F2 — Automaten und formale Sprachen

Berndt Farwer

Fachbereich Informatik

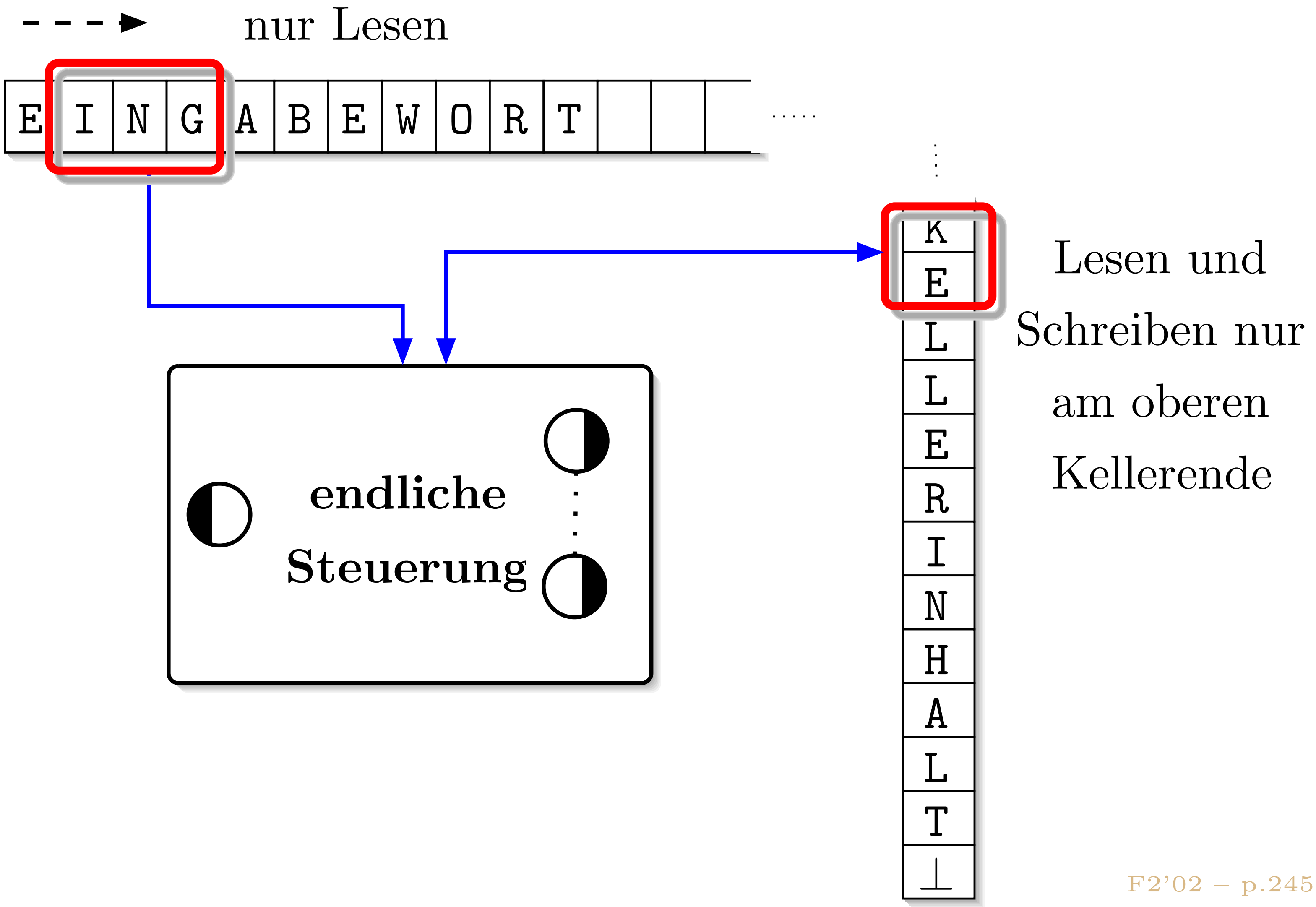
AB „Theoretische Grundlagen der Informatik“ (TGI)

Universität Hamburg

farwer@informatik.uni-hamburg.de

Kellerautomaten

- Eine Kellerautomat ist ein endlicher Automat mit Kellerspeicher:



Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$, 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 $A = (Z, \Sigma, \Gamma, K, Z_{\textsf{start}}, Z_{\textsf{end}}, \perp)$, 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 $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$, 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**.
  - $Z_{\mathsf{start}} \subseteq Z$ ist die Menge der **Startzustände**.
  - $Z_{\mathsf{end}} \subseteq Z$ ist die Menge der **Endzustände**.

Kellerautomat (formal)

- Ein **nichtdeterministischer Kellerautomat** (PDA für *push down automaton*) ist ein Tupel $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \bot)$, 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**.
  - $Z_{\mathsf{start}} \subseteq Z$ ist die Menge der **Startzustände**.
  - $Z_{\mathsf{end}} \subseteq Z$ ist die Menge der **Endzustände**.
  - $\bot \in \Gamma$ 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

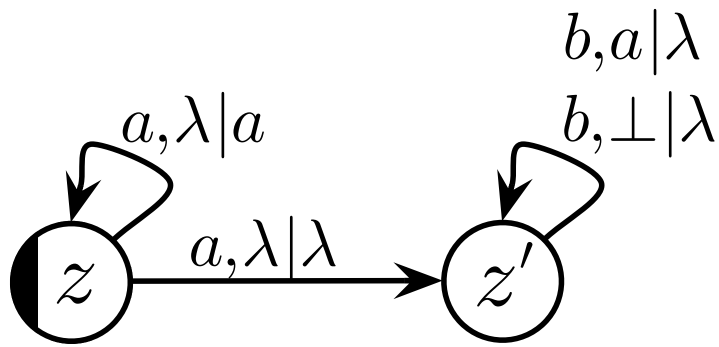
- **Wichtige Notation:** Der Kellerinhalt wird durch ein Wort $v \in \Gamma^*$ so beschrieben, dass das oberste Zeichen des Kellerinhaltes in v ganz am Anfang, d.h. links, steht.
- **Beispiel:** $abab\bot$ bedeutet, dass a das oberste und $\bot$ das unterste Zeichen im Keller ist.

Einige Notationen

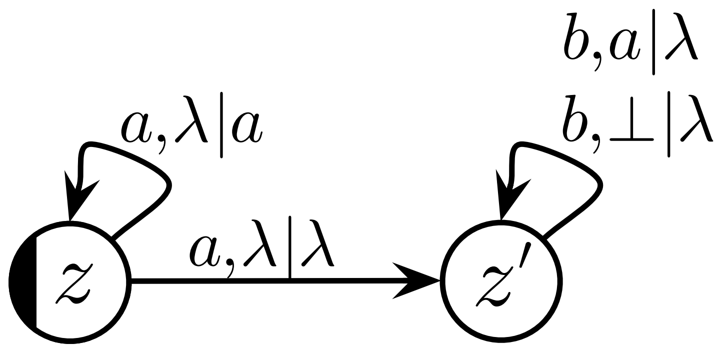
- **Wichtige Notation:** Der Kellerinhalt wird durch ein Wort $v \in \Gamma^*$ so beschrieben, dass das oberste Zeichen des Kellerinhaltes in v ganz am Anfang, d.h. links, steht.
- **Beispiel:** $abab\perp$ bedeutet, dass a das oberste und $\perp$ das unterste Zeichen im Keller ist.
- Zunächst eine weitere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

$$z_1 \xrightarrow{u,v|w} z_2$$

Beispiel

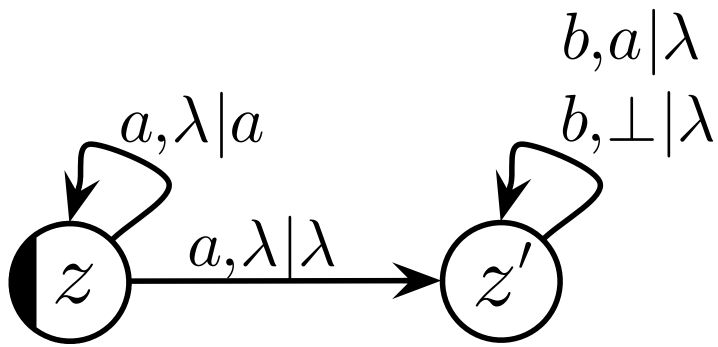


Beispiel



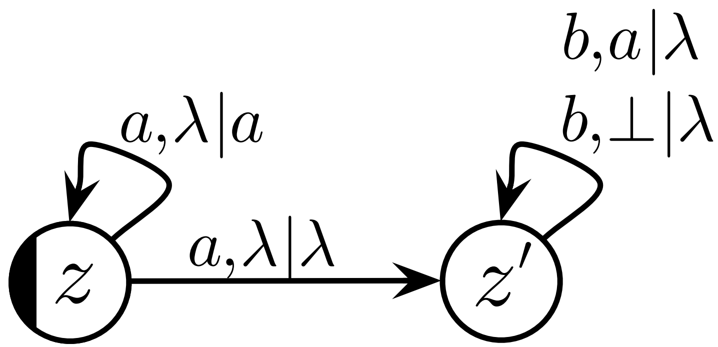
- **Offensichtliche Frage:**

Beispiel



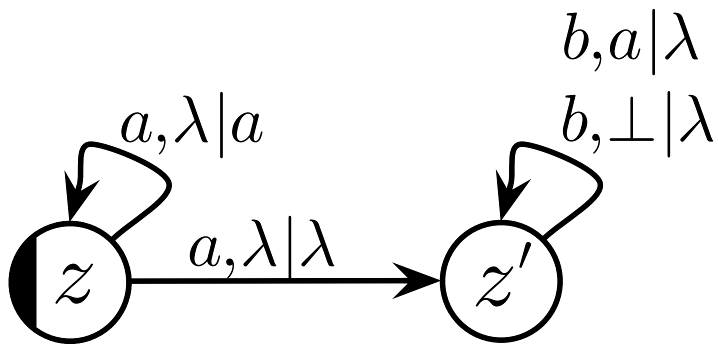
- **Offensichtliche Frage:**
  - Was ist die akzeptierte Sprache eines PDA?

Beispiel



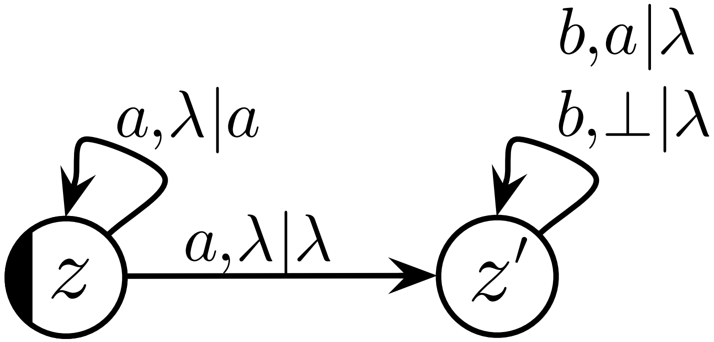
- **Offensichtliche Frage:**
  - Was ist die akzeptierte Sprache eines PDA?
  - ... dazu benötigen wir folgende Begriffe:

Beispiel



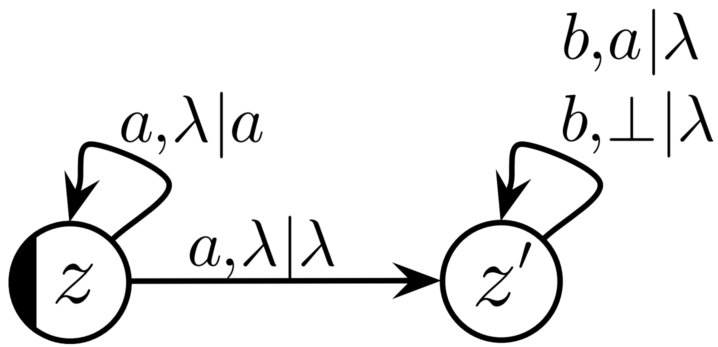
- **Offensichtliche Frage:**
  - Was ist die akzeptierte Sprache eines PDA?
  - ... dazu benötigen wir folgende Begriffe:
    - Rechnung/Erfolgsrechnung eines PDA

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.

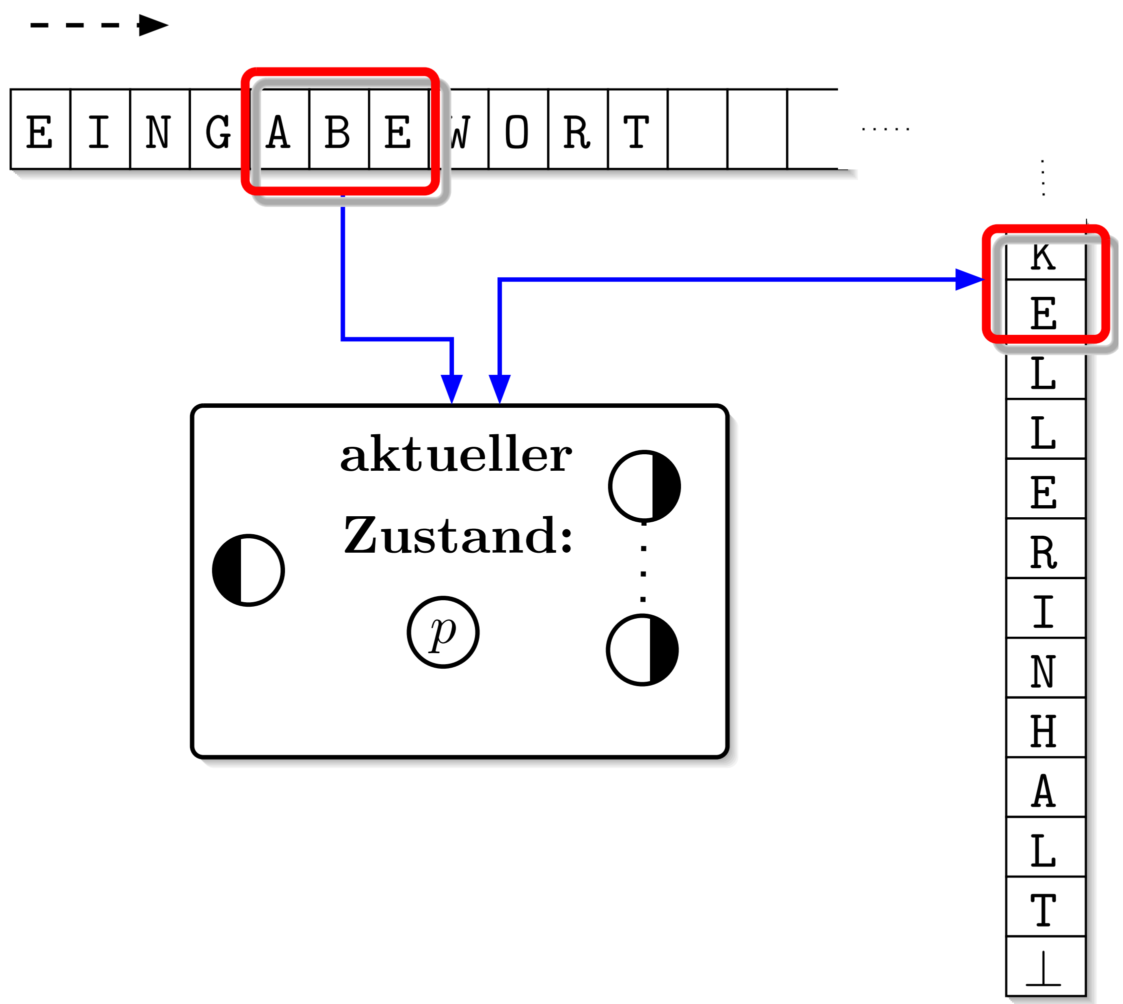
Konfiguration

- 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

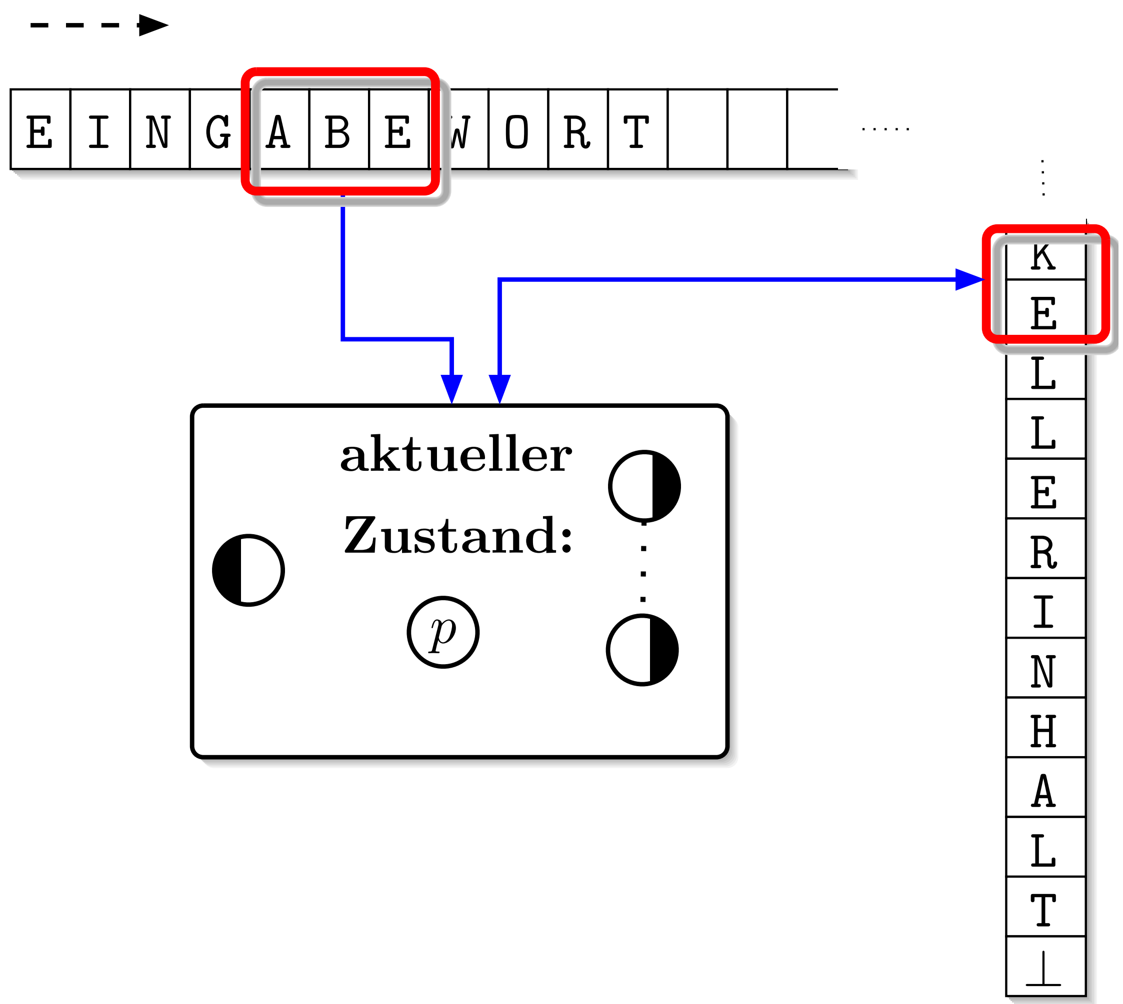
- Als **Konfiguration** wird im Allgemeinen eine vollständige Beschreibung eines bestimmten Systems bezeichnet.
- Dabei zu beachten: Konfigurationen
  - sind **relativ** zu einer festen Struktur.
  - beinhalten eine Beschreibung des **Speicherinhaltes** und des **Systemzustands**
- **Definition:**
 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

Beispiel: Konfigurationen



- Ein PDA . . .

Beispiel: Konfigurationen



- Ein PDA ...
- ...und seine aktuelle Konfiguration:
  KELLERINHALT$\perp p$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) $\longmapsto \subseteq \Gamma^* \cdot Z \cdot \Sigma^* \times \Gamma^* \cdot Z \cdot \Sigma^*$ definiert durch:

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

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) $\longmapsto\ \subseteq \Gamma^* \cdot Z \cdot \Sigma^* \times \Gamma^* \cdot Z \cdot \Sigma^*$ definiert durch:

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

- Wie üblich bezeichnet $\overset{*}{\longmapsto}$ die reflexive, transitive Hülle von $\longmapsto$ und wir schreiben $\underset{A}{\longmapsto}$, 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 $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ **mit Endzustand akzeptierte Sprache** $L(A)$ ist

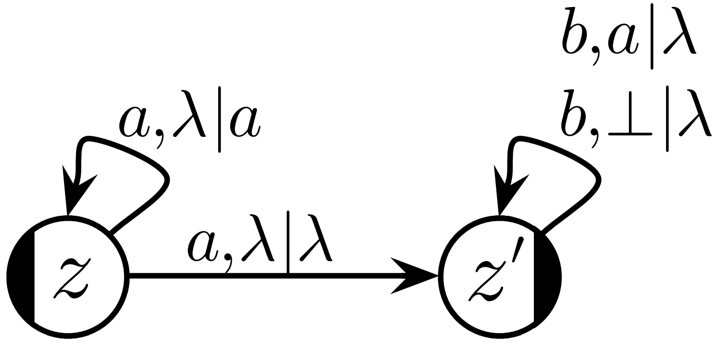
$$\begin{aligned} L(A) \quad := \quad & \{w \in \Sigma^* \mid \exists z_0 \in Z_{\mathsf{start}} : \exists z_e \in Z_{\mathsf{end}} : \\ & \exists v \in \Gamma^* : \perp z_0 w \overset{*}{\vdash} v z_e\} \end{aligned}$$

$L(A)$ 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

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

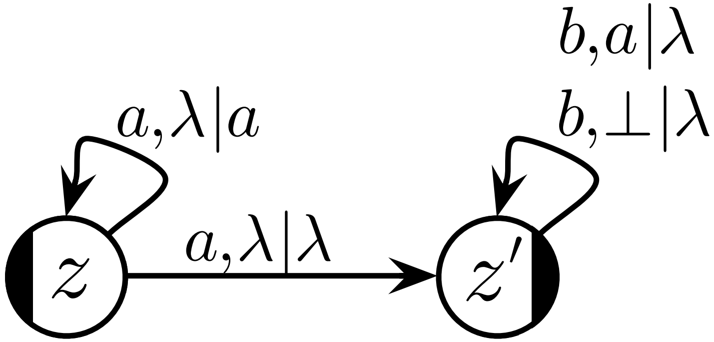
- **Beispiel:**



$L(A)$ für Kellerautomaten

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

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



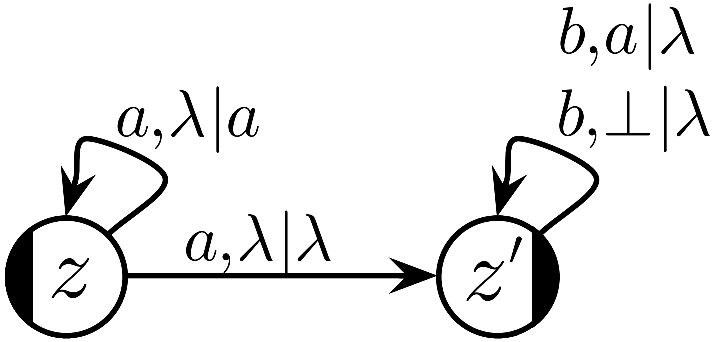
- **Beispiel:**

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

$L(A)$ 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

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

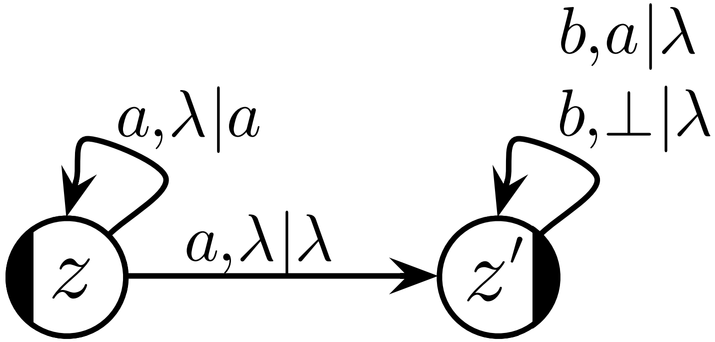


- **Beispiel:**

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

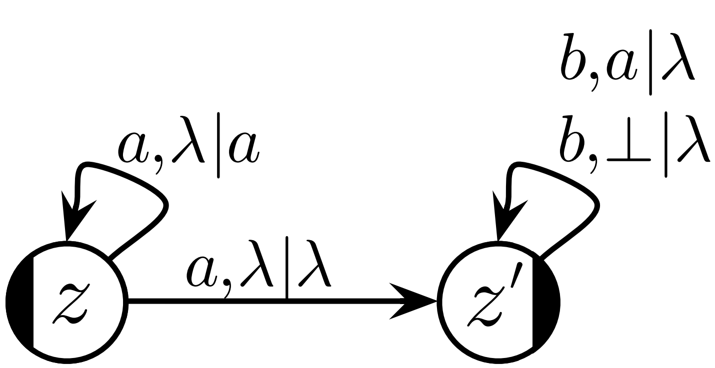
…für jedes gelesene b muss vorher ein a gelesen worden sein!

Beispielrechnungen



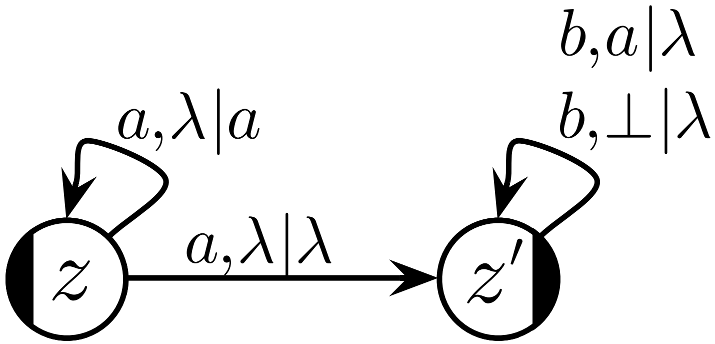
- $\perp\mathbf{z}aaabb$

Beispielrechnungen



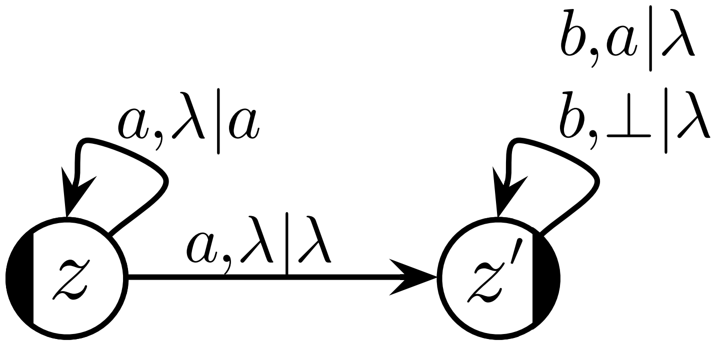
- $\perp \mathbf{z} aaabb$ ist Startkonfiguration.

Beispielrechnungen



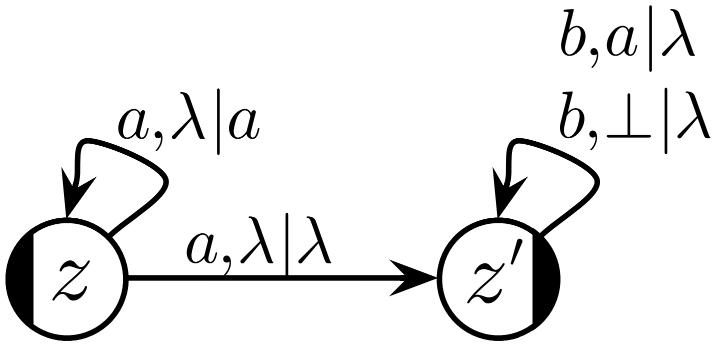
- $\perp \mathbf{z} aaabb \;\vdash\; a \perp \mathbf{z} aabb$

Beispielrechnungen



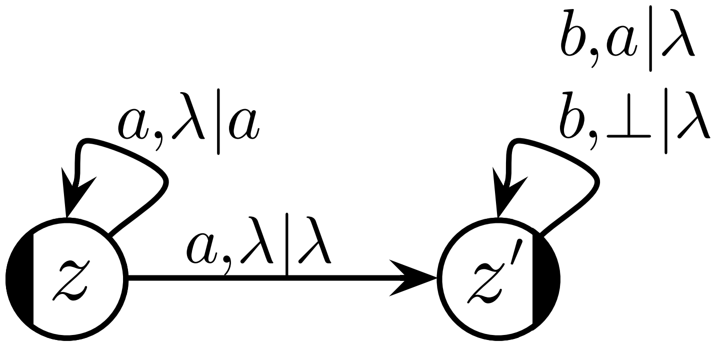
- $\perp \mathbf{z} aaabb \;\vdash\; a\perp \mathbf{z} aabb \;\vdash\; aa\perp \mathbf{z} abb$

Beispielrechnungen



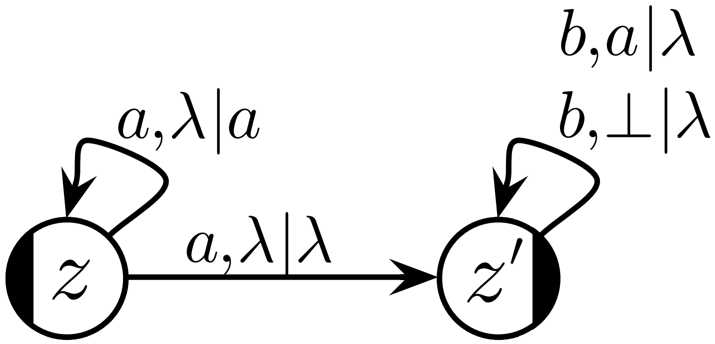
- $\perp\mathbf{z}aaabb \;\longmapsto\; a\perp\mathbf{z}aabb \;\longmapsto\; aa\perp\mathbf{z}abb$
  $\longmapsto\; aaa\perp\mathbf{z}bb$

Beispielrechnungen



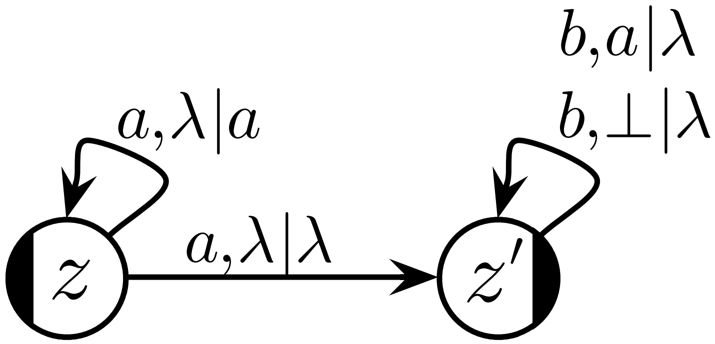
- $\perp\mathbf{z}aaabb \;\vdash\; a\perp\mathbf{z}aabb \;\vdash\; aa\perp\mathbf{z}abb$
  $\vdash\; aaa\perp\mathbf{z}bb$ und hier blockiert der PDA.

Beispielrechnungen



- $\perp \mathbf{z} aaabb \;\vdash\!\!\!- \; a \perp \mathbf{z} aabb \;\vdash\!\!\!- \; aa \perp \mathbf{z} abb$
  $\vdash\!\!\!- \; aaa \perp \mathbf{z} bb$ und hier blockiert der PDA.
- Eine weitere Rechnung für $aaabb$:
  $\perp \mathbf{z} aaabb \vdash\!\!\!- \; a \perp \mathbf{z} aabb \vdash\!\!\!- \; aa \perp \mathbf{z} abb \vdash\!\!\!-$
  $aa \perp \mathbf{z}' bb \vdash\!\!\!- \; a \perp \mathbf{z}' b \vdash\!\!\!- \; \perp \mathbf{z}'$

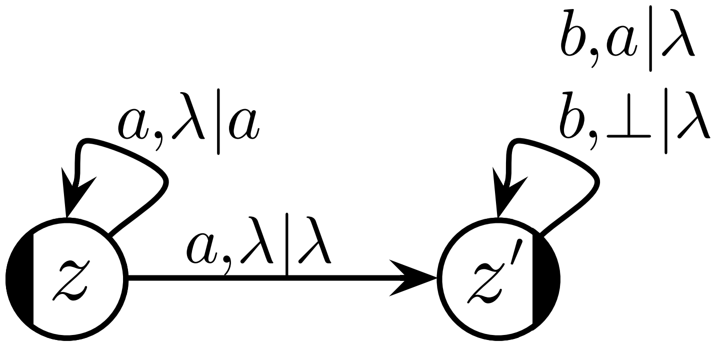
Beispielrechnungen



- $\perp \mathbf{z} aaabb \;\vdash\!\!\!-\; a \perp \mathbf{z} aabb \;\vdash\!\!\!-\; aa \perp \mathbf{z} abb$ $\;\vdash\!\!\!-\; aaa \perp \mathbf{z} bb$ und hier blockiert der PDA.
- Eine weitere Rechnung für $aaabb$:
  $\perp \mathbf{z} aaabb \vdash\!\!\!- a \perp \mathbf{z} aabb \vdash\!\!\!- aa \perp \mathbf{z} abb \vdash\!\!\!-$
  $aa \perp \mathbf{z}' bb \vdash\!\!\!- a \perp \mathbf{z}' b \vdash\!\!\!- \perp \mathbf{z}'$

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

Beispielrechnungen



- $\perp\mathbf{z}aaabb \vdash\!\!\!- a\perp\mathbf{z}aabb \vdash\!\!\!- aa\perp\mathbf{z}abb$ $\vdash\!\!\!- aaa\perp\mathbf{z}bb$ und hier blockiert der PDA.
- Eine weitere Rechnung für $aaabb$:
  $\perp\mathbf{z}aaabb \vdash\!\!\!- a\perp\mathbf{z}aabb \vdash\!\!\!- aa\perp\mathbf{z}abb \vdash\!\!\!-$
  $aa\perp\mathbf{z}'bb \vdash\!\!\!- a\perp\mathbf{z}'b \vdash\!\!\!- \perp\mathbf{z}'$

  Zwar ist der Keller *nicht* leer (er enthält noch $\perp$), aber es ist ein Endzustand erreicht $\Rightarrow$ akzeptieren!
- Für $aabbb$ existiert *keine* Erfolgsrechnung.

$N(A)$ für Kellerautomaten

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

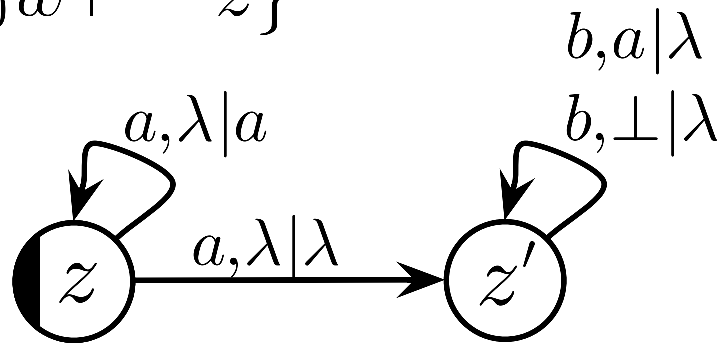
$$N(A) := \{w \in \Sigma^* \mid \exists z_0 \in Z_{\mathsf{start}} : \exists z \in Z : \perp z_0 w \overset{*}{\longmapsto} z\}$$

$N(A)$ für Kellerautomaten

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

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

- **Beispiel:**

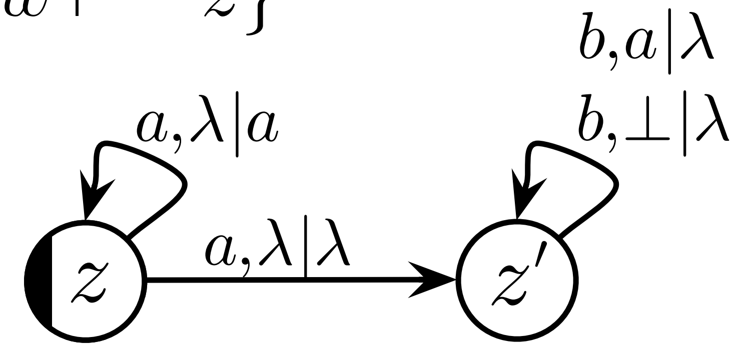


$N(A)$ für Kellerautomaten

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

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

- **Beispiel:**

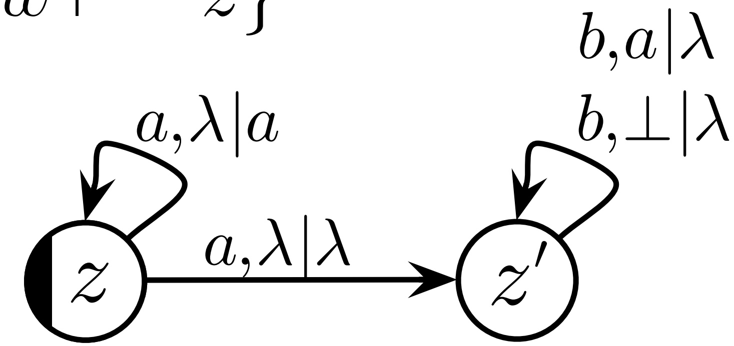


- $N(A) = \{a^n b^n \mid n \in I\!N, n \geq 1\}$

$N(A)$ 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:**
- $N(A) = \{a^n b^n \mid n \in I\!N, n \geq 1\}$
- ... 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 $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ heißt

Eigenschaften von PDAs

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

Eigenschaften von PDAs

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

    $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z$ gilt.
  - **buchstabierend** genau dann, wenn

    $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$ 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 $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z$ gilt.
  - **buchstabierend** genau dann, wenn $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$ 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 $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z$ gilt.
  - **buchstabierend** genau dann, wenn $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$ 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 $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \times \Gamma^* \times Z$ gilt.
  - **buchstabierend** genau dann, wenn $K \subseteq Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Z$ 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 $\forall x \in \Sigma$: $\forall z'' \in Z : (z, x, X, w', z'') \notin K$.

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

Schnitt mit regulären Mengen

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

Konstruktionen auf PDAs

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

Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$.
- **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:
    $Z'' := Z \cup \{[z, u'] \mid (z, u, v, w, z') \in K \ \wedge u' \in \Sigma^*$ ist echter Präfix von u mit $|u'| \geq 1\}$

Konstruktionen auf PDAs

- **Theorem:** Es existiert zu jedem PDA $A = (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ ein äquivalenter fast-buchstabierender PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$.
- **Beweisidee:** Konstruktion eines äquivalenten PDA, der je Schritt max. ein Eingabesymbol liest.
  - Speichern der jeweils gelesenen Symbolkette in einem neuen Zustand: $Z'' := Z \cup \{[z, u'] \mid (z, u, v, w, z') \in K \ \wedge u' \in \Sigma^*$ ist echter Präfix von u mit $|u'| \geq 1\}$
  - Für Kante $k = (z, u_1u_2 \ldots u_n, v, w, z') \in K$ neu: $(z, u_1, v, \lambda, [z', u_1]), ([z', u_1], u_2, \lambda, \lambda, [z', u_1u_2]), \ldots, ([z', u_1 \ldots u_{n-1}], u_n, \lambda, w, z')$

CFG vs. PDA

- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ 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 $A_G := (Z, \Sigma, \Gamma, K, \ Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ 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 $A_G := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ 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 $A_G := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ mit drei Zuständen: $Z := \{z_0, z_1, z_2\}$

CFG vs. PDA

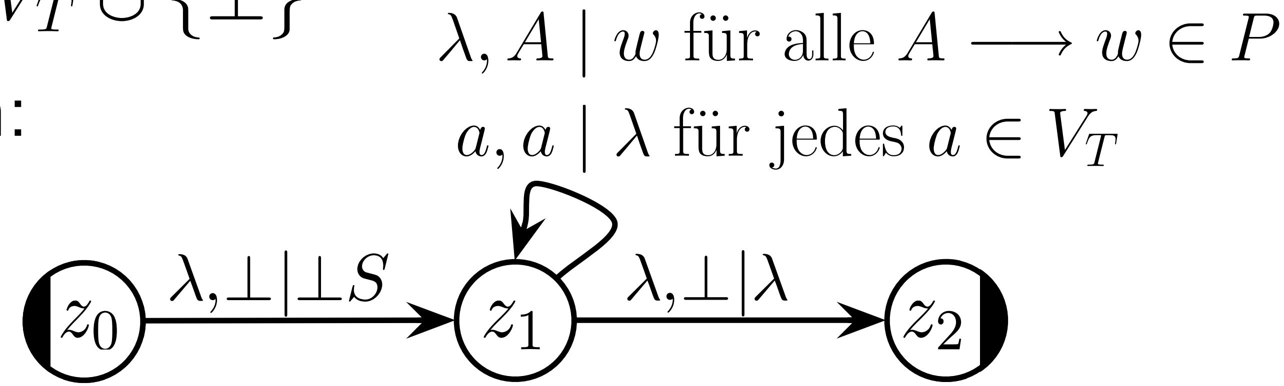
- **Theorem:** Zu jeder kontextfreien Grammatik $G = (V_N, V_T, P, S)$ kann effektiv ein Kellerautomat $A_G := (Z, \Sigma, \Gamma, K, Z_{\text{start}}, Z_{\text{end}}, \perp)$ konstruiert werden, für den $N(A_G) = L(A_G) = L(G)$ gilt.
- **Beweis:** Sei $G = (V_N, V_T, P, S)$ eine CFG.
- Konstruiere $A_G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



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

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

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

- **Lemma:** Es existiert zu jedem PDA $A := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ ein PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ 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
  $A := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ ein PDA
  $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ mit
  - neuem Kellersymbol \$, also $\Gamma' := \Gamma \uplus \{\$\}$;
  - neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
  - $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp\$, z) \mid z \in Z_{\mathsf{start}}\}$ „Initialisieren“

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

- **Lemma:** Es existiert zu jedem PDA
  $A := (Z, \Sigma, \Gamma, K, Z_{\mathsf{start}}, Z_{\mathsf{end}}, \perp)$ ein PDA
  $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\mathsf{start}}, Z'_{\mathsf{end}}, \perp)$ mit
  - neuem Kellersymbol $\$$, also $\Gamma' := \Gamma \uplus \{\$\}$;
  - neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
  - $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp\$, z) \mid z \in Z_{\mathsf{start}}\}$ „Initialisieren“
  - $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\text{empty}}) \mid z_e \in Z_{\mathsf{end}}\}$ „Übergänge zum den Keller leerenden Zustand z_{empty}“

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

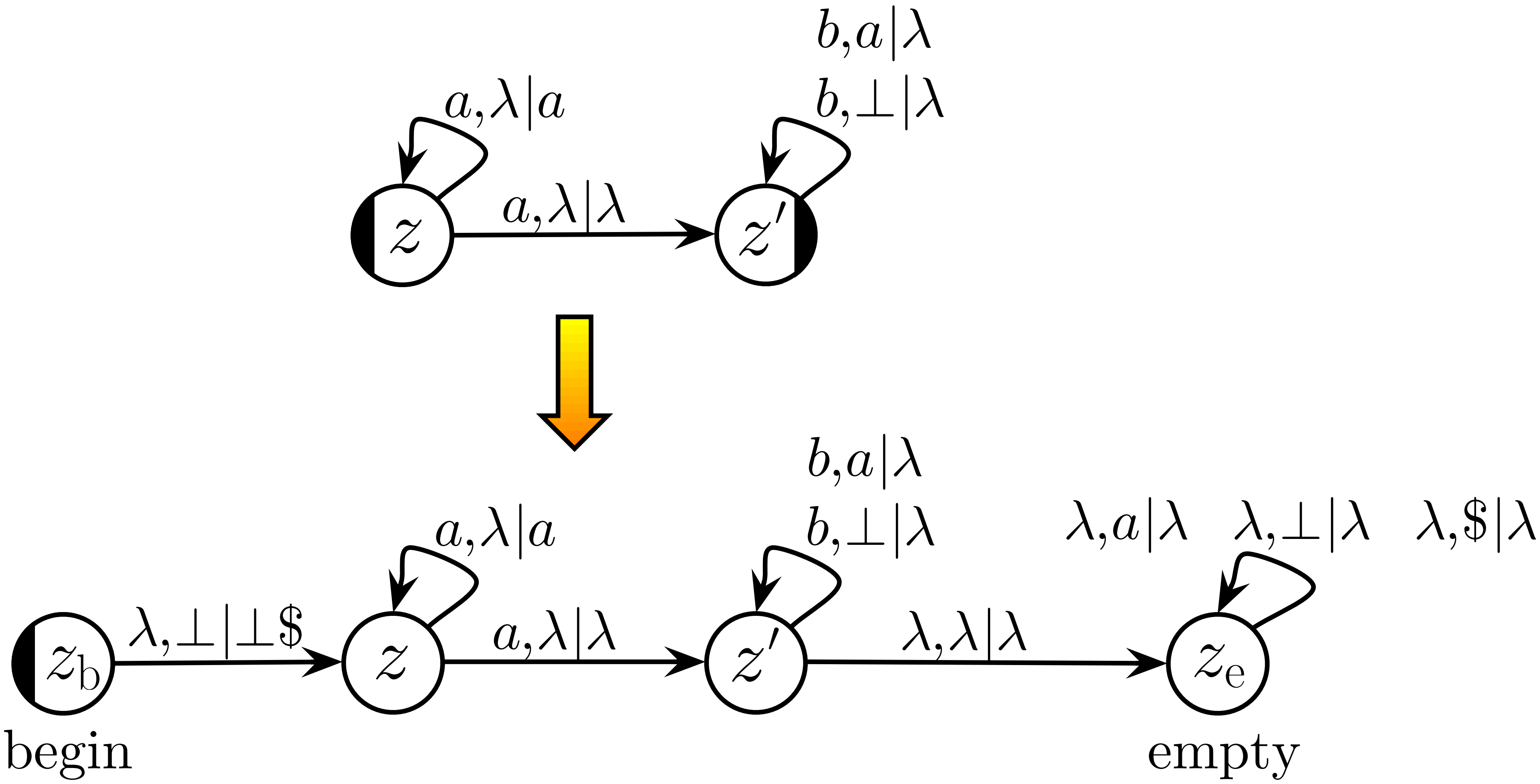
- **Lemma:** Es existiert zu jedem PDA
  $A := (Z, \Sigma, \Gamma, K, Z_{\textsf{start}}, Z_{\textsf{end}}, \perp)$ ein PDA
  $B := (Z', \Sigma, \Gamma', K', Z'_{\textsf{start}}, Z'_{\textsf{end}}, \perp)$ mit $L(A) = N(B)$.
- **Beweis:** O.B.d.A. sei A fast-buchstabierend. Konstruiere PDA $B := (Z', \Sigma, \Gamma', K', Z'_{\textsf{start}}, Z'_{\textsf{end}}, \perp)$ mit
  - neuem Kellersymbol \$, also $\Gamma' := \Gamma \uplus \{\$\}$;
  - neuen Zuständen $Z' := Z \uplus \{z_{\text{begin}}, z_{\text{empty}}\}$
  - $K_1 := \{(z_{\text{begin}}, \lambda, \perp, \perp\$, z) \mid z \in Z_{\textsf{start}}\}$ „Initialisieren“
  - $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\text{empty}}) \mid z_e \in Z_{\textsf{end}}\}$ „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 $Z' := Z \uplus \{z_{\mathrm{begin}}, z_{\mathrm{empty}}\}$
  - $K_1 := \{(z_{\mathrm{begin}}, \lambda, \perp, \perp\$, z) \mid z \in Z_{\text{start}}\}$ „Initialisieren“
  - $K_2 := \{(z_e, \lambda, \lambda, \lambda, z_{\mathrm{empty}}) \mid z_e \in Z_{\text{end}}\}$ „Übergänge zum den Keller leerenden Zustand z_{empty}“
  - $K_3 := \{(z_{\mathrm{empty}}, \lambda, a, \lambda, z_{\mathrm{empty}}) \mid a \in \Gamma'\}$ „Leeren“
  - $K' := K \cup K_1 \cup K_2 \cup K_3$

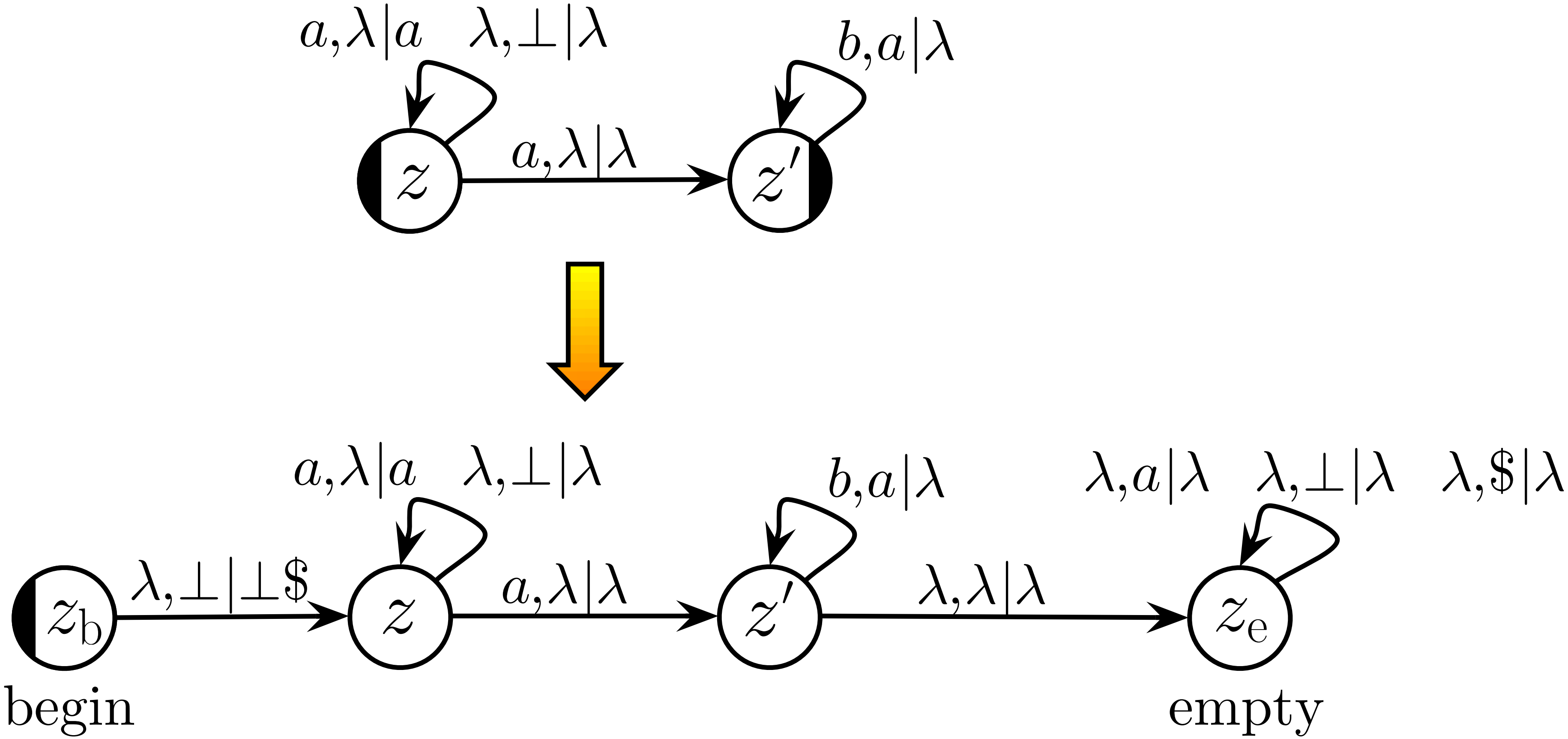
Beispiel: $L(A) = N(B)$



…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 $L(A) \in \mathcal{C}f$?

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

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

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

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

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

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

- **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 $L(A) \in \mathcal{C}f$?

- **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 $L(A) \in \mathcal{C}f$?

- **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 $L(A) \in \mathcal{C}f$?

- **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:

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

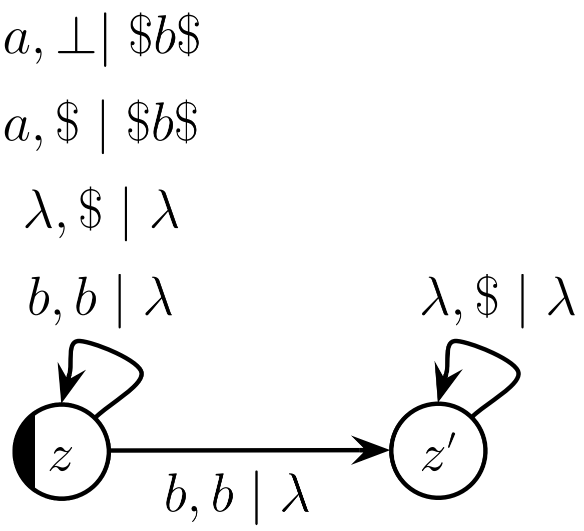
Tripelkonstruktion

- **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:

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

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

Beispiel: Tripelkonstruktion



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

$$
\begin{aligned}
P := &\{S \longrightarrow [z, \perp, z],\; S \longrightarrow [z, \perp, z']\} \cup \\
&\{[z, b, z] \longrightarrow b,\; [z, b, z'] \longrightarrow b\} \cup \\
&\{[z, \$, z] \longrightarrow \lambda,\; [z', \$, z'] \longrightarrow \lambda\} \cup \\
&\{[z, \$, z_3] \longrightarrow a[z, \$, z_1][z_1, b, z_2][z_2, \$, z_3] \mid z_1, z_2, z_3 \in \{z, z'\}\} \cup \\
&\{[z, \perp, z_3] \longrightarrow a[z, \$, z_1][z_1, b, z_2][z_2, \$, z_3] \mid z_1, z_2, z_3 \in \{z, z'\}\}
\end{aligned}
$$

Die Grammatik ist *nicht* reduziert!
($[z', b, z']$, $[z', b, z]$ und $[z', \$, z]$ sind nicht produktiv)

Die Sprachfamilie $\mathcal{C}f$

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

Die Sprachfamilie $\mathcal{C}f$

- **Theorem:** Folgende Aussagen sind äquivalent:
  1. $L \in \mathcal{C}f$

Die Sprachfamilie $\mathcal{C}f$

- **Theorem:** Folgende Aussagen sind äquivalent:
  1. $L \in \mathcal{C}f$
  2. $L = L(A)$ für einen beliebigen PDA A

Die Sprachfamilie $\mathcal{C}f$

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

Die Sprachfamilie $\mathcal{C}f$

- **Theorem:** Folgende Aussagen sind äquivalent:
  1. $L \in \mathcal{C}f$
  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 $\mathcal{C}f$

- **Theorem:** Folgende Aussagen sind äquivalent:
  1. $L \in \mathcal{C}f$
  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 $\mathcal{C}f$

- **Theorem:** Folgende Aussagen sind äquivalent:
  1. $L \in \mathcal{C}f$
  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
- **Was nun?** Eigenschaften deterministischer Kellerautomaten

Präfixfreiheit

- **Definition:** Eine Sprache $L \subseteq \Sigma^*$ heißt **präfixfrei** gdw.

$$\forall w \in L : \forall v \in \Sigma^* : (wv \in L) \to (v = \lambda).$$

Präfixfreiheit

- **Definition:** Eine Sprache $L \subseteq \Sigma^*$ heißt **präfixfrei** gdw.

$$\forall w \in L : \forall v \in \Sigma^* : (wv \in L) \rightarrow (v = \lambda).$$

- $\min(L) := \{w \in L \mid (w = uv \wedge u \in L) \rightarrow (v = \lambda)\}$ bezeichnet den **präfixfreien Anteil** der Sprache L.

Präfixfreiheit

- **Definition:** Eine Sprache $L \subseteq \Sigma^*$ heißt **präfixfrei** gdw.

$$\forall w \in L : \forall v \in \Sigma^* : (wv \in L) \rightarrow (v = \lambda).$$

- $\min(L) := \{w \in L \mid (w = uv \wedge u \in L) \rightarrow (v = \lambda)\}$ bezeichnet den **präfixfreien Anteil** der Sprache L.
- **Beispiel:** Sei $M := \{a^m b^n \mid m, n \in \mathbb{N} : n \geq m\}$, dann ist $\min(M) = \{a^n b^n \mid n \in \mathbb{N}\}$, denn es gibt für $n > m$ mindestens einen echten Präfix eines Wortes $a^m b^n$, der wieder als Wort der Menge M vorkommt. Also muss $n = m$ gelten.

Eigenschaften von $det\,\mathcal{C}f$

- **Theorem:** Eine deterministische kontextfreie Sprache $L \in det\,\mathcal{C}f$ ist präfixfrei gdw. ein DPDA A existiert, mit $L = N(A)$.

Eigenschaften von $det\,\mathcal{C}f$

- **Theorem:** Eine deterministische kontextfreie Sprache $L \in det\,\mathcal{C}f$ ist präfixfrei gdw. ein DPDA A existiert, mit $L = N(A)$.
- **Beweisidee:**

Eigenschaften von $det\,\mathcal{C}f$

- **Theorem:** Eine deterministische kontextfreie Sprache $L \in det\,\mathcal{C}f$ ist präfixfrei gdw. ein DPDA A existiert, mit $L = N(A)$.
- **Beweisidee:**

  $\Rightarrow$ DPDA für L kann kein Wort akzeptieren, nachdem ein Endzustand verlassen wurde. Hinzufügen eines neuen Zustands, der den Keller leert und genau von den vormaligen Endzuständen erreichbar ist.

Eigenschaften von $det\,\mathcal{C}f$

- **Theorem:** Eine deterministische kontextfreie Sprache $L \in det\,\mathcal{C}f$ ist präfixfrei gdw. ein DPDA A existiert, mit $L = N(A)$.
- **Beweisidee:**

  $\Rightarrow$ DPDA für L kann kein Wort akzeptieren, nachdem ein Endzustand verlassen wurde. Hinzufügen eines neuen Zustands, der den Keller leert und genau von den vormaligen Endzuständen erreichbar ist.

  $\Leftarrow$ DPDAs können bei leerem Keller keine weiteren Bewegungen vornehmen!

Eigenschaften von $det\,\mathcal{C}f$

- **Korrollar** Die präfixfreien deterministisch kontextfreien Sprachen bilden eine echte Teilfamilie der Familie $det\,\mathcal{C}f$.

Eigenschaften von $det\,\mathcal{C}f$

- **Korrollar** Die präfixfreien deterministisch kontextfreien Sprachen bilden eine echte Teilfamilie der Familie $det\,\mathcal{C}f$.
- **Beweis:** Die Menge $\{a\}^*\{b\}^* = \{a^n b^m \mid m, n \in I\!N\}$ ist, da regulär, deterministisch kontextfrei, aber offensichtlich nicht präfixfrei.

Abschlusseigenschaften

- **Theorem:** Für $L \in det\,\mathcal{C}f$ ist auch $\min(L) \in det\,\mathcal{C}f$.

Abschlusseigenschaften

- **Theorem:** Für $L \in det\,\mathcal{C}f$ ist auch $\min(L) \in det\,\mathcal{C}f$.
- **Beweisidee:** Kanten aus Endzuständen entfernen.

Abschlusseigenschaften

- **Theorem:** Für $L \in \mathit{det}\,\mathcal{C}\!f$ ist auch $\min(L) \in \mathit{det}\,\mathcal{C}\!f$.
- **Beweisidee:** Kanten aus Endzuständen entfernen.
- **Theorem:** $\mathit{det}\,\mathcal{C}\!f \wedge \mathcal{R}eg \subseteq \mathit{det}\,\mathcal{C}\!f$, d.h., die Familie der deterministischen kontextfreien Sprachen ist gegenüber Durchschnittsbildung mit regulären Mengen abgeschlossen.

Abschlusseigenschaften

- **Theorem:** Für $L \in det\,\mathcal{C}f$ ist auch $\min(L) \in det\,\mathcal{C}f$.
- **Beweisidee:** Kanten aus Endzuständen entfernen.
- **Theorem:** $det\,\mathcal{C}f \wedge \mathcal{R}eg \subseteq det\,\mathcal{C}f$, d.h., die Familie der deterministischen kontextfreien Sprachen ist gegenüber Durchschnittsbildung mit regulären Mengen abgeschlossen.
- **Beweisidee:** Produktautomat zweier DPDAs

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$
  - Wäre nun $PAL \in det\,\mathcal{C}f$, dann wäre auch $K := PAL \cap (ab)^+(ba)^+(ab)^+(ba)^+ \in det\,\mathcal{C}f$.

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$
  - Wäre nun $PAL \in det\,\mathcal{C}f$, dann wäre auch $K := PAL \cap (ab)^+(ba)^+(ab)^+(ba)^+ \in det\,\mathcal{C}f$.
  - Dann ist auch $\min(K) \in det\,\mathcal{C}f$.

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$
  - Wäre nun $PAL \in det\,\mathcal{C}f$, dann wäre auch $K := PAL \cap (ab)^+(ba)^+(ab)^+(ba)^+ \in det\,\mathcal{C}f$.
  - Dann ist auch $\min(K) \in det\,\mathcal{C}f$.
  - Nun ergibt sich: $\min(K) = \{(ab)^m(ba)^n(ab)^n(ba)^m \mid 0 \leq n \leq m\}$

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$
  - Wäre nun $PAL \in det\,\mathcal{C}f$, dann wäre auch $K := PAL \cap (ab)^+(ba)^+(ab)^+(ba)^+ \in det\,\mathcal{C}f$.
  - Dann ist auch $\min(K) \in det\,\mathcal{C}f$.
  - Nun ergibt sich: $\min(K) = \{(ab)^m(ba)^n(ab)^n(ba)^m \mid 0 \leq n \leq m\}$
  - Diese Sprache ist nicht einmal mehr kontextfrei, wie man mit dem $uvwxy$-Theorem zeigen kann.

Eindeutigkeit vs. Determinismus

- **Theorem:** Die kontextfreie Sprache PAL ist eindeutig, aber nicht deterministisch.
- **Beweis:**
  - Eindeutigkeit: $S \longrightarrow aSa \mid bSb \mid a \mid b \mid \lambda$
  - Wäre nun $PAL \in det\,\mathcal{C}f$, dann wäre auch $K := PAL \cap (ab)^+(ba)^+(ab)^+(ba)^+ \in det\,\mathcal{C}f$.
  - Dann ist auch $\min(K) \in det\,\mathcal{C}f$.
  - Nun ergibt sich: $\min(K) = \{(ab)^m(ba)^n(ab)^n(ba)^m \mid 0 \leq n \leq m\}$
  - Diese Sprache ist nicht einmal mehr kontextfrei, wie man mit dem $uvwxy$-Theorem zeigen kann.
  - Also kann PAL selbst nicht deterministisch kontextfrei gewesen sein.

Komplementbildung

- **Theorem:** $\mathit{det}\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in \mathit{det}\,\mathcal{C}f : \overline{L} \in \mathit{det}\,\mathcal{C}f$.

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!
- Gründe für das Nicht-Akzeptieren einer Eingabe beim DPDA:

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!
- Gründe für das Nicht-Akzeptieren einer Eingabe beim DPDA:
  1. Die Rechnung bricht ab, weil der Keller leer ist.

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!
- Gründe für das Nicht-Akzeptieren einer Eingabe beim DPDA:
  1. Die Rechnung bricht ab, weil der Keller leer ist.
  2. Das Wort w wird mangels Folgekonfiguration nicht vollständig gelesen.

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!
- Gründe für das Nicht-Akzeptieren einer Eingabe beim DPDA:
  1. Die Rechnung bricht ab, weil der Keller leer ist.
  2. Das Wort w wird mangels Folgekonfiguration nicht vollständig gelesen.
  3. Der DPDA gerät in eine Endlosschleife, ohne weitere Symbole von der Eingabe zu lesen.

Komplementbildung

- **Theorem:** $det\,\mathcal{C}f$ ist gegen Komplementbildung abgeschlossen, d.h. $\forall L \in det\,\mathcal{C}f : \overline{L} \in det\,\mathcal{C}f$.
- **Beweisidee:** Endzustände mit Nicht-Endzuständen zu vertauschen reicht hier nicht aus!
- Gründe für das Nicht-Akzeptieren einer Eingabe beim DPDA:
  1. Die Rechnung bricht ab, weil der Keller leer ist.
  2. Das Wort w wird mangels Folgekonfiguration nicht vollständig gelesen.
  3. Der DPDA gerät in eine Endlosschleife, ohne weitere Symbole von der Eingabe zu lesen.
  4. Erfolgsrechnung, bei der der DPDA ohne weiteres Lesen der Eingabe abwechselnd in End- und Nicht-Endzustände gerät.

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Zu 2: Einführen einer Senke, in die bisher in anderen Zuständen nicht definierte Folgekonfigurationen führen.

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Zu 2: Einführen einer Senke, in die bisher in anderen Zuständen nicht definierte Folgekonfigurationen führen.

Zu 3: Zwei Unter-Fälle:

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Zu 2: Einführen einer Senke, in die bisher in anderen Zuständen nicht definierte Folgekonfigurationen führen.

Zu 3: Zwei Unter-Fälle:

a): In der λ-Schleife wird der Keller nie kürzer, aber ein Endzustand kommt vor. Neuer Endzustand und Schleifenanfangskante dahin „umbiegen“, Keller leeren.

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Zu 2: Einführen einer Senke, in die bisher in anderen Zuständen nicht definierte Folgekonfigurationen führen.

Zu 3: Zwei Unter-Fälle:

a): In der λ-Schleife wird der Keller nie kürzer, aber ein Endzustand kommt vor. Neuer Endzustand und Schleifenanfangskante dahin „umbiegen“, Keller leeren.

b): Falls die Konfiguration xq der Beginn einer λ-Schleife ist, in der niemals ein Endzustand vorkommt: Keller leeren.

Beweisansatz

Zu 1: Neues Kellerbodenzeichen vor das alte setzen.

Zu 2: Einführen einer Senke, in die bisher in anderen Zuständen nicht definierte Folgekonfigurationen führen.

Zu 3: Zwei Unter-Fälle:

a): In der λ-Schleife wird der Keller nie kürzer, aber ein Endzustand kommt vor. Neuer Endzustand und Schleifenanfangskante dahin „umbiegen“, Keller leeren.

b): Falls die Konfiguration xq der Beginn einer λ-Schleife ist, in der niemals ein Endzustand vorkommt: Keller leeren.

Fall 4: Zu kompliziert . . .

Wichtig für Syntaxanalyse

- **Theorem:** Jede Sprache aus $det\,\mathcal{C}f$ ist eindeutig.

$$\begin{tikzcd}
L \in det\mathcal{C}f \arrow[d] & \\
L\{\cent\} \in det\mathcal{C}f \text{ und präfixfrei} \arrow[d] & \\
L\{\cent\} = N(A) \text{ für DPDA } A \arrow[r, Rightarrow, "\text{Tripelkonstruktion}"] \arrow[d] & L(G) = L\{\cent\} \arrow[d] \\
\text{jede Rechnung} \arrow[r, dashed] \arrow[d] & \text{Linksableitung} \arrow[d] \\
\text{jedes } w\cent \text{ hat genau eine Erfolgsrechnung} \arrow[r, dashed] & \text{genau ein Ableitungsbaum für } w\cent \arrow[d] \\
& \text{aus } G \text{ für } L\{\cent\} \text{ eindeutig folgt } G' \text{ für } L \text{ eindeutig}
\end{tikzcd}$$