sigma^* \text{ ist echter Präfix von } u \text{ mit } |u'| \geq 1\}$$
 - Für Kante $k = (z, u_1 u_2 \dots u_n, v, w, z') \in K$ neu:
$$(z, u_1, v, \lambda, [z', u_1]), ([z', u_1], u_2, \lambda, \lambda, [z', u_1 u_2]), \dots, ([z', u_1 \dots u_{n-1}], u_n, \lambda, w, z')$$



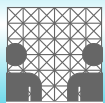
CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.



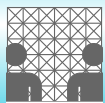
CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.



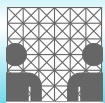
CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.
- Konstruiere $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit drei Zuständen: $Z := \{z_0, z_1, z_2\}$



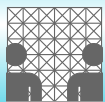
CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.
- Konstruiere $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit drei Zuständen: $Z := \{z_0, z_1, z_2\}$
- $\Sigma := V_T$



CFG vs. PDA

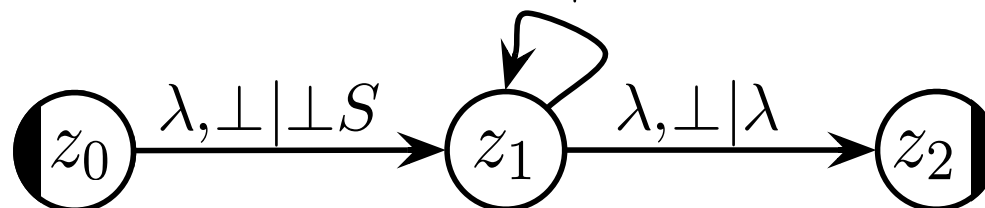
- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.
- Konstruiere $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit drei Zuständen: $Z := \{z_0, z_1, z_2\}$
- $\Sigma := V_T$
- $\Gamma := V_N \cup V_T \cup \{\perp\}$



CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.
- Konstruiere $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ mit drei Zuständen: $Z := \{z_0, z_1, z_2\}$
- $\Sigma := V_T$
- $\Gamma := V_N \cup V_T \cup \{\perp\}$
- und Kanten:

$\lambda, A \mid w$ für alle $A \longrightarrow w \in P$
 $a, a \mid \lambda$ für jedes $a \in V_T$





$L(A)$ vs. $N(A)$

● **Lemma:** Es existiert zu jedem PDA

$A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA

$B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA
 $A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA
 $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere
PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA
 $A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA
 $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere
PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit
 - neuem Kellersymbol $\$,$ also $\Gamma' := \Gamma \uplus \{\$\}$;



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA
 $A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA
 $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit
 - neuem Kellersymbol $\$,$ also $\Gamma' := \Gamma \uplus \{\$\}$;
 - neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA

$A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA

$B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.

- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere

PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit

- neuem Kellersymbol $\$$, also $\Gamma' := \Gamma \uplus \{\$\}$;
- neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
- $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp \$, z) \mid z \in Z_{\text{start}}\}$ „Initialisieren“



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA

$A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA

$B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.

- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere

PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit

- neuem Kellersymbol $\$$, also $\Gamma' := \Gamma \uplus \{\$\}$;
- neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
- $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp \$, z) \mid z \in Z_{\text{start}}\}$ „Initialisieren“
- $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\text{empty}}) \mid z_e \in Z_{\text{end}}\}$ „Übergänge zum den Keller leerenden Zustand z_{empty} “



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA

$A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA

$B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.

- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere

PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit

- neuem Kellersymbol $\$$, also $\Gamma' := \Gamma \uplus \{\$\}$;
- neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
- $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp \$, z) \mid z \in Z_{\text{start}}\}$ „Initialisieren“
- $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\text{empty}}) \mid z_e \in Z_{\text{end}}\}$ „Übergänge zum den Keller leerenden Zustand z_{empty} “
- $K_3 := \{(z_{\text{empty}}, \lambda, a, \lambda, z_{\text{empty}}) \mid a \in \Gamma'\}$ „Leeren“



$L(A)$ vs. $N(A)$

- **Lemma:** Es existiert zu jedem PDA

$A := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ ein PDA

$B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit $L(A) = N(B)$.

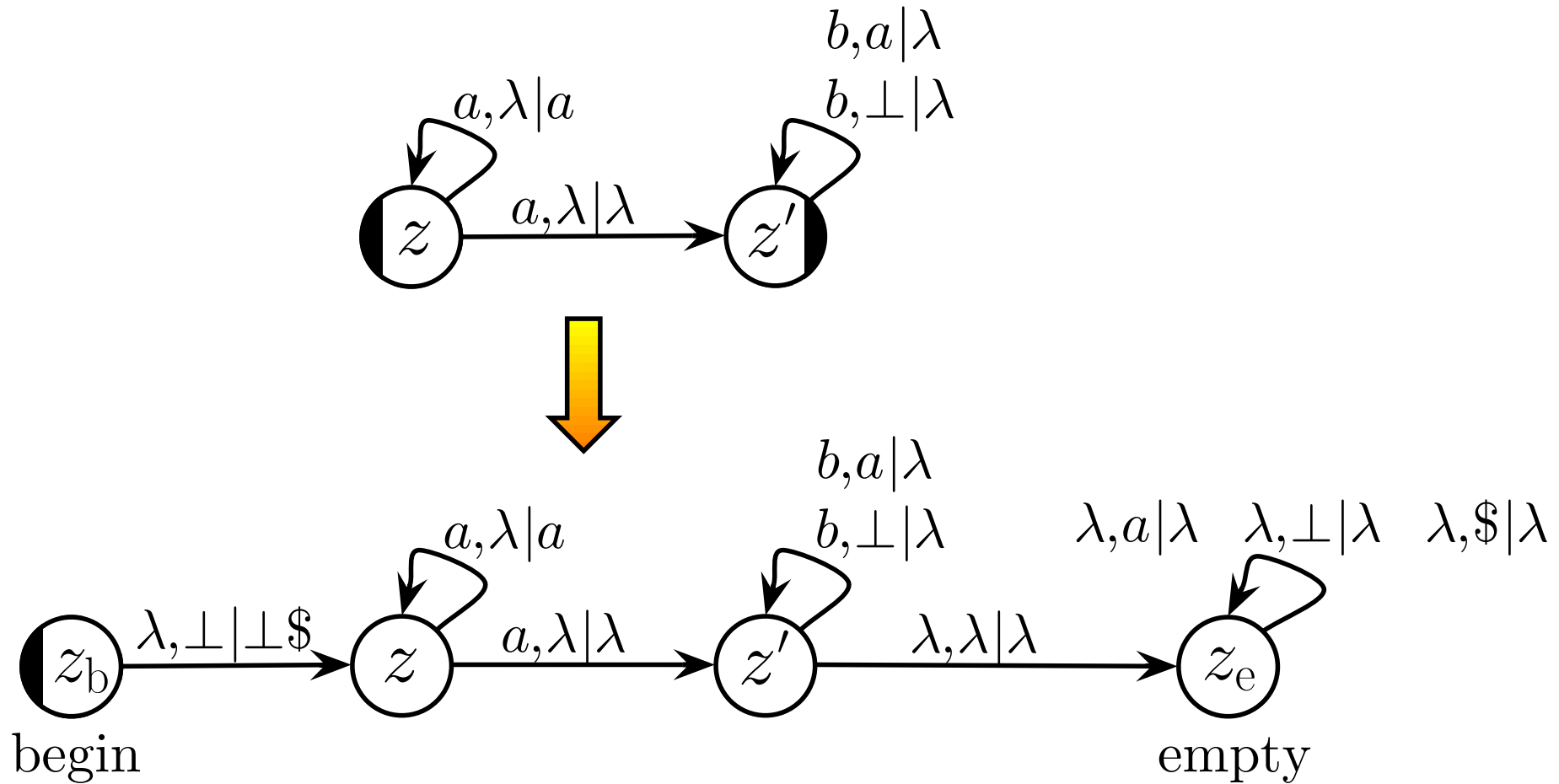
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere

PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ mit

- neuem Kellersymbol $\$$, also $\Gamma' := \Gamma \uplus \{\$\}$;
- neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
- $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp \$, z) \mid z \in Z_{\text{start}}\}$ „Initialisieren“
- $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\text{empty}}) \mid z_e \in Z_{\text{end}}\}$ „Übergänge zum den Keller leerenden Zustand z_{empty} “
- $K_3 := \{(z_{\text{empty}}, \lambda, a, \lambda, z_{\text{empty}}) \mid a \in \Gamma'\}$ „Leeren“
- $K' := K \cup K_1 \cup K_2 \cup K_3$



Beispiel: $L(A) = N(B)$

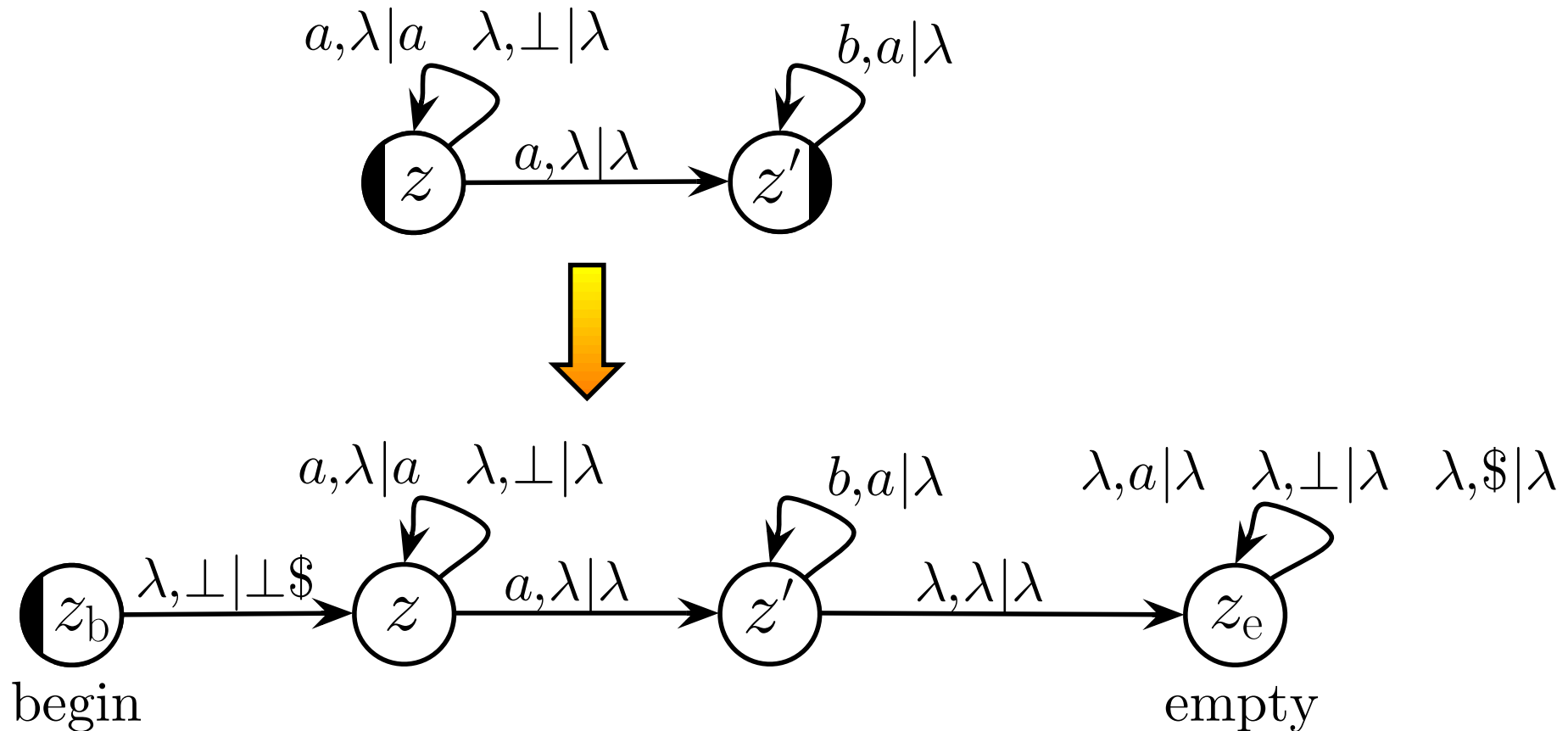


... aber warum wird das $\$$ neu eingeführt?

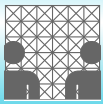


Beispiel: $L(A) = N(B)$

Ein etwas modifizierter Automat:



... ohne das neue Bodensymbol würde auch das leere Wort λ akzeptiert!



Gilt immer $L(A) \in Cf$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.



Gilt immer $L(A) \in Cf$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.



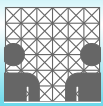
Gilt immer $L(A) \in Cf$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.
 - Definiere PDA $C := (Z'', \Sigma, \Gamma'', K'', Z''_{\text{start}}, Z''_{\text{end}}, \perp)$ mit



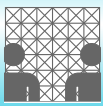
Gilt immer $L(A) \in \mathcal{Cf}$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.
 - Definiere PDA $C := (Z'', \Sigma, \Gamma'', K'', Z''_{\text{start}}, Z''_{\text{end}}, \perp)$ mit
 - $K_4 := \{(z, x, X, vX, z') \mid X \in \Gamma'\} \subseteq K''$ falls $(z, x, \lambda, v, z') \in K'$ mit $x \in \Sigma \cup \{\lambda\}$



Gilt immer $L(A) \in \mathcal{Cf}$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.
 - Definiere PDA $C := (Z'', \Sigma, \Gamma'', K'', Z''_{\text{start}}, Z''_{\text{end}}, \perp)$ mit
 - $K_4 := \{(z, x, X, vX, z') \mid X \in \Gamma'\} \subseteq K''$ falls $(z, x, \lambda, v, z') \in K'$ mit $x \in \Sigma \cup \{\lambda\}$
 - Insgesamt:
$$K'' := K' \setminus \{(z, x, \lambda, v, z') \in K' \mid x \in \Sigma \cup \{\lambda\}\} \cup K_4$$



Gilt immer $L(A) \in Cf$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.
 - Definiere PDA $C := (Z'', \Sigma, \Gamma'', K'', Z''_{\text{start}}, Z''_{\text{end}}, \perp)$ mit
 - $K_4 := \{(z, x, X, vX, z') \mid X \in \Gamma'\} \subseteq K''$ falls $(z, x, \lambda, v, z') \in K'$ mit $x \in \Sigma \cup \{\lambda\}$
 - Insgesamt:
$$K'' := K' \setminus \{(z, x, \lambda, v, z') \in K' \mid x \in \Sigma \cup \{\lambda\}\} \cup K_4$$
 - Es gilt $N(C) = L(A)$.



Gilt immer $L(A) \in Cf$?

- **Theorem:** Für jeden PDA A ist $L(A)$ eine kontextfreie Sprache.
- **Beweisskizze:** Sei $B := (Z', \Sigma, \Gamma', K', Z'_{\text{start}}, Z'_{\text{end}}, \perp)$ der A existierende fast-buchstabierende PDA mit $L(A) = N(B)$.
 - Definiere PDA $C := (Z'', \Sigma, \Gamma'', K'', Z''_{\text{start}}, Z''_{\text{end}}, \perp)$ mit
 - $K_4 := \{(z, x, X, vX, z') \mid X \in \Gamma'\} \subseteq K''$ falls $(z, x, \lambda, v, z') \in K'$ mit $x \in \Sigma \cup \{\lambda\}$
 - Insgesamt:
$$K'' := K' \setminus \{(z, x, \lambda, v, z') \in K' \mid x \in \Sigma \cup \{\lambda\}\} \cup K_4$$
 - Es gilt $N(C) = L(A)$.
 - Nun wird mittels **Tripelkonstruktion** eine CFG konstruiert, die mögliche Rechnungen des PDA C simuliert.



Tripelkonstruktion

- **Idee:** Verwendung des Nonterminalalphabets zur Codierung von Zustandsübergängen des PDA.



Tripelkonstruktion

- **Idee:** Verwendung des Nonterminalalphabets zur Codierung von Zustandsübergängen des PDA.
- Die Regeln „**raten**“ eine Zustandsfolge des Kellerautomaten.



Tripelkonstruktion

- **Idee:** Verwendung des Nonterminalalphabets zur Codierung von Zustandsübergängen des PDA.
- Die Regeln „**raten**“ eine Zustandsfolge des Kellerautomaten.
- Die Grammatik $G_C := (V_N, V_T, P, S)$ besteht aus:

$$V_N := \{S\} \cup \{[z, X, z'] \mid X \in \Gamma'', z, z' \in Z''\}$$

$$V_T := \Sigma$$

$$\begin{aligned} P := & \{S \longrightarrow [z, \$, z'] \mid z \in Z''_{\text{start}}, z' \in Z''\} \cup \\ & \{[z, X, z'] \longrightarrow a \mid (z, a, X, \lambda, z') \in K'', a \in \Sigma \cup \{\lambda\}, \\ & X \in \Gamma'', z, z' \in Z''\} \cup \\ & \{[z, X, z_k] \longrightarrow a[z', B_1, z_1][z_1, B_2, z_2] \dots [z_{k-1}, B_k, z_k] \mid \\ & k \geq 1, z_i \in Z'', (z, a, X, B_1 B_2 \dots B_k, z') \in K''\} \end{aligned}$$



Tripelkonstruktion

- **Idee:** Verwendung des Nonterminalalphabets zur Codierung von Zustandsübergängen des PDA.
- Die Regeln „**raten**“ eine Zustandsfolge des Kellerautomaten.
- Die Grammatik $G_C := (V_N, V_T, P, S)$ besteht aus:

$$V_N := \{S\} \cup \{[z, X, z'] \mid X \in \Gamma'', z, z' \in Z''\}$$

$$V_T := \Sigma$$

$$\begin{aligned} P := & \{S \longrightarrow [z, \$, z'] \mid z \in Z''_{\text{start}}, z' \in Z''\} \cup \\ & \{[z, X, z'] \longrightarrow a \mid (z, a, X, \lambda, z') \in K'', a \in \Sigma \cup \{\lambda\}, \\ & X \in \Gamma'', z, z' \in Z''\} \cup \\ & \{[z, X, z_k] \longrightarrow a[z', B_1, z_1][z_1, B_2, z_2] \dots [z_{k-1}, B_k, z_k] \mid \\ & k \geq 1, z_i \in Z'', (z, a, X, B_1 B_2 \dots B_k, z') \in K''\} \end{aligned}$$

- Für diese Grammatik gilt $L(G) = N(C)$.



Beispiel: Tripelkonstruktion

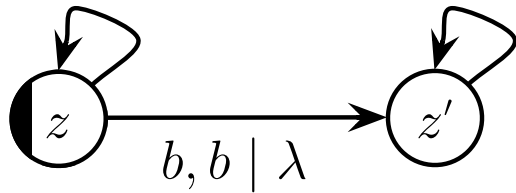
$a, \perp \mid \$b\$$

$a, \$ \mid \$b\$$

$\lambda, \$ \mid \lambda$

$b, b \mid \lambda$

$\lambda, \$ \mid \lambda$



$$V_N := \{S\} \cup \{[z_1, y, z_2] \mid z_1, z_2 \in \{z, z'\} \wedge y \in \{a, b, \perp, \$\}\}$$

$$P := \{S \longrightarrow [z, \perp, z], S \longrightarrow [z, \perp, z']\} \cup$$

$$\{[z, b, z] \longrightarrow b, [z, b, z'] \longrightarrow b\} \cup$$

$$\{[z, \$, z] \longrightarrow \lambda, [z', \$, z'] \longrightarrow \lambda\} \cup$$

$$\{[z, \$, z_3] \longrightarrow a[z, \$, z_1][z_1, b, z_2][z_2, \$, z_3] \mid z_1, z_2, z_3 \in \{z, z'\}\} \cup$$

$$\{[z, \perp, z_3] \longrightarrow a[z, \$, z_1][z_1, b, z_2][z_2, \$, z_3] \mid z_1, z_2, z_3 \in \{z, z'\}\}$$

Die Grammatik ist *nicht* reduziert!

($[z', b, z']$, $[z', b, z]$ und $[z', \$, z]$ sind nicht produktiv)



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:
 1. $L \in Cf$



Die Sprachfamilie Cf

● **Theorem:** Folgende Aussagen sind äquivalent:

1. $L \in Cf$
2. $L = L(A)$ für einen beliebigen PDA A



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:
1. $L \in Cf$
 2. $L = L(A)$ für einen beliebigen PDA A
 3. $L = L(A)$ für einen fast-buchstabierenden PDA A



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:
1. $L \in Cf$
 2. $L = L(A)$ für einen beliebigen PDA A
 3. $L = L(A)$ für einen fast-buchstabierenden PDA A
 4. $L = L(A)$ für einen buchstabierenden PDA A



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:
1. $L \in Cf$
 2. $L = L(A)$ für einen beliebigen PDA A
 3. $L = L(A)$ für einen fast-buchstabierenden PDA A
 4. $L = L(A)$ für einen buchstabierenden PDA A
 5. $L = N(A)$ für einen fast-buchstabierenden PDA A



Die Sprachfamilie Cf

- **Theorem:** Folgende Aussagen sind äquivalent:
 1. $L \in Cf$
 2. $L = L(A)$ für einen beliebigen PDA A
 3. $L = L(A)$ für einen fast-buchstabierenden PDA A
 4. $L = L(A)$ für einen buchstabierenden PDA A
 5. $L = N(A)$ für einen fast-buchstabierenden PDA A
- **Nächste Woche:** Eigenschaften deterministischer Kellerautomaten