

Theoretische Grundlagen der Informatik

Übung

8. Übungstermin · 16. Januar 2020
Guido Brückner

INSTITUT FÜR THEORETISCHE INFORMATIK · LEHRSTUHL ALGORITHMIK

Anmeldung zur Hauptklausur:

→ Anmeldeschluss

15.03.2020

- Eine Anmeldung zur Hauptklausur ist nach Anmeldeschluss nicht mehr möglich!
- Anmeldung erfolgt in der Regel online über das Studierendenportal.
- Bei Problemen bitte bei Ioana Gheata melden.

Inhalt

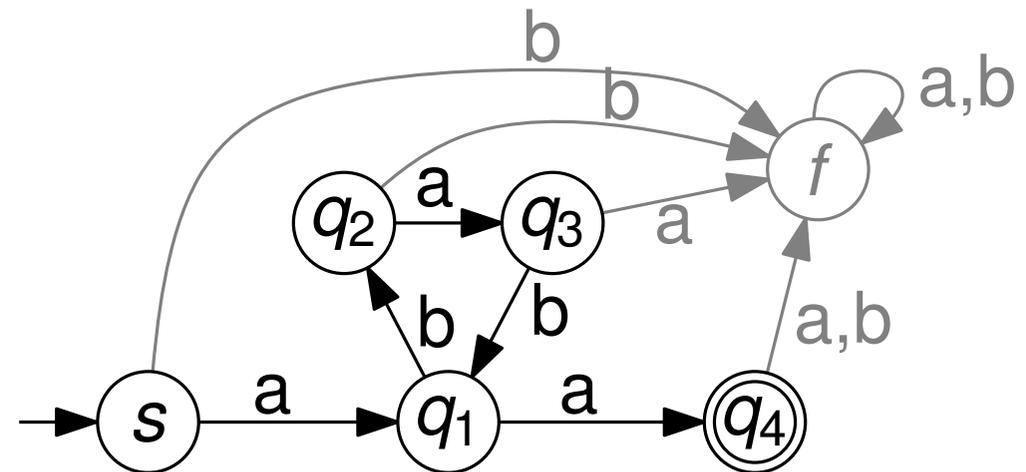
- Kontextfreie Grammatiken
 - Pumpinglemma
- Greibach-Normalform
- Kellerautomaten

Wiederholung: Pumpingl. für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n \text{ und } v \neq \varepsilon$$

existiert, bei der auch $uv^i x \in L$ ist für alle $i \in \mathbb{N}_0$.



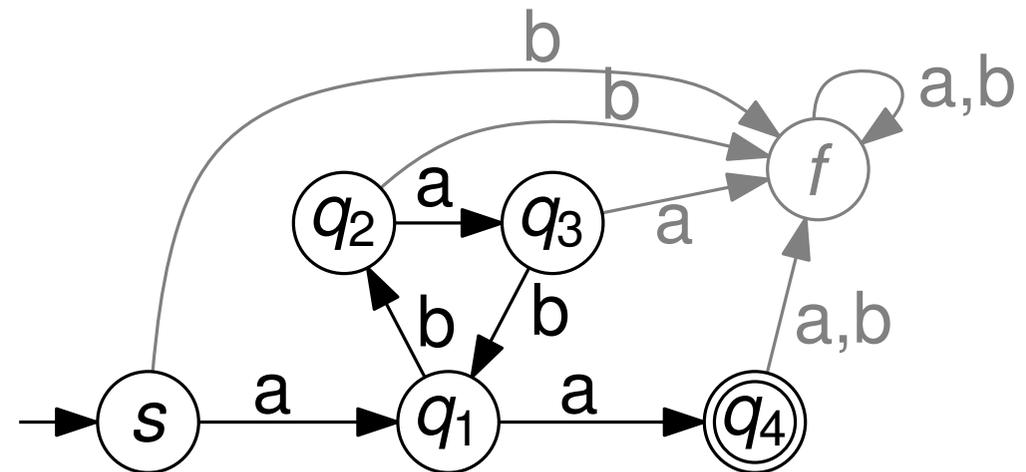
Wiederholung: Pumpingl. für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n \text{ und } v \neq \varepsilon$$

existiert, bei der auch $uv^i x \in L$ ist für alle $i \in \mathbb{N}_0$.

Erklärung: Sei L reguläre Sprache und \mathcal{A} entsprechender endlicher Automat.



Wiederholung: Pumpingl. für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

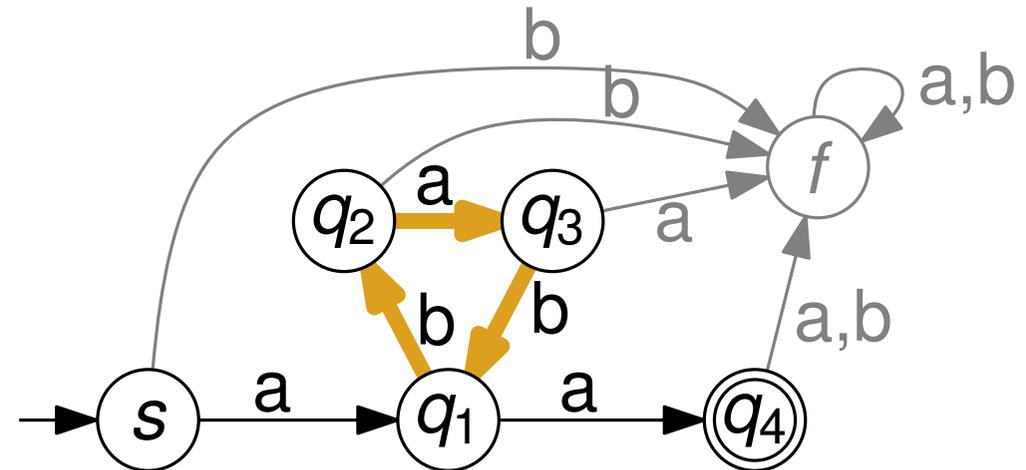
$$w = uvx \text{ mit } |uv| \leq n \text{ und } v \neq \varepsilon$$

existiert, bei der auch $uv^i x \in L$ ist für alle $i \in \mathbb{N}_0$.

Erklärung: Sei L reguläre Sprache und \mathcal{A} entsprechender endlicher Automat.

Egal wie viele Zustände für \mathcal{A} verwendet werden, für jedes Wort $w \in L$, das mehr Zeichen hat als \mathcal{A} Zustände, gilt:

Während der Abarbeitung von w durchläuft man einen **Zyklus Z** in \mathcal{A} .



Wiederholung: Pumpingl. für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n \text{ und } v \neq \varepsilon$$

existiert, bei der auch $uv^i x \in L$ ist für alle $i \in \mathbb{N}_0$.

Erklärung: Sei L reguläre Sprache und \mathcal{A} entsprechender endlicher Automat.

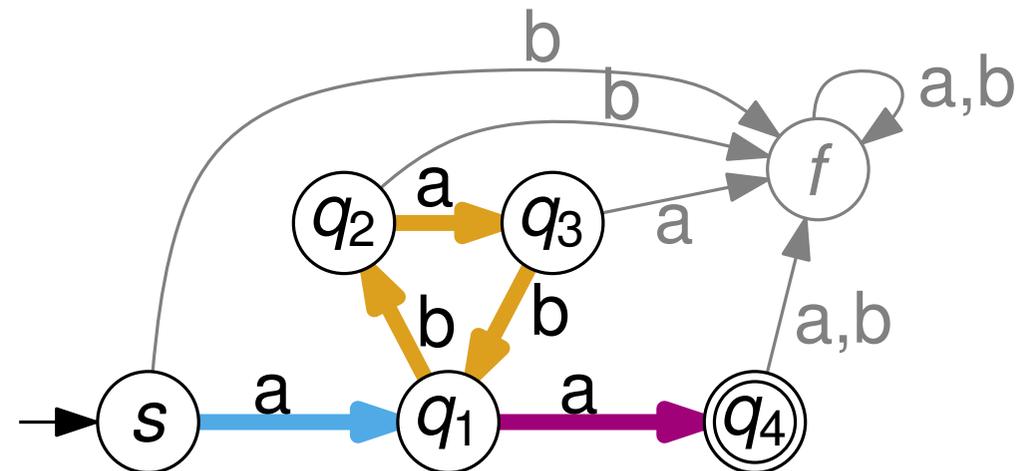
Egal wie viele Zustände für \mathcal{A} verwendet werden, für jedes Wort $w \in L$, das mehr Zeichen hat als \mathcal{A} Zustände, gilt:

Während der Abarbeitung von w durchläuft man einen Zyklus \mathcal{Z} in \mathcal{A} .

Sei

- u das Teilwort von w , das vor \mathcal{Z} ,
- v das Teilwort von w , das in \mathcal{Z} , und
- x das Teilwort von w , das nach \mathcal{Z}

abgearbeitet wird.



Wiederholung: Pumpingl. für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, so dass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n \text{ und } v \neq \varepsilon$$

existiert, bei der auch $uv^i x \in L$ ist für alle $i \in \mathbb{N}_0$.

Erklärung: Sei L reguläre Sprache und \mathcal{A} entsprechender endlicher Automat.

Egal wie viele Zustände für \mathcal{A} verwendet werden, für jedes Wort $w \in L$, das mehr Zeichen hat als \mathcal{A} Zustände, gilt:

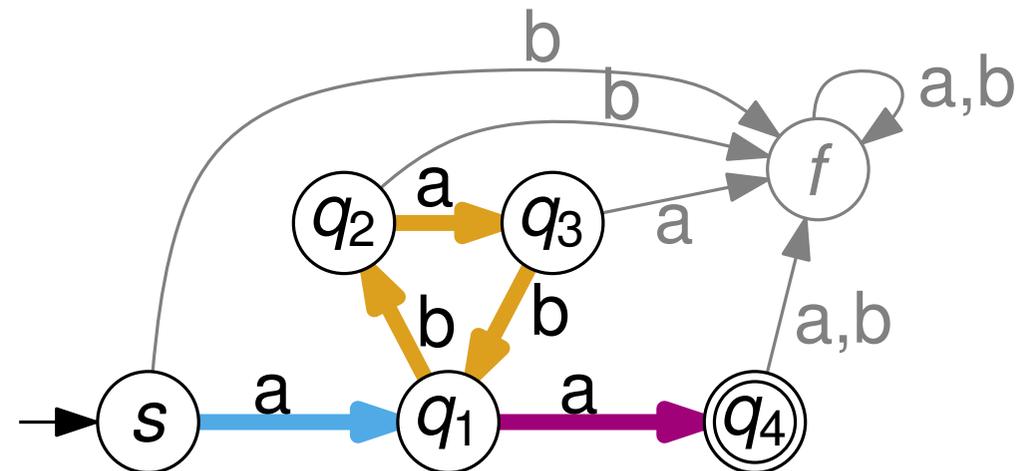
Während der Abarbeitung von w durchläuft man einen Zyklus \mathcal{Z} in \mathcal{A} .

Sei

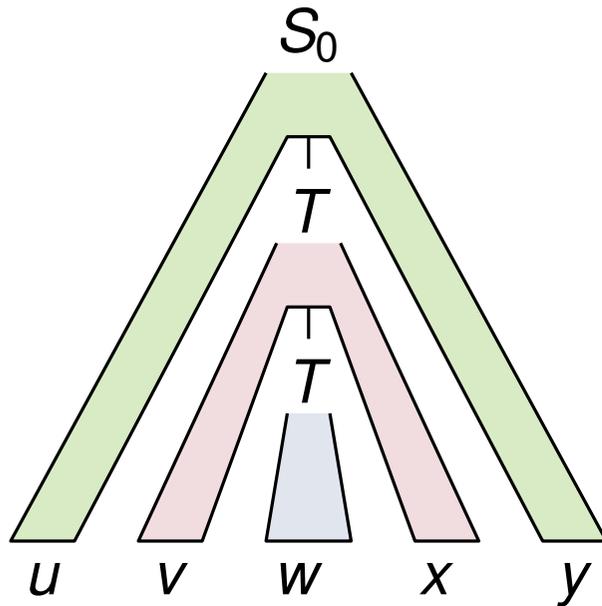
- u das Teilwort von w , das vor \mathcal{Z} ,
- v das Teilwort von w , das in \mathcal{Z} , und
- x das Teilwort von w , das nach \mathcal{Z}

abgearbeitet wird.

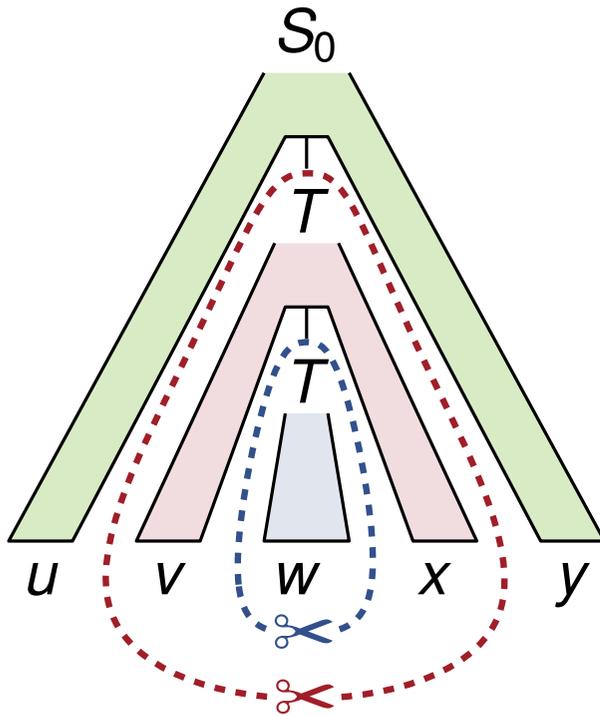
→ $uv^i x$ (mit $i \in \mathbb{N}_0$) ist auch in L enthalten.
Durchlaufe \mathcal{Z} entsprechend häufig.



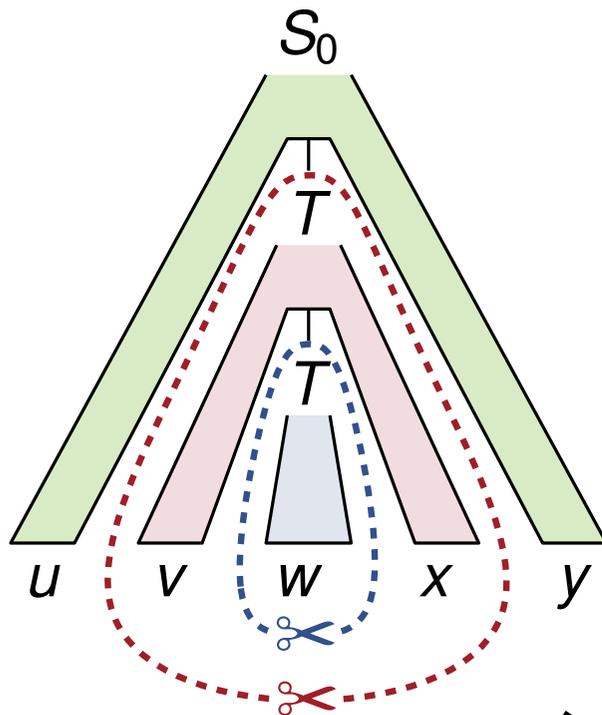
Pumpinglemma für kontextfreie Sprachen



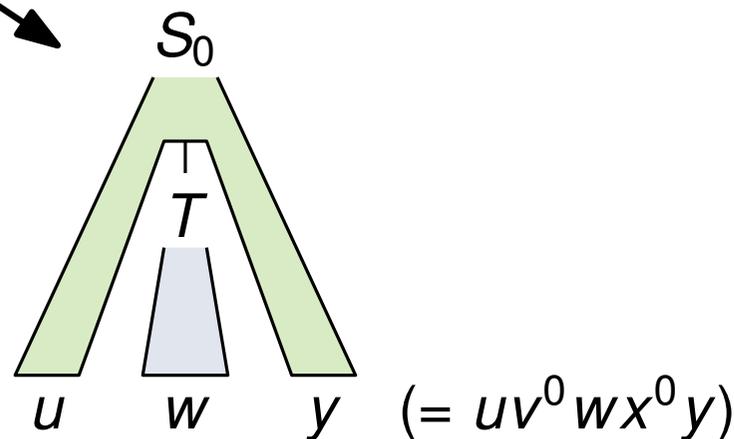
Pumpinglemma für kontextfreie Sprachen



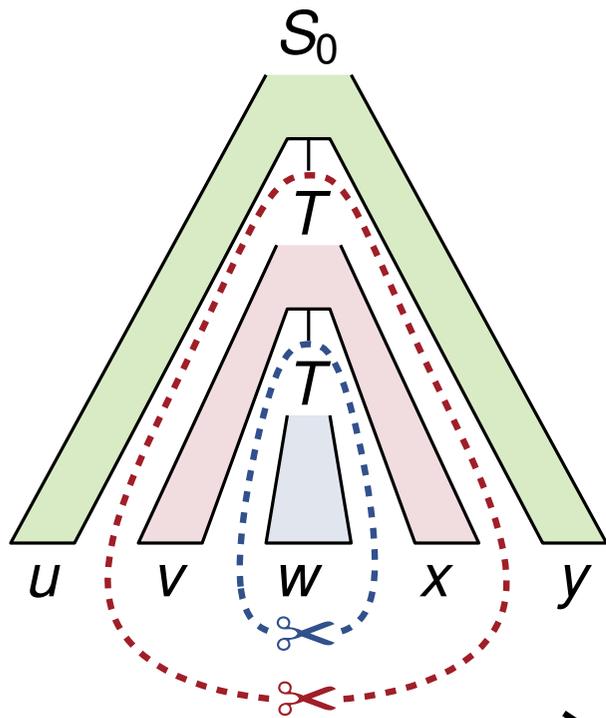
Pumpinglemma für kontextfreie Sprachen



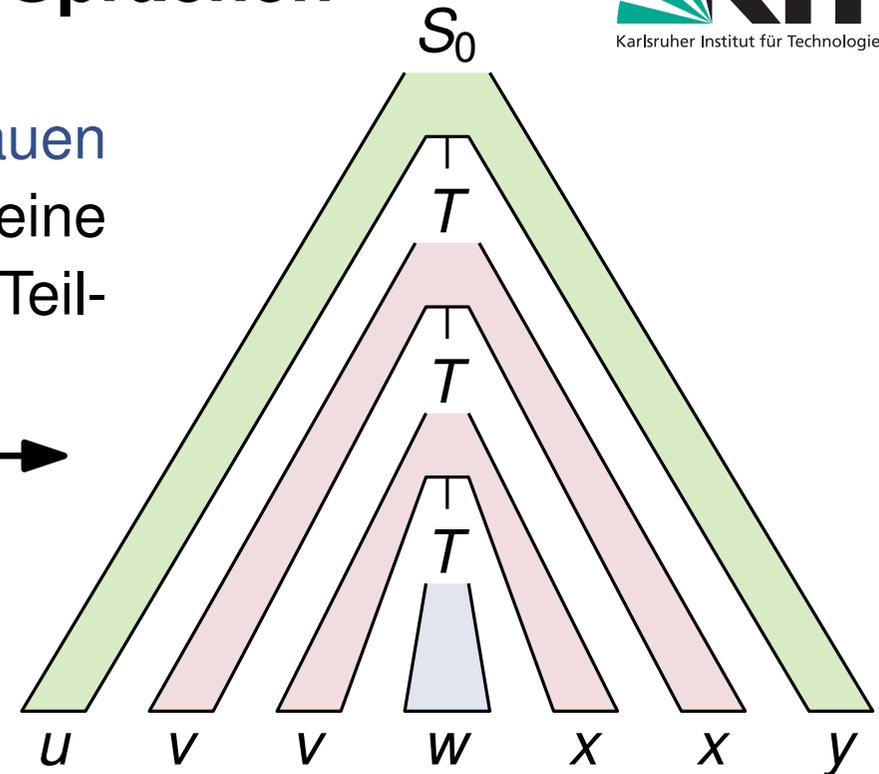
ersetze den roten Teilbaum durch den blauen Teilbaum



Pumpinglemma für kontextfreie Sprachen

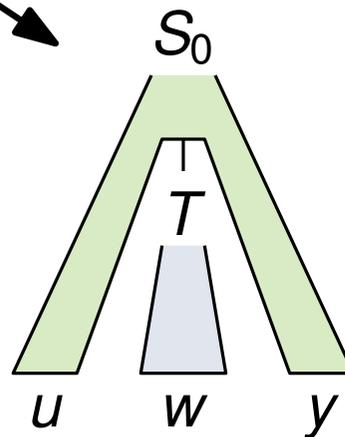


ersetze den **blauen** Teilbaum durch eine Kopie des **roten** Teilbaums



$$(= uv^2wx^2y)$$

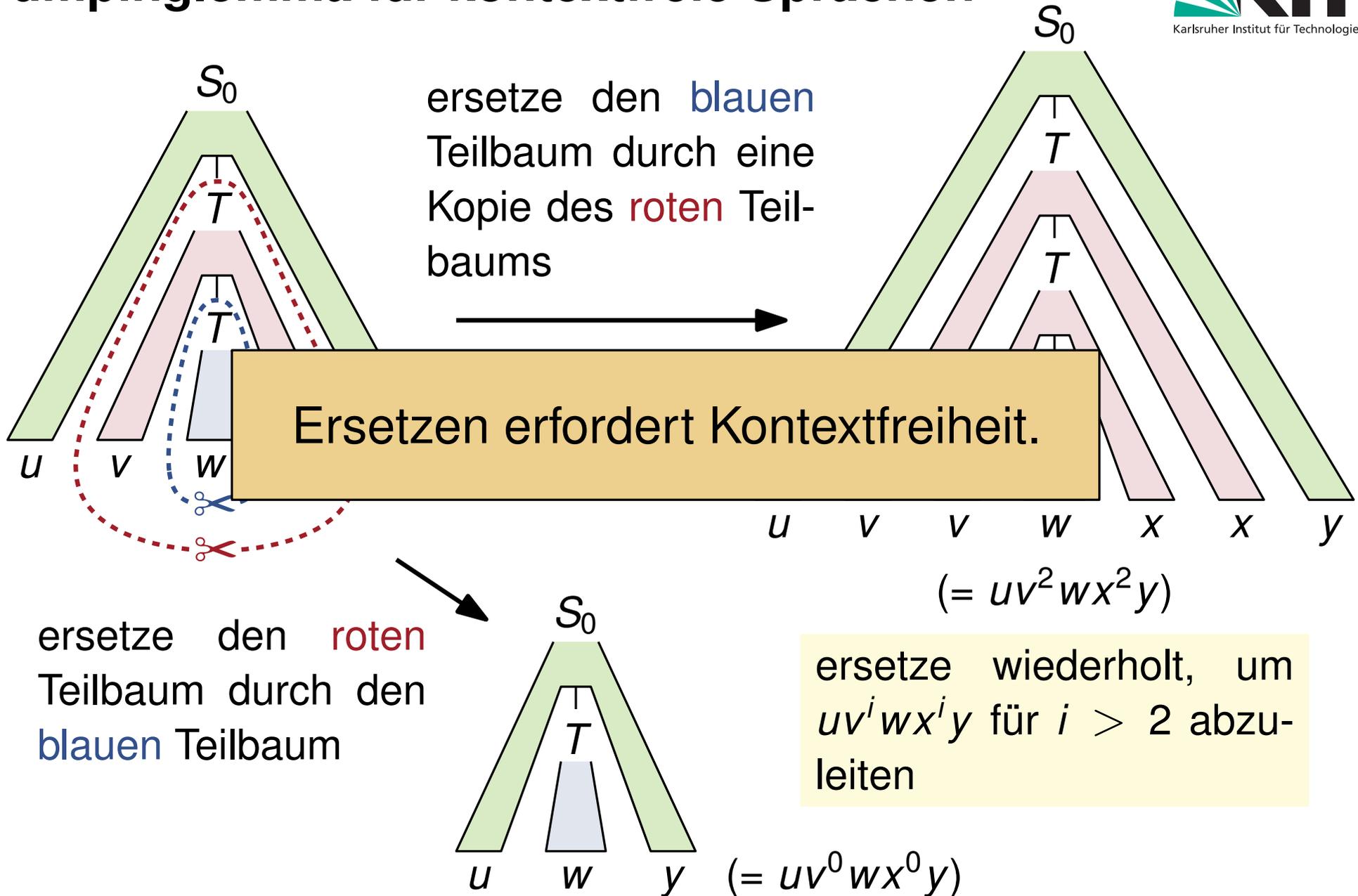
ersetze den **roten** Teilbaum durch den **blauen** Teilbaum



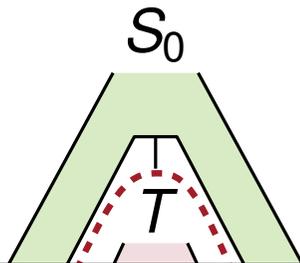
$$(= uv^0wx^0y)$$

ersetze wiederholt, um uv^iwx^iy für $i > 2$ abzuleiten

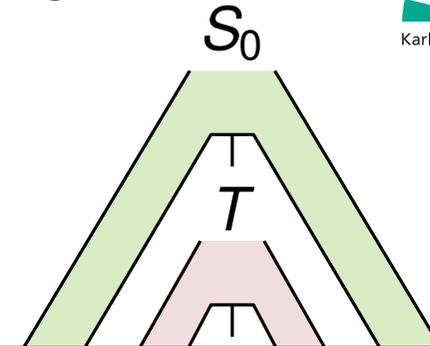
Pumpinglemma für kontextfreie Sprachen



Pumpinglemma für kontextfreie Sprachen



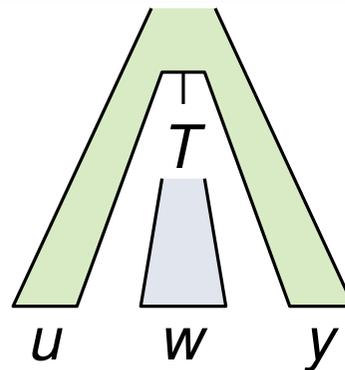
ersetze den **blauen** Teilbaum durch eine Kopie des **roten** Teil-



Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ so in $z = uvwxy$ zerlegen lässt, dass gilt:

- $|vx| \geq 1$,
- $|vwx| \leq n$ und
- $uv^iwx^iy \in L$ für alle $i \geq 0$.

ersetze den **roten** Teilbaum durch den **blauen** Teilbaum



$$u w y \quad (= uv^0wx^0y)$$

ersetze wiederholt, um uv^iwx^iy für $i > 2$ abzuleiten

Kontextfreiheit widerlegen $L_1 = \{a^k ba^{2k} ba^{3k} \in \{a, b\}^* \mid k \geq 0\}$

Betrachte Zerlegung $a^n ba^{2n} ba^{3n} = uvwxy$

Fall 1: v oder x enthält ein b

Fall 2: vwx enthält kein b

Fall 3: vwx überspannt ein b .

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen $L_1 = \{a^k ba^{2k} ba^{3k} \in \{a, b\}^* \mid k \geq 0\}$

Betrachte Zerlegung $a^n ba^{2n} ba^{3n} = uvwxy$

Fall 1: v oder x enthält ein b

Betrachte $w' = uv^2wx^2y$

→ wegen $|vx| \geq 1$ muss w' mehr als zwei b 's enthalten.


 $uv^2wx^2y \in L$

Fall 2: vwx enthält kein b

Fall 3: vwx überspannt ein b .

so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen $L_1 = \{a^k ba^{2k} ba^{3k} \in \{a, b\}^* \mid k \geq 0\}$

Betrachte Zerlegung $a^n ba^{2n} ba^{3n} = uvwxy$

Fall 1: v oder x enthält ein b

Fall 2: vwx enthält kein b

Betrachte das Wort $w' = uv^0wx^0y$

Fall 2.1: vwx tritt vor dem ersten b auf.

→ wegen $|vx| \geq 1$ gilt $w' = a^m b a^{2n} b a^{3n}$ mit $m < n$  $w' \in L_1$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Fall 3: vwx überspannt ein b .

Kontextfreiheit widerlegen $L_1 = \{a^k ba^{2k} ba^{3k} \in \{a, b\}^* \mid k \geq 0\}$

Betrachte Zerlegung $a^n ba^{2n} ba^{3n} = uvwxy$

Fall 1: v oder x enthält ein b

Fall 2: vwx enthält kein b

Betrachte das Wort $w' = uv^0wx^0y$

Fall 2.1: vwx tritt vor dem ersten b auf.

→ wegen $|vx| \geq 1$ gilt $w' = a^m b a^{2n} b a^{3n}$ mit $m < n$ ⚡ $w' \in L_1$

Fall 2.2: vwx liegt zwischen den b 's.

→ wegen $|vx| \geq 1$ gilt $w' = a^n b a^m b a^{3n}$ mit $m < 2n$ ⚡ $w' \in L_1$

Fall 3: vwx überspannt ein b .

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen $L_1 = \{a^k ba^{2k} ba^{3k} \in \{a, b\}^* \mid k \geq 0\}$

Betrachte Zerlegung $a^n ba^{2n} ba^{3n} = uvwxy$

Fall 1: v oder x enthält ein b

Fall 2: vwx enthält kein b

Betrachte das Wort $w' = uv^0wx^0y$

Fall 2.1: vwx tritt vor dem ersten b auf.

→ wegen $|vx| \geq 1$ gilt $w' = a^m b a^{2n} b a^{3n}$ mit $m < n$ ⚡ $w' \in L_1$

Fall 2.2: vwx liegt zwischen den b 's.

→ wegen $|vx| \geq 1$ gilt $w' = a^n b a^m b a^{3n}$ mit $m < 2n$ ⚡ $w' \in L_1$

Fall 2.3: vwx liegt hinter dem zweiten b .

→ wegen $|vx| \geq 1$ gilt
 $w' = a^n b a^{2n} b a^m$ mit $m < 3n$ ⚡

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$,
so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Fall 3: vwx überspannt ein b .

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v\}$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Betrachte das Wort $w' = uv^2wx^2y = w_1cw_2$ für $w_1, w_2 \in \{a, b\}^*$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

- $z = uvwxy$

zerlegen lässt und

- $|vx| \geq 1$,
- $|vwx| \leq n$ und
- $uv^iwx^iy \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v\}$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Betrachte $w' = uv^2wx^2y \in L_2$

→ wegen $|vx| \geq 1$ muss w' mehr als ein c enthalten.



Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

- $z = uvwxy$

zerlegen lässt und

- $|vx| \geq 1$,
- $|vwx| \leq n$ und
- $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Betrachte das Wort $w' = uv^2wx^2y = w_1cw_2$ für $w_1, w_2 \in \{a, b\}^*$

→ wegen $|vx| \geq 1$ gilt $|w_1| > |w_2|$

→ w_1 ist nicht Teilwort von w_2

 $w' \in L_2$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^iy \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v\}$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Betrachte das Wort $w' = uv^0wx^0y = w_1cw_2$ für $w_1, w_2 \in \{a, b\}^*$

→ wegen $|vx| \geq 1$ gilt $|w_1| > |w_2|$

→ w_1 ist nicht Teilwort von w_2  $w' \in L_2$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Fall 4: w überspannt c

Fall 4.1: x beginnt mit einem a

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

- $z = uvwxy$

zerlegen lässt und

- $|vx| \geq 1$,

- $|vwx| \leq n$ und

- $uv^i wx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v\}$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Fall 4: w überspannt c

Fall 4.1: x beginnt mit einem a

→ Betrachte das Wort $w' = uv^0wx^0y = w_1cw_2$ für $w_1, w_2 \in \{a, b\}^*$

→ w_1 enthält ein a , während w_2 nur aus b 's besteht.

→ w_1 ist kein Teilwort von w_2 ⚡ $w' \in L_2$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^iy \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Fall 4: w überspannt c

Fall 4.1: x beginnt mit einem a

Fall 4.2: x beginnt mit einem b

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

- $z = uvwxy$

zerlegen lässt und

- $|vx| \geq 1$,

- $|vwx| \leq n$ und

- $uv^i wx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Fall 4: w überspannt c

Fall 4.1: x beginnt mit einem a

Fall 4.2: x beginnt mit einem b (Wegen $|vwx| \leq n$ nicht möglich)

Fall 4.3: $x = \varepsilon$

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

- $z = uvwxy$

zerlegen lässt und

- $|vx| \geq 1$,

- $|vwx| \leq n$ und

- $uv^i wx^i y \in L$ für alle $i \geq 0$.

Kontextfreiheit widerlegen

$$L_2 = \{wcv \in \{a, b, c\}^* \mid w \text{ ist ein Teilwort von } v\}$$

Betrachte Zerlegung $a^n b^n c a^n b^n = uvwxy$ mit $v \in \{a, b\}^*$

Fall 1: v oder x enthält ein c

Fall 2: vwx tritt vor c auf

Fall 3: vwx tritt nach c auf

Fall 4: w überspannt c

Fall 4.1: x beginnt mit einem a

Fall 4.2: x beginnt mit einem b (Wegen $|vwx| \leq n$ nicht möglich)

Fall 4.3: $x = \varepsilon$

Betrachte das Wort $w' = uv^2wx^2y = w_1cw_2$ für $w_1, w_2 \in \{a, b\}^*$

→ wegen $|vx| \geq 1$ gilt $|w_1| > |w_2|$

→ w_1 ist nicht Teilwort von w_2

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ in

■ $z = uvwxy$

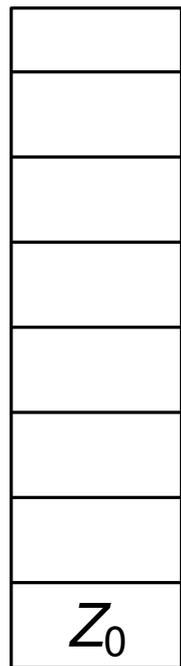
zerlegen lässt und

■ $|vx| \geq 1$,

■ $|vwx| \leq n$ und

■ $uv^iwx^i y \in L$ für alle $i \geq 0$.

Kellerautomaten

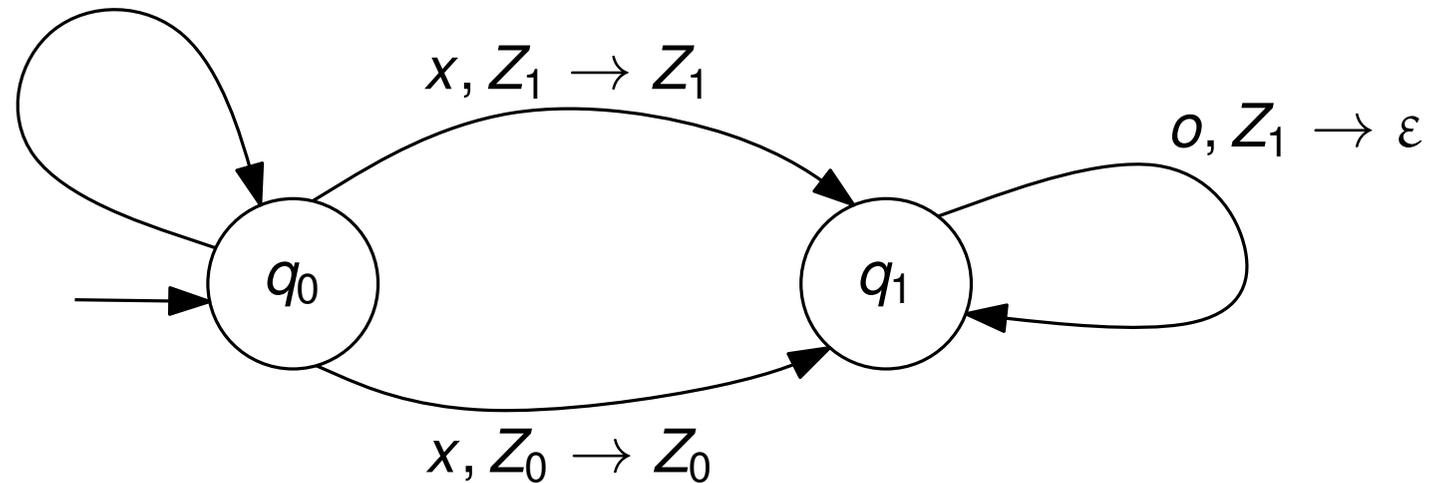


Stack

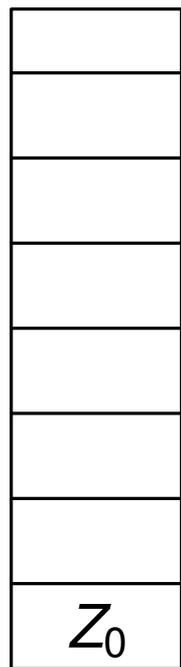
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Ein (nichtdet.) *Kellerautomat* $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$ besteht aus

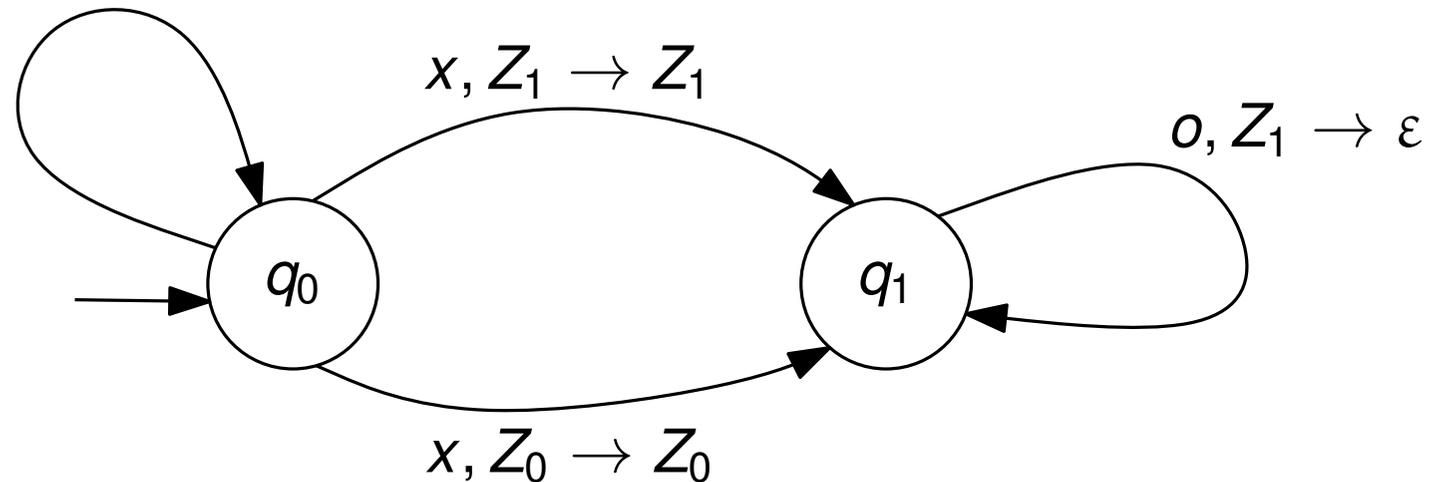


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

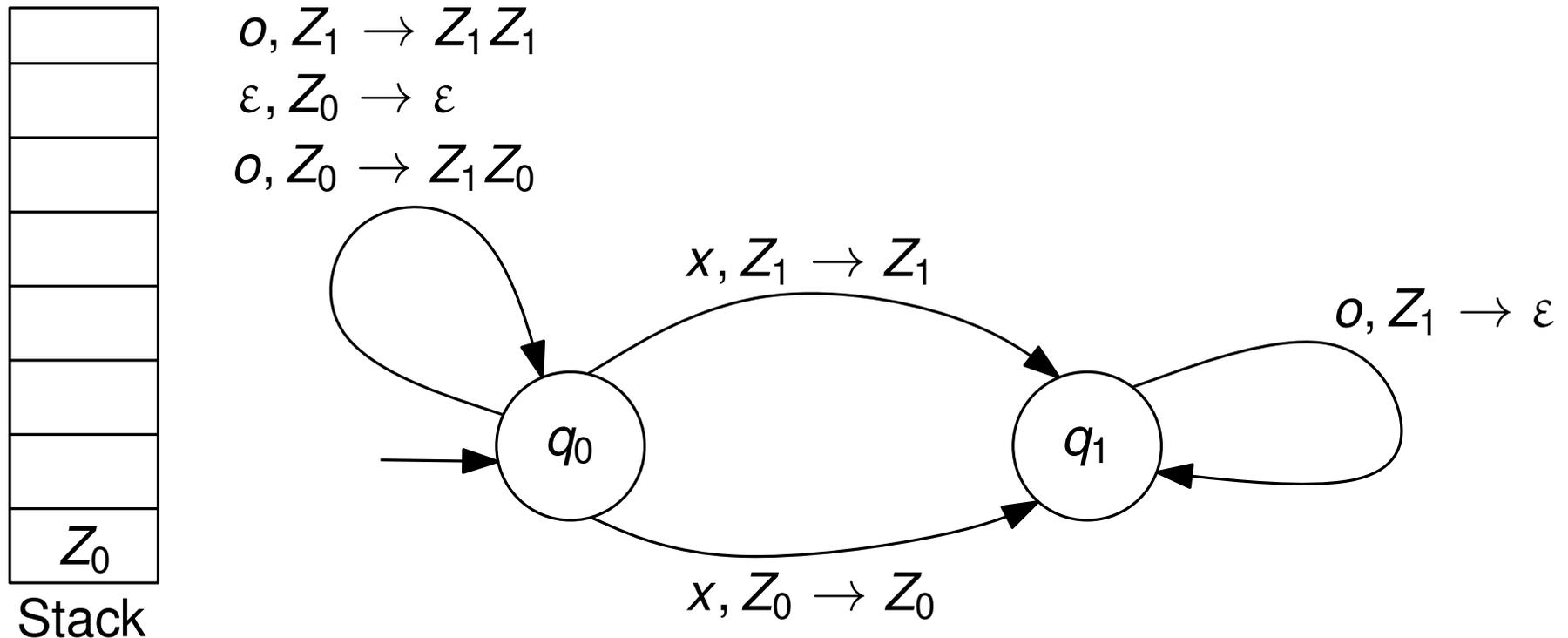
$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Ein (nichtdet.) *Kellerautomat* $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$ besteht aus

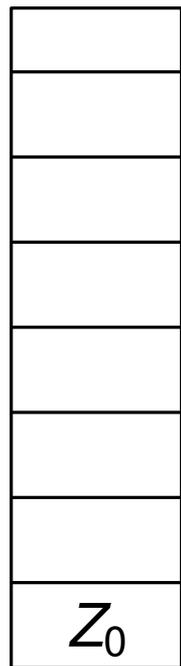
- Q endliche Zustandsmenge und Anfangszustand $q_0 \in Q$
- F Menge akzeptierender Zustände
- Σ endliches Eingabealphabet
- Γ endliches Stackalphabet und Stackinitialisierung $Z_0 \in \Gamma$



Ein (nichtdet.) *Kellerautomat* $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$ besteht aus

- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$
 - $\delta(q, a, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$
 - $\delta(q, \varepsilon, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$

Abarbeitung eines Wortes

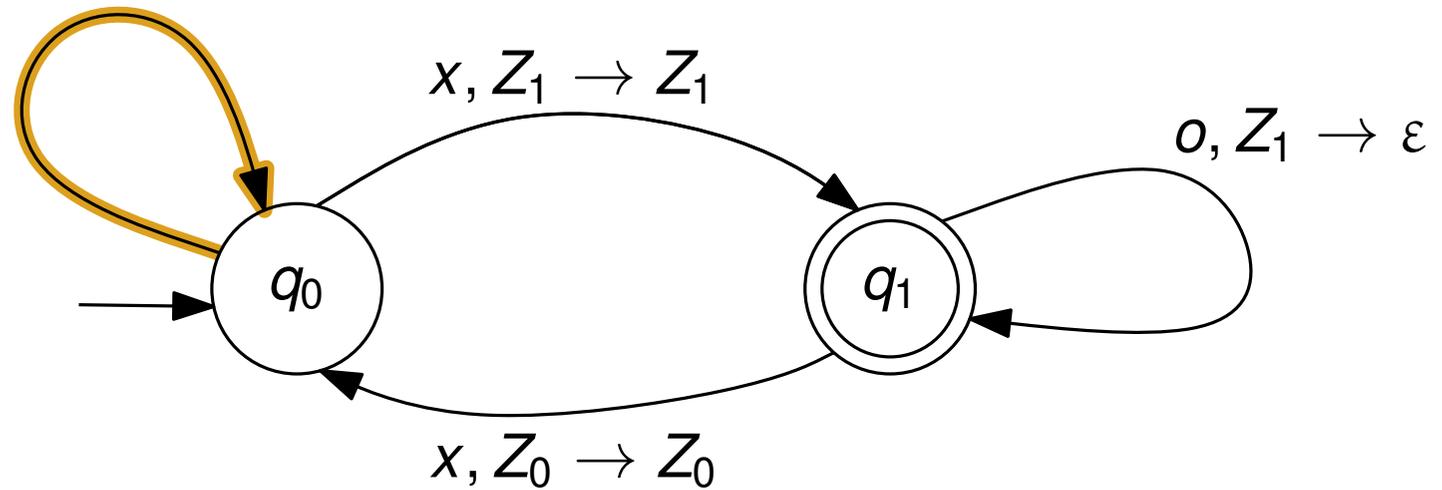


Stack

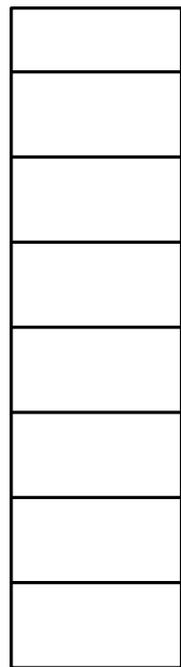
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

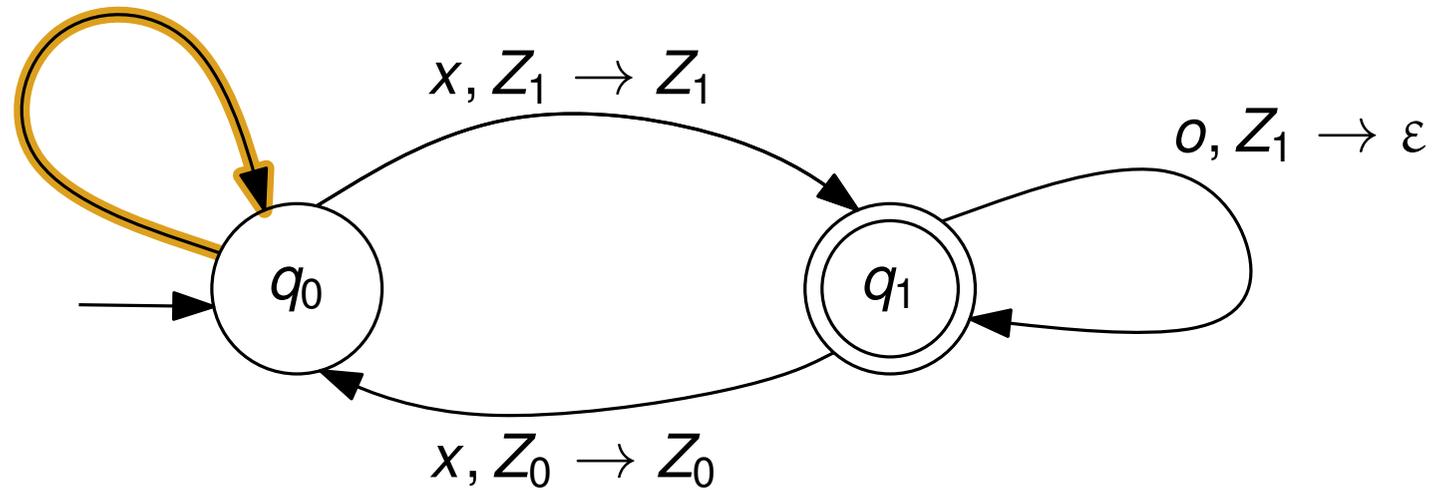


Stack

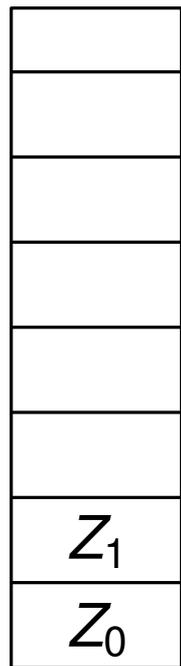
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

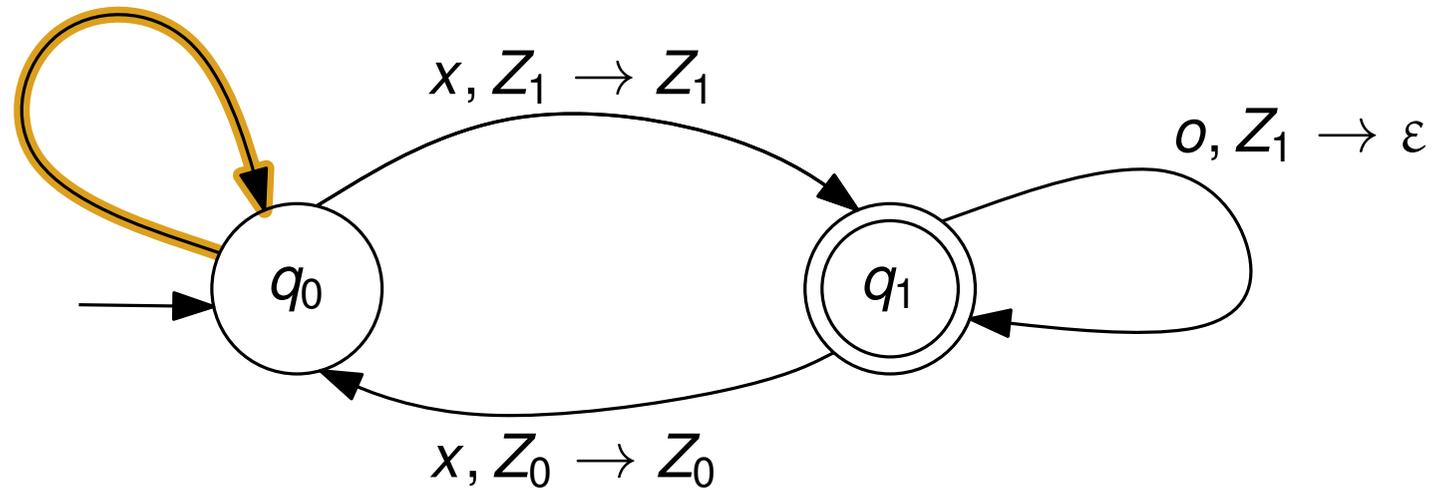


Stack

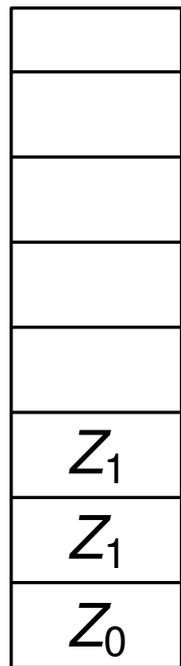
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

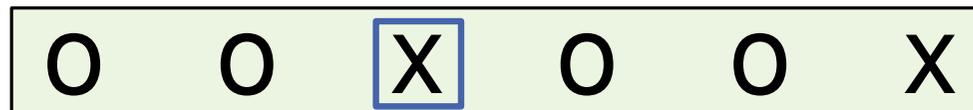
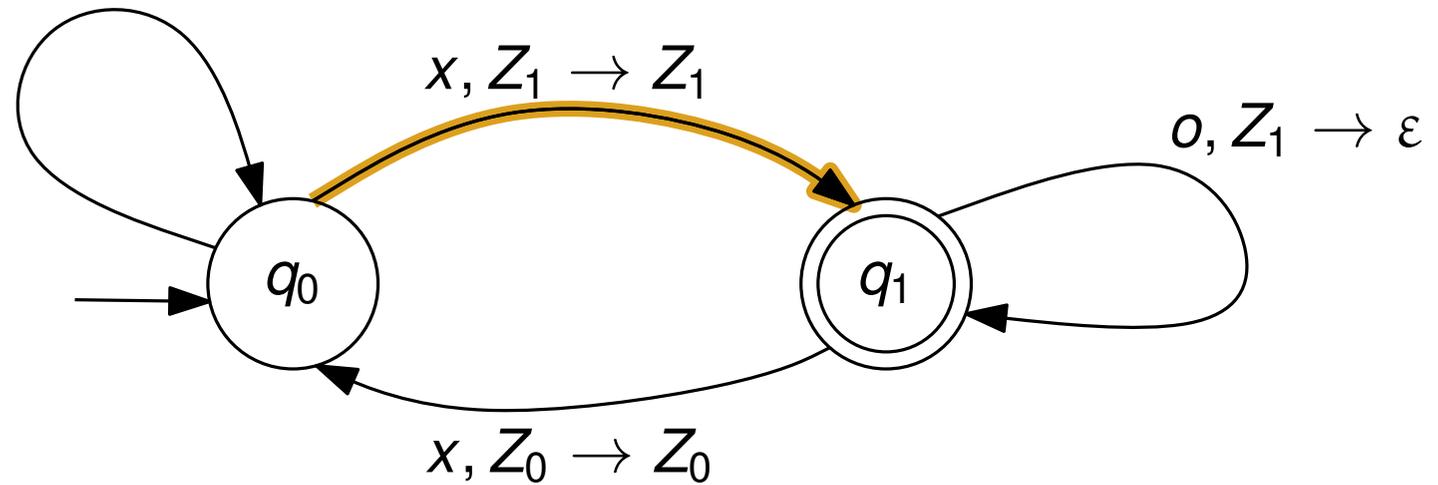


Stack

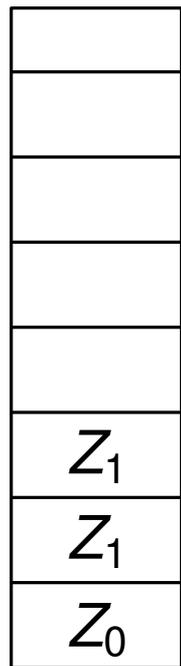
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

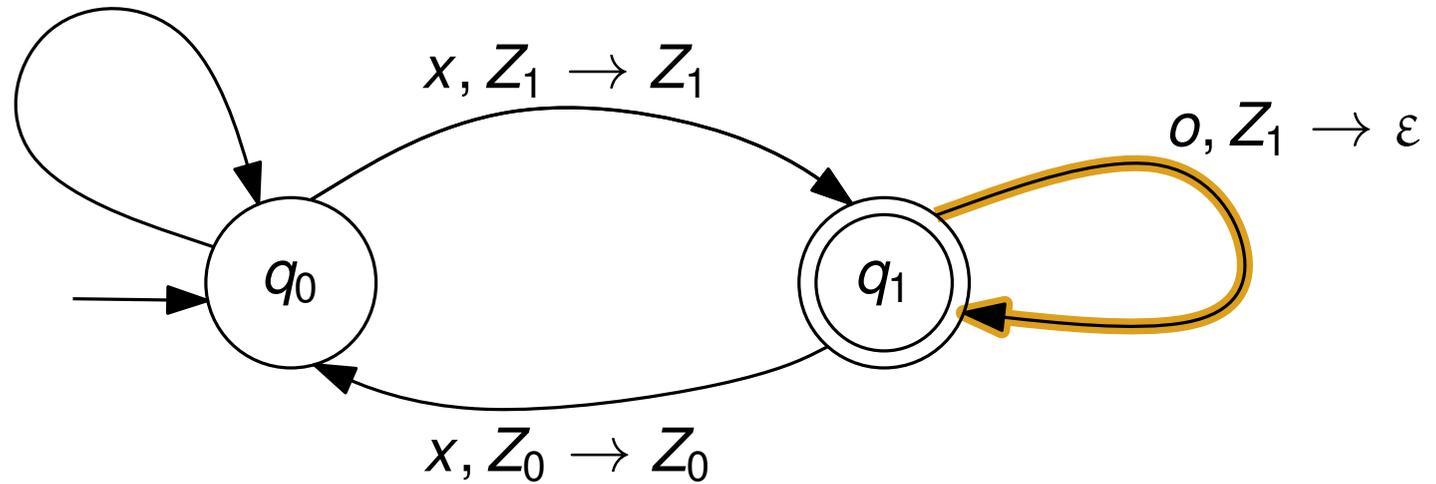


Stack

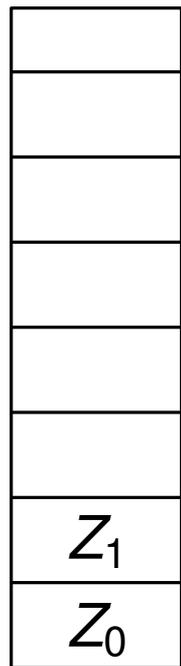
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

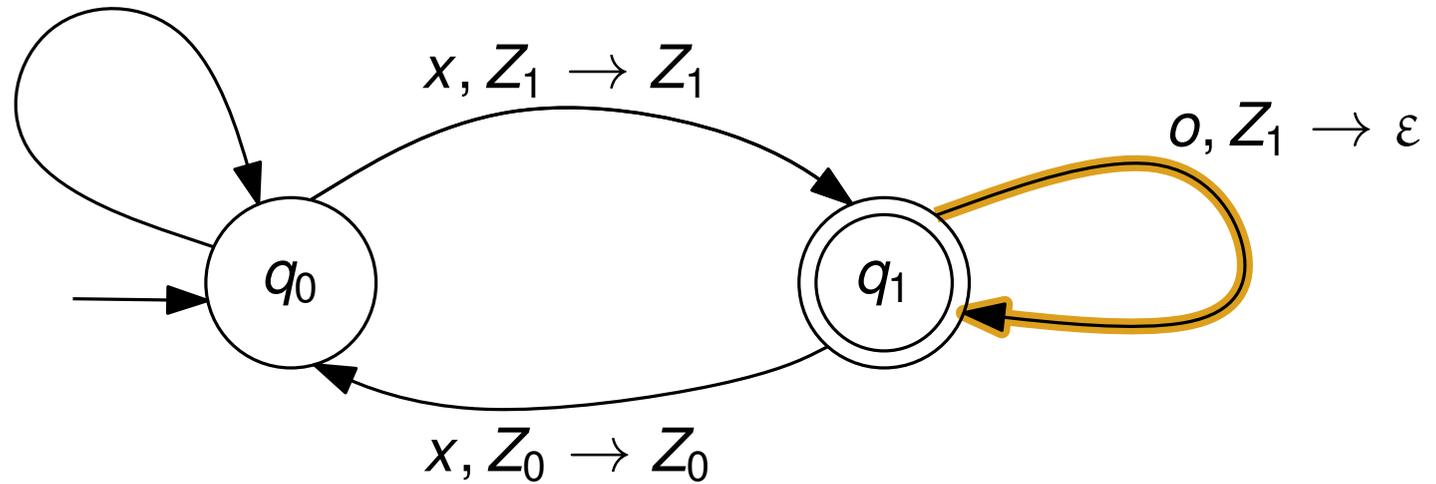


Stack

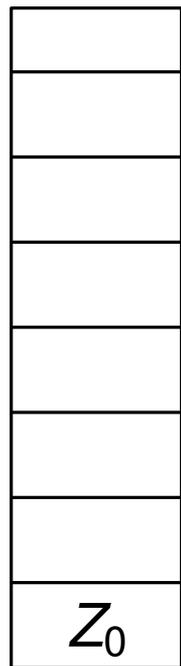
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

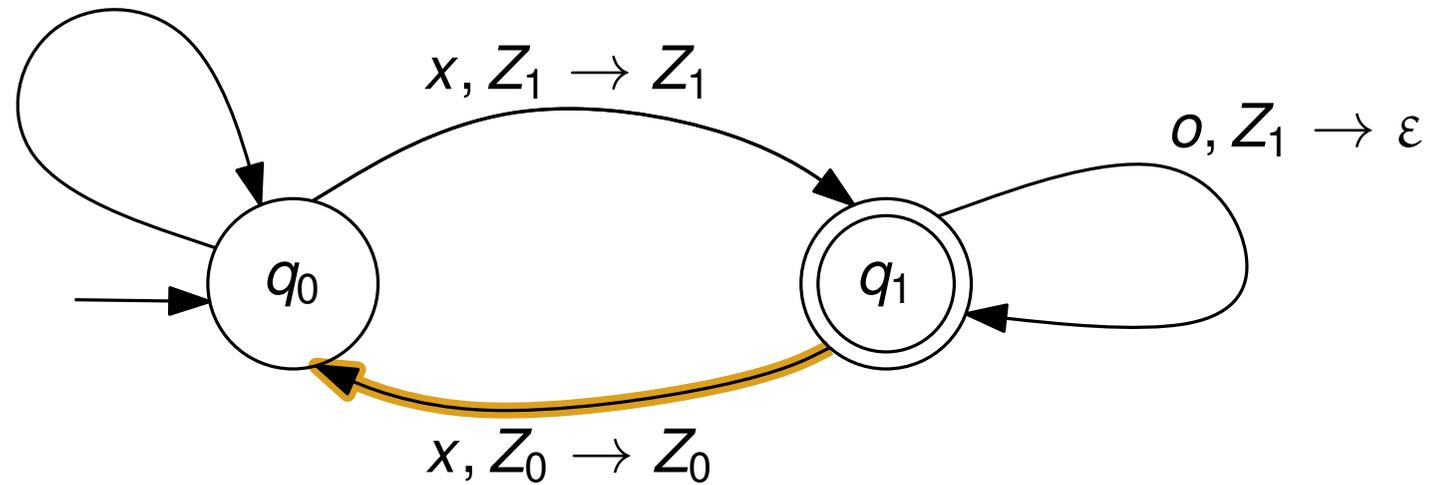


Stack

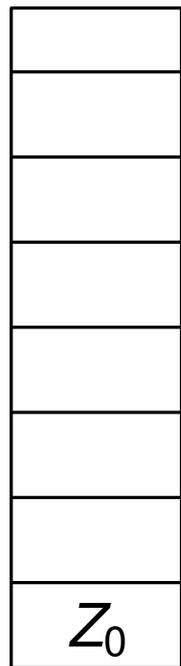
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



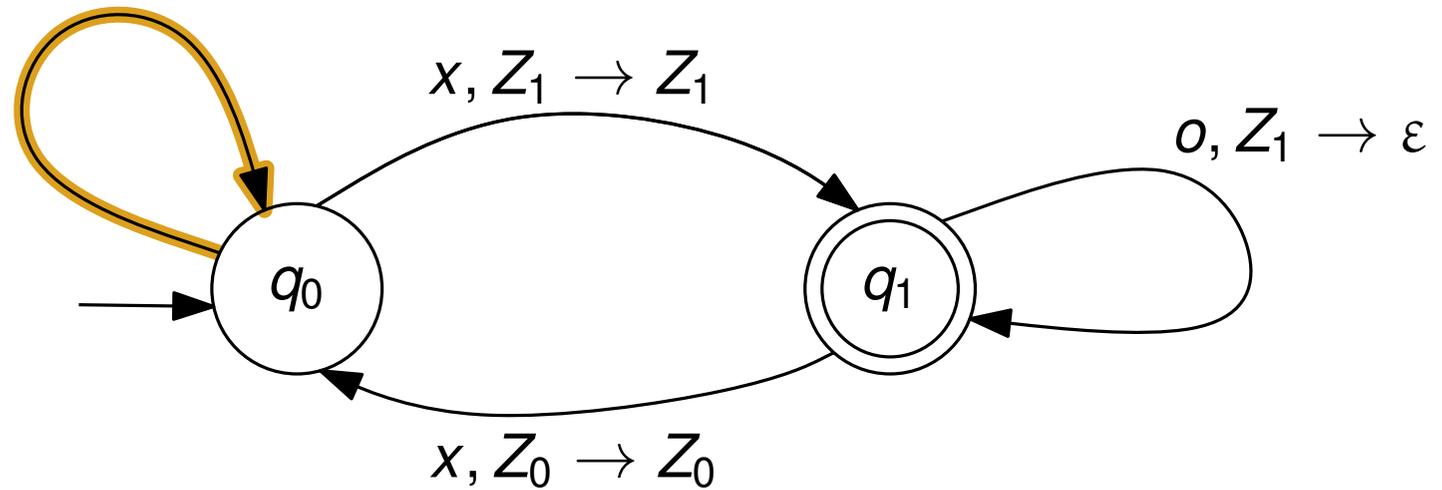
Abarbeitung eines Wortes



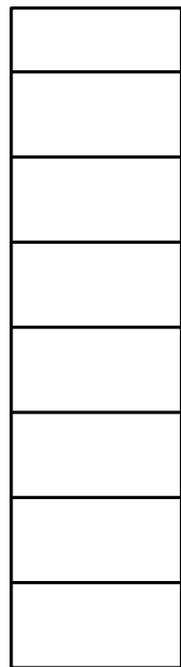
$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



Abarbeitung eines Wortes

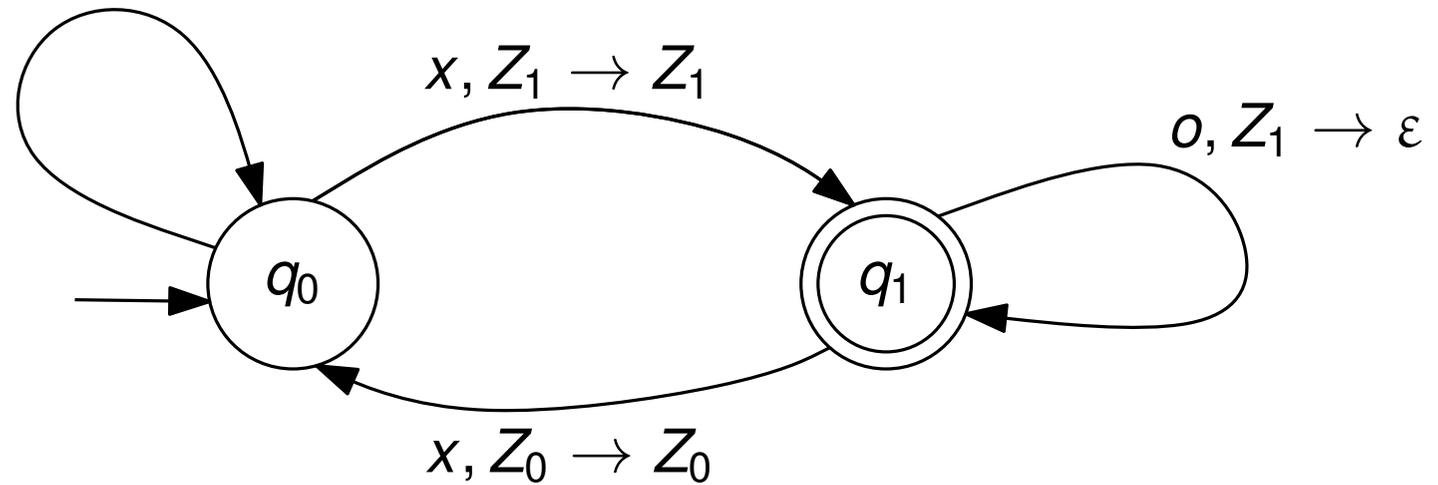


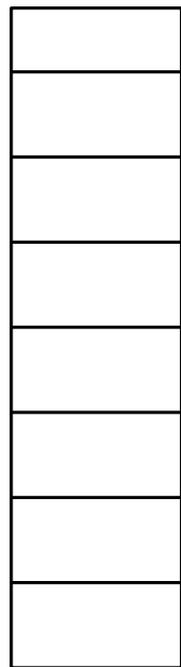
Stack

$$0, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$0, Z_0 \rightarrow Z_1 Z_0$$



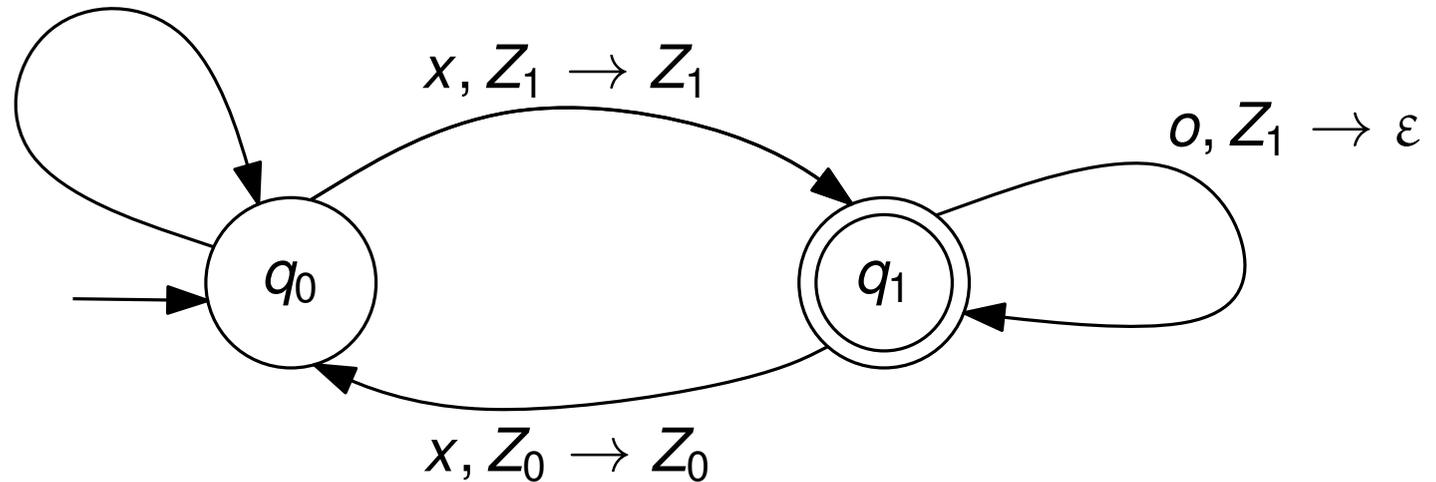


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$

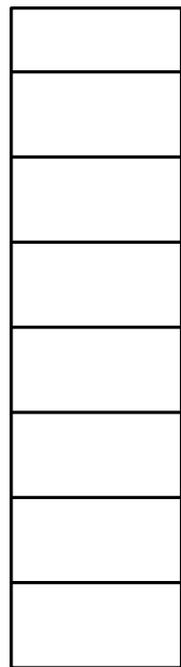


Ein Kellerautomat ist *deterministisch*, falls

$$|\delta(q, a, Z)| + |\delta(q, \varepsilon, Z)| \leq 1$$

für alle $q \in Q, a \in \Sigma, Z \in \Gamma$ gilt.

Determinismus

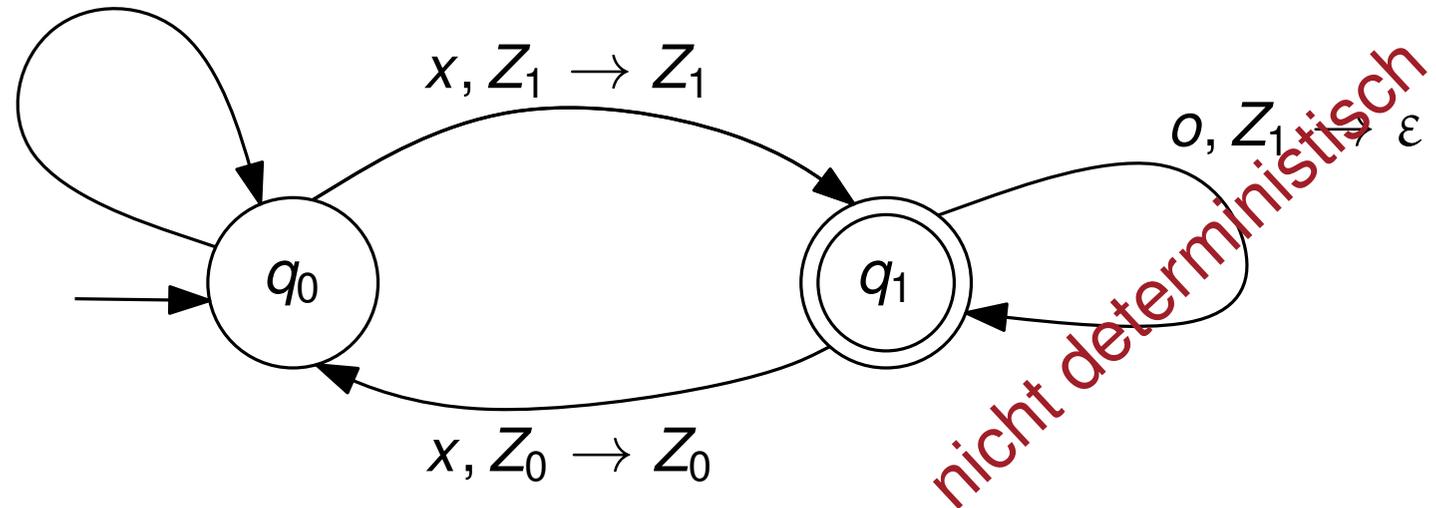


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Ein Kellerautomat ist *deterministisch*, falls

$$|\delta(q, a, Z)| + |\delta(q, \varepsilon, Z)| \leq 1$$

für alle $q \in Q, a \in \Sigma, Z \in \Gamma$ gilt.

Akzeptanz eines Wortes

Sei $w \in \Sigma^*$ das Eingabewort.

Akzeptanz durch leeren Stack

Es gibt zulässige Folge von Konfigurationen aus der Anfangskonfiguration (q_0, w, Z_0) in eine Konfiguration $(q, \varepsilon, \varepsilon)$, $q \in Q$.

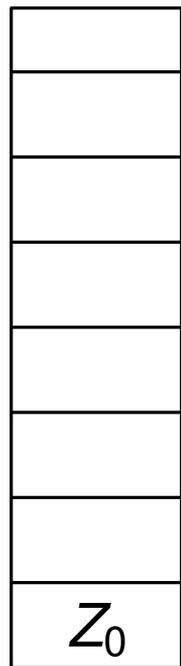
Akzeptanz durch Endzustände

Es gibt zulässige Folge von Konfigurationen aus der Anfangskonfiguration (q_0, w, Z_0) in eine Konfiguration (q, ε, γ) mit $q \in F$ und $\gamma \in \Gamma^*$ gibt.

Konfiguration (q, w, α) :

- q = Zustand,
- w = der Teil der Eingabe, der noch nicht gelesen wurde,
- α = Stackinhalt.

Akzeptanz eines Wortes

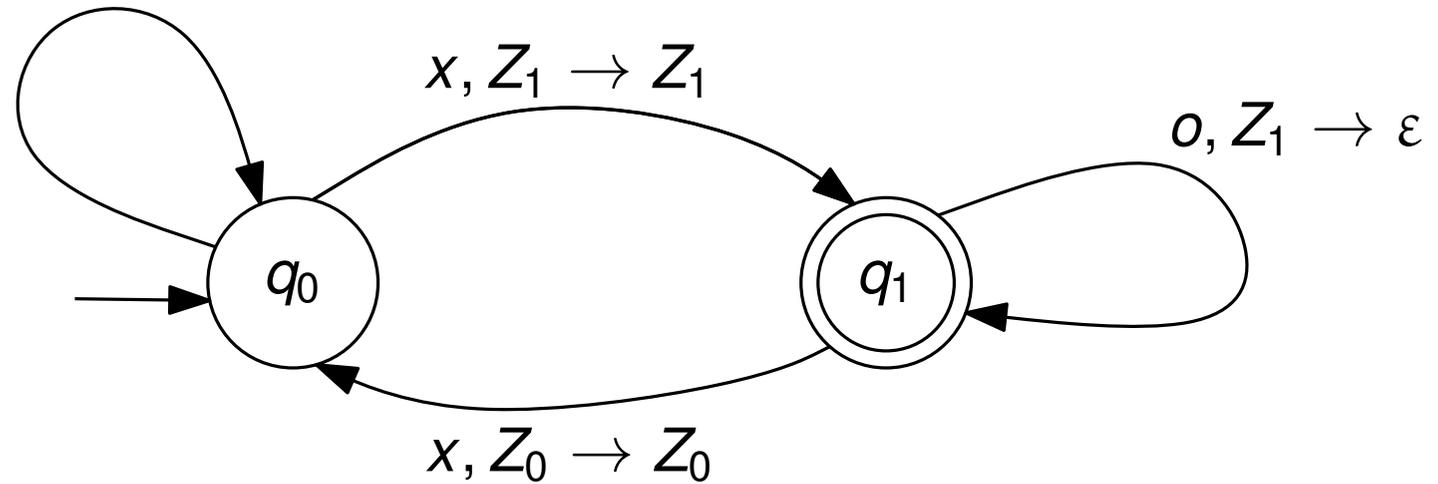


Stack

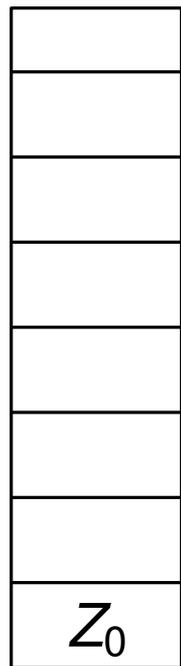
$$o, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Akzeptanz eines Wortes

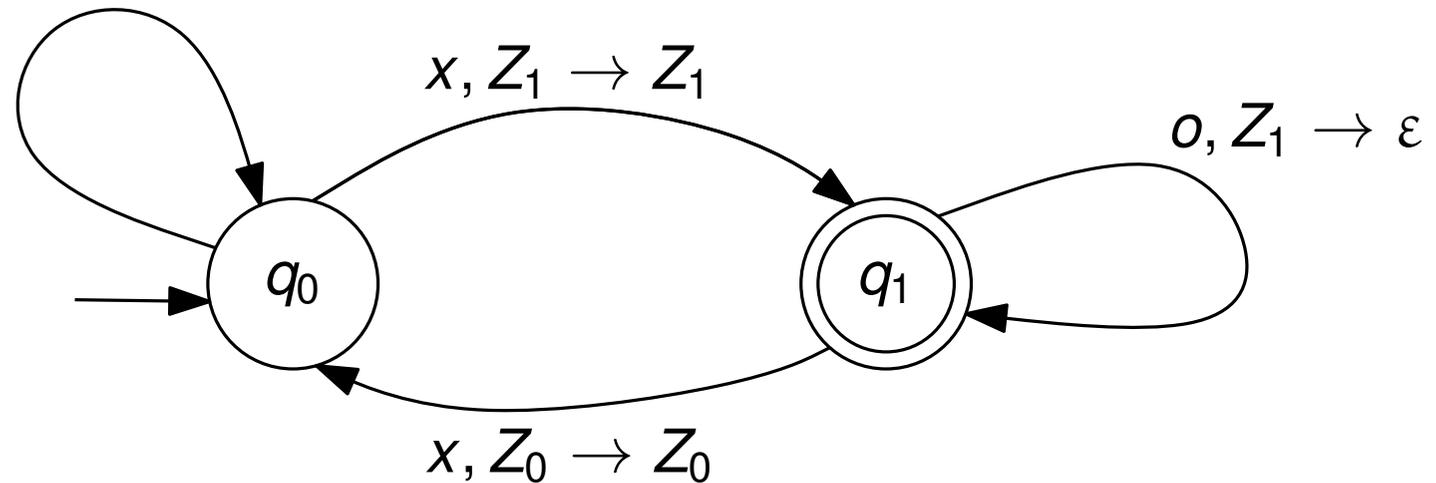


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

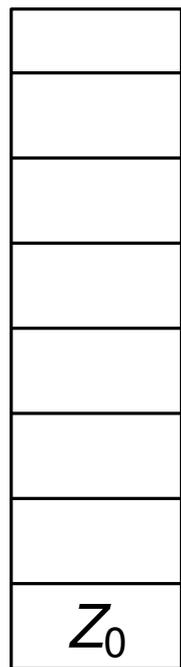
$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Gesucht: Sprache L_F , die \mathcal{A} mit Endzuständen akzeptiert.

Akzeptanz eines Wortes

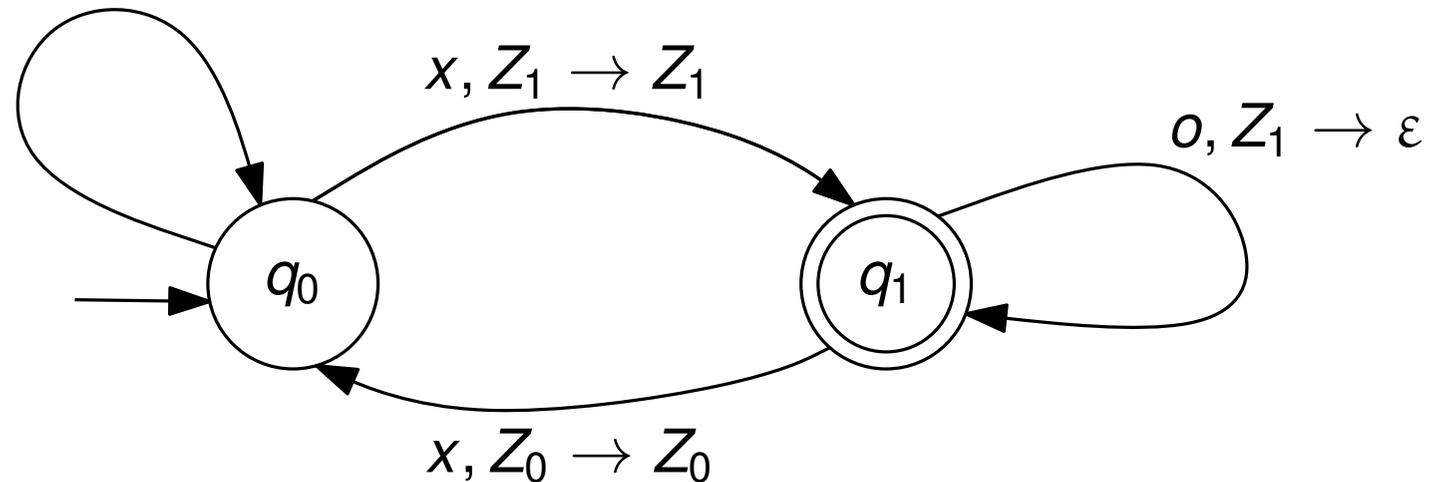


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

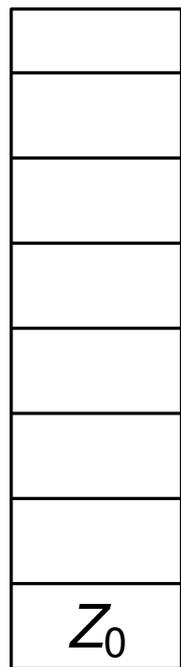
$$o, Z_0 \rightarrow Z_1 Z_0$$



Gesucht: Sprache L_F , die \mathcal{A} mit Endzuständen akzeptiert.

$$L_F = \{ o^{i_1} x o^{i_1} x o^{i_2} x o^{i_2} x \dots o^{i_k} x o^{i_k} x o^j x o^j \mid i_1, i_2, \dots, i_k, i \geq 1, i \geq j \}$$

Akzeptanz eines Wortes

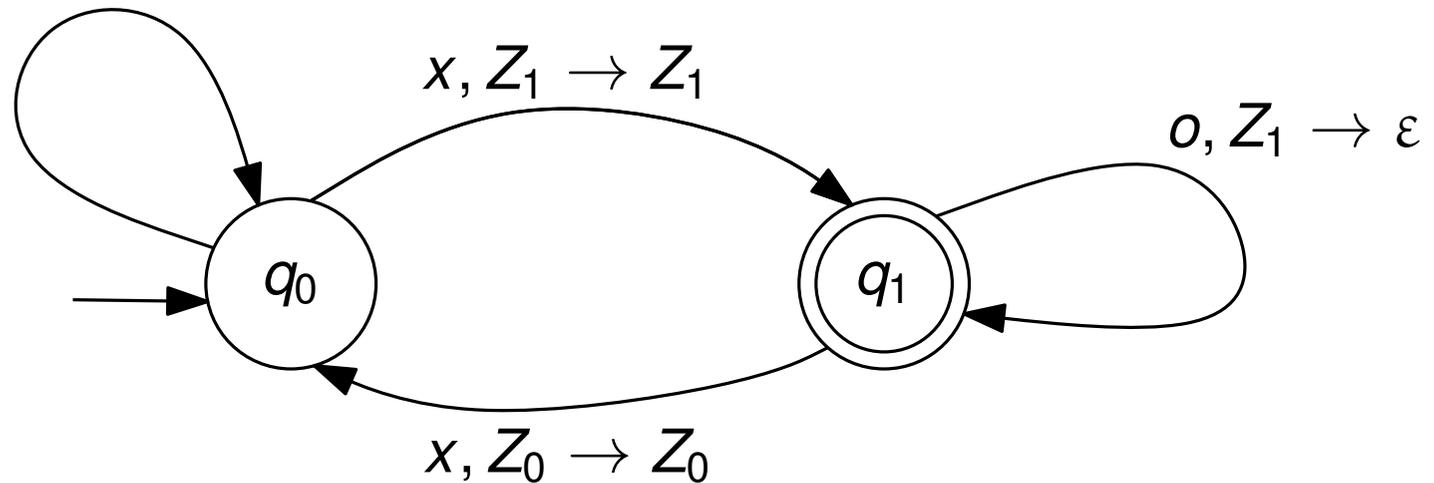


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

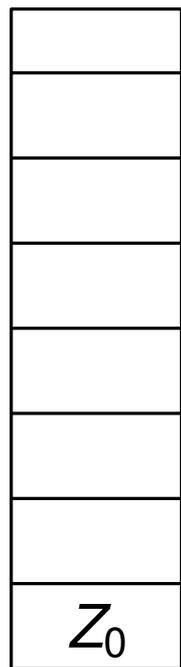
$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Gesucht: Sprache L_ε , die \mathcal{A} durch leeren Stack akzeptiert.

Akzeptanz eines Wortes

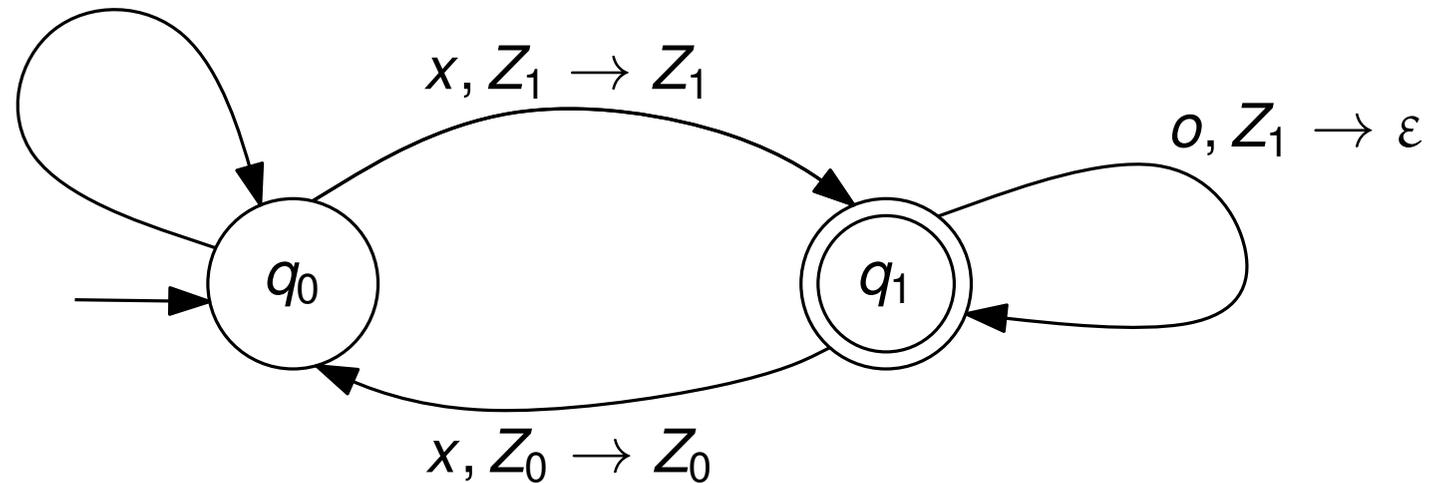


Stack

$$o, Z_1 \rightarrow Z_1 Z_1$$

$$\varepsilon, Z_0 \rightarrow \varepsilon$$

$$o, Z_0 \rightarrow Z_1 Z_0$$



Gesucht: Sprache L_ε , die \mathcal{A} durch leeren Stack akzeptiert.

Sei $L = \{o^i x o^i x \mid i > 0\}$, dann ist $L_\varepsilon = L^*$

Grammatik von L_ε

Sei $L = \{o^i x o^i x \mid i > 0\}$, dann ist $L_\varepsilon = L^*$

Die kontextfreie Grammatik $G = (\{o, x\}, \{S, B\}, S, R)$ für L_ε mit

$$R = \{S \rightarrow \varepsilon \mid SS \mid oBox,$$
$$B \rightarrow oBo \mid x\}$$

Eine kontextfreie Grammatik ist in *Greibach-Normalform*, wenn alle Ableitungsregeln von der Form

$$A \rightarrow a\alpha \text{ mit } A \in V, a \in \Sigma \text{ und } \alpha \in V^*$$

sind.

Für jede kontextfreie Grammatik G , für die $L(G)$ das leere Wort nicht enthält, kann eine (äquivalente) kontextfreie Grammatik G' mit $L(G) = L(G')$ in Greibach-Normalform konstruiert werden.

Greibach-Normalform

Ersetzung (i). Eine Regel

$$A \rightarrow \alpha_1 B \alpha_2$$

wobei

$$B \rightarrow \beta_1, \quad B \rightarrow \beta_2, \quad \dots, \quad B \rightarrow \beta_r$$

alle Regeln sind, deren linke Seite B ist, kann durch die Regeln

$$A \rightarrow \alpha_1 \beta_1 \alpha_2$$

$$A \rightarrow \alpha_1 \beta_2 \alpha_2$$

...

$$A \rightarrow \alpha_1 \beta_r \alpha_2$$

ersetzt werden.

Ersetzung (ii). Seien

$$A \rightarrow A\alpha_1, \dots, A \rightarrow A\alpha_r$$

$$A \rightarrow \beta_1, \dots, A \rightarrow \beta_s$$

alle Regeln, deren linke Seite A ist, wobei β_i nicht mit A beginnen. Dann können die Regeln

$$A \rightarrow A\alpha_1, \dots, A \rightarrow A\alpha_r$$

durch die Regeln

$$A \rightarrow \beta_1 B, \dots, A \rightarrow \beta_s B$$

$$B \rightarrow \alpha_1, \dots, B \rightarrow \alpha_r,$$

$$B \rightarrow \alpha_1 B, \dots, B \rightarrow \alpha_r B$$

ersetzt werden. Dabei sei B eine neu eingeführte Variable.

Annahme: G befindet sich in Chomsky-Normalform

1. Schritt: Falls $A_i \rightarrow A_j \alpha$ Regel ist, so gilt $j > i$.

2. Schritt: Bringe Regeln mit linker Seite in V in Form

$$A_k \rightarrow a\alpha, a \in \Sigma, \alpha \in (V')^*$$

3. Schritt: Ersetze Regeln $B_i \rightarrow A_j \alpha, \alpha \in (\Sigma \cup V')^*$ mit Ersetzung (i).

Konstruktion von Greibach-Normalform

$G = (\Sigma, V, S, R)$ mit $\Sigma = \{a, b, c\}$ und $V = \{S, B, C\}$

$S \rightarrow BC \mid a$

$B \rightarrow SC \mid c$

$C \rightarrow BS \mid b$

Konstruktion von Greibach-Normalform

$G = (\Sigma, V, S, R)$ mit $\Sigma = \{a, b, c\}$ und $V = \{S, B, C\}$

$$S \rightarrow BC \mid a$$

$$B \rightarrow SC \mid c$$

$$C \rightarrow BS \mid b$$

0. Schritt: Nichtterminale umbenennen.

$$A_1 \rightarrow A_2 A_3 \mid a$$

$$A_2 \rightarrow A_1 A_3 \mid c$$

$$A_3 \rightarrow A_2 A_1 \mid b$$

Konstruktion von Greibach-Normalform

1. Schritt: Falls $A_i \rightarrow A_j \alpha$ Regel ist, so gilt $j > i$.

$$A_1 \rightarrow A_2 A_3 \mid a$$

$$A_2 \rightarrow A_1 A_3 \mid c$$

$$A_3 \rightarrow A_2 A_1 \mid b$$

Konstruktion von Greibach-Normalform

1. Schritt: Falls $A_i \rightarrow A_j \alpha$ Regel ist, so gilt $j > i$.

$$A_1 \rightarrow A_2 A_3 \mid a$$

$$A_2 \rightarrow A_1 A_3 \mid c$$

$$A_3 \rightarrow A_2 A_1 \mid b$$

Ersetzung (i)
auf A_2 

$$A_1 \rightarrow A_2 A_3 \mid a$$

$$A_2 \rightarrow A_2 A_3 A_3 \mid a A_3 \mid c$$

$$A_3 \rightarrow A_2 A_1 \mid b$$

Konstruktion von Greibach-Normalform

1. Schritt: Falls $A_i \rightarrow A_j \alpha$ Regel ist, so gilt $j > i$.

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow A_1 A_3 \mid c \\ A_3 \rightarrow A_2 A_1 \mid b \end{array}$$

Ersetzung (i)
auf A_2 →

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow A_2 A_3 A_3 \mid a A_3 \mid c \\ A_3 \rightarrow A_2 A_1 \mid b \end{array}$$

Ersetzung (ii)
auf A_2 →

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow a A_3 \mid c \mid a A_3 B_1 \mid c B_1 \\ A_3 \rightarrow A_2 A_1 \mid b \\ B_1 \rightarrow A_3 A_3 \mid A_3 A_3 B_1 \end{array}$$

Konstruktion von Greibach-Normalform

1. Schritt: Falls $A_i \rightarrow A_j \alpha$ Regel ist, so gilt $j > i$.

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow A_1 A_3 \mid c \\ A_3 \rightarrow A_2 A_1 \mid b \end{array}$$

Ersetzung (i)
auf A_2

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow A_2 A_3 A_3 \mid a A_3 \mid c \\ A_3 \rightarrow A_2 A_1 \mid b \end{array}$$

Ersetzung (ii)
auf A_2

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow a A_3 \mid c \mid a A_3 B_1 \mid c B_1 \\ A_3 \rightarrow A_2 A_1 \mid b \\ B_1 \rightarrow A_3 A_3 \mid A_3 A_3 B_1 \end{array}$$

Ersetzung (ii)
auf A_3

$$\begin{array}{l} A_1 \rightarrow A_2 A_3 \mid a \\ A_2 \rightarrow a A_3 \mid c \mid a A_3 B_1 \mid c B_1 \\ A_3 \rightarrow a A_3 A_1 \mid c A_1 \mid a A_3 B_1 A_1 \mid c B_1 A_1 \mid b \\ B_1 \rightarrow A_3 A_3 \mid A_3 A_3 B_1 \end{array}$$

2. Schritt: Bringe Regeln mit linker Seite in V in Form

$$A_k \rightarrow a\alpha, a \in \Sigma, \alpha \in (V')^*$$

$$A_1 \rightarrow A_2A_3 \mid a$$

$$A_2 \rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1$$

$$A_3 \rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b$$

$$B_1 \rightarrow A_3A_3 \mid A_3A_3B_1$$

Konstruktion von Greibach-Normalform

2. Schritt: Bringe Regeln mit linker Seite in V in Form

$$A_k \rightarrow a\alpha, a \in \Sigma, \alpha \in (V')^*$$

$$A_1 \rightarrow A_2A_3 \mid a$$

$$A_2 \rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1$$

$$A_3 \rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b$$

$$B_1 \rightarrow A_3A_3 \mid A_3A_3B_1$$

Ersetzung (i)

auf A_2

$$A_1 \rightarrow aA_3A_3 \mid cA_3 \mid aA_3B_1A_3 \mid cB_1A_3 \mid a$$

$$A_2 \rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1$$

$$A_3 \rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b$$

$$B_1 \rightarrow A_3A_3 \mid A_3A_3B_1$$

Konstruktion von Greibach-Normalform

3. Schritt: Ersetze Regeln $B_i \rightarrow A_j \alpha$, $\alpha \in (\Sigma \cup V')^*$ mit Ersetzung (i).

$$\begin{aligned} A_1 &\rightarrow aA_3A_3 \mid cA_3 \mid aA_3B_1A_3 \mid cB_1A_3 \mid a \\ A_2 &\rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1 \\ A_3 &\rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b \\ B_1 &\rightarrow A_3A_3 \mid A_3A_3B_1 \end{aligned}$$

Ersetzung (i)
auf A_3 

Konstruktion von Greibach-Normalform

3. Schritt: Ersetze Regeln $B_i \rightarrow A_j \alpha$, $\alpha \in (\Sigma \cup V')^*$ mit Ersetzung (i).

$$\begin{aligned}
 A_1 &\rightarrow aA_3A_3 \mid cA_3 \mid aA_3B_1A_3 \mid cB_1A_3 \mid a \\
 A_2 &\rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1 \\
 A_3 &\rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b \\
 B_1 &\rightarrow A_3A_3 \mid A_3A_3B_1
 \end{aligned}$$

Ersetzung (i)

 auf A_3

$$\begin{aligned}
 A_1 &\rightarrow aA_3A_3 \mid cA_3 \mid aA_3B_1A_3 \mid cB_1A_3 \mid a \\
 A_2 &\rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1 \\
 A_3 &\rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b \\
 B_1 &\rightarrow aA_3A_1A_3 \mid cA_1A_3 \mid aA_3B_1A_1A_3 \mid cB_1A_1A_3 \mid \\
 &\quad bA_3 \mid aA_3A_1A_3B_1 \mid cA_1A_3B_1 \mid aA_3B_1A_1A_3B_1 \mid cB_1A_1A_3B_1 \mid bA_3B_1
 \end{aligned}$$

Greibach-Normalform und Kellerautomaten

Sei $G = (\Sigma, V, S, R)$ eine Grammatik in Greibach-Normalform.



Kellerautomat $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$

$$Q := \{q_0\}$$

$$\Gamma := V$$

$$Z_0 := S$$

$$\delta(q_0, a, A) := \{(q_0, \alpha) \mid (A \rightarrow a\alpha) \in R\}$$

$$\delta(q_0, a, A) := \{(q_0, \alpha) \mid (A \rightarrow a\alpha) \in R\}$$

$$\begin{aligned} A_1 &\rightarrow aA_3A_3 \mid cA_3 \mid aA_3B_1A_3 \mid cB_1A_3 \mid a \\ A_2 &\rightarrow aA_3 \mid c \mid aA_3B_1 \mid cB_1 \\ A_3 &\rightarrow aA_3A_1 \mid cA_1 \mid aA_3B_1A_1 \mid cB_1A_1 \mid b \\ B_1 &\rightarrow aA_3A_1A_3 \mid cA_1A_3 \mid aA_3B_1A_1A_3 \mid cB_1A_1A_3 \mid \\ &\quad bA_3 \mid aA_3A_1A_3B_1 \mid cA_1A_3B_1 \mid aA_3B_1A_1A_3B_1 \mid cB_1A_1A_3B_1 \mid bA_3B_1 \end{aligned}$$

Für A_1

$$\delta(q_0, a, A_1) = \{(q_0, A_3A_3), (q_0, A_3B_1A_3), (q_0, \varepsilon)\}$$

$$\delta(q_0, b, A_1) = \{\}$$

$$\delta(q_0, c, A_1) = \{(q_0, A_3), (q_0, B_1A_3)\}$$

Analog für A_1, A_2, A_3, B_1 .

Umgekehrte Polnische Notation

UPN: Schreibweise von arithmetischen Ausdrücken

$$\left(\frac{5}{7 - (1 + 1)} \times 3 \right) - (2 + (1 + 1))$$

wird geschrieben als

$$5 7 1 1 + - / 3 \times 2 1 1 + + -$$

Umgekehrte Polnische Notation

UPN: Schreibweise von arithmetischen Ausdrücken

$$\left(\frac{5}{7 - (1 + 1)} \times 3 \right) - (2 + (1 + 1))$$

wird geschrieben als

$$5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times\ 2\ 1\ 1\ +\ +\ -$$

Umgekehrte Polnische Notation

UPN: Schreibweise von arithmetischen Ausdrücken

$$\left(\frac{5}{7 - (1 + 1)} \times 3 \right) - (2 + (1 + 1))$$

wird geschrieben als

$$5\ 7\ 1\ 1\ +\ -\ /\ 3\ \times\ 2\ 1\ 1\ +\ +\ -$$

Umgekehrte Polnische Notation

UPN: Schreibweise von arithmetischen Ausdrücken

$$\left(\frac{5}{7 - (1 + 1)} \times 3 \right) - (2 + (1 + 1))$$

wird geschrieben als

$$5711 + - / 3 \times 211 + + -$$

Gesucht: kontextfr. Gr., die arithmetische Ausdrücke in UPN erzeugt.

Ausdruck	Zahl	Verknüpfung
$A \rightarrow Z$	$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow AAV$		

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck

Zahl

Verknüpfung

$A \rightarrow Z$

$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$

$V \rightarrow + \mid - \mid \times \mid /$

$A \rightarrow AAV$

Gesucht: äquivalente Grammatik in Greibach-Normalform.

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck

Zahl

Verknüpfung

$$A \rightarrow Z$$

$$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$$

$$V \rightarrow + \mid - \mid \times \mid /$$

$$A \rightarrow AAV$$

Gesucht: äquivalente Grammatik in Greibach-Normalform.

Verknüpfung

$$V \rightarrow + \mid - \mid \times \mid /$$

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck

Zahl

Verknüpfung

$$A \rightarrow Z$$

$$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$$

$$V \rightarrow + \mid - \mid \times \mid /$$

$$A \rightarrow AAV$$

Gesucht: äquivalente Grammatik in Greibach-Normalform.

Verknüpfung

$$V \rightarrow + \mid - \mid \times \mid /$$

Zahl

$$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$$

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck

$$A \rightarrow Z$$

$$A \rightarrow AAV$$

Zahl

$$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$$

Verknüpfung

$$V \rightarrow + \mid - \mid \times \mid /$$

Gesucht: äquivalente Grammatik in Greibach-Normalform.

Ausdruck

$$A \rightarrow 0 \mid 1 \mid \dots \mid 9$$

$$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$$

Verknüpfung

$$V \rightarrow + \mid - \mid \times \mid /$$

Zahl

$$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$$

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck	Zahl	Verknüpfung
$A \rightarrow Z$	$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow AAV$		

Gesucht: äquivalente Grammatik in Greibach-Normalform.

Ausdruck	Rest: „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow AV$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow AVR$	Zahl
		$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$

Umgekehrte Polnische Notation

Kontextfreie Grammatik, die arithmetische Ausdrücke in UPN erzeugt:

Ausdruck	Zahl	Verknüpfung
$A \rightarrow Z$	$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow AAV$		

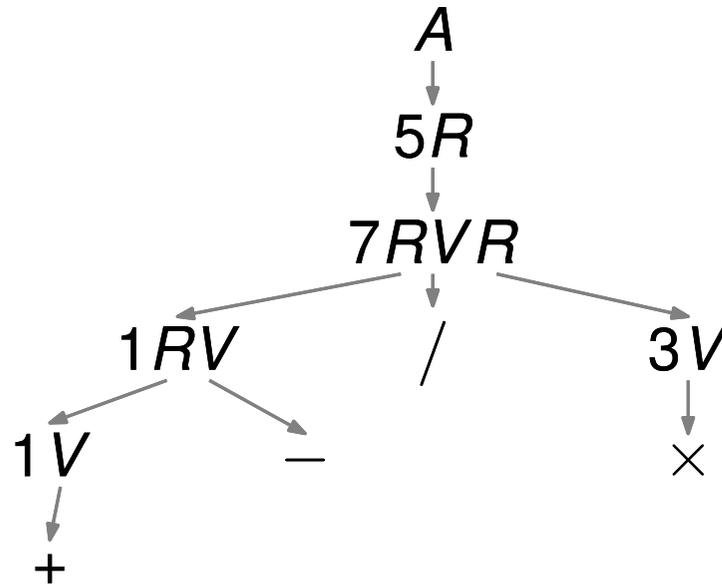
Gesucht: äquivalente Grammatik in Greibach-Normalform.

Ausdruck	Rest: „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	Zahl
	$R \rightarrow 0VR \mid 1VR \mid \dots$	$Z \rightarrow 0 \mid 1 \mid \dots \mid 9$
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: Ableitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

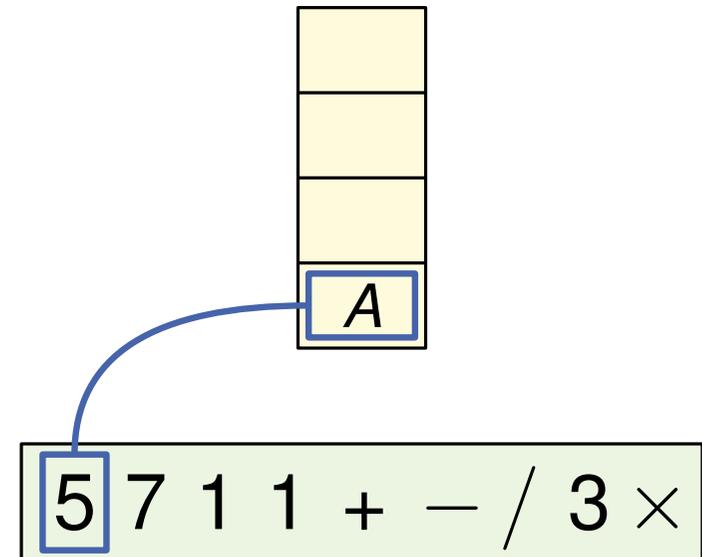
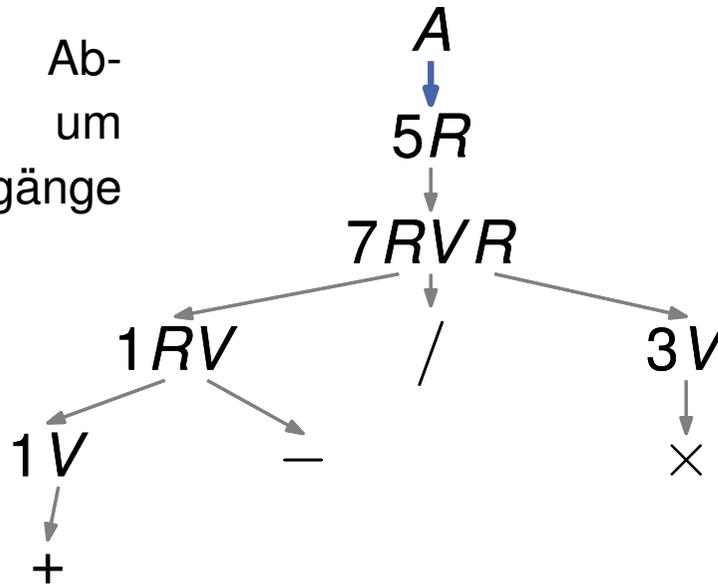


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

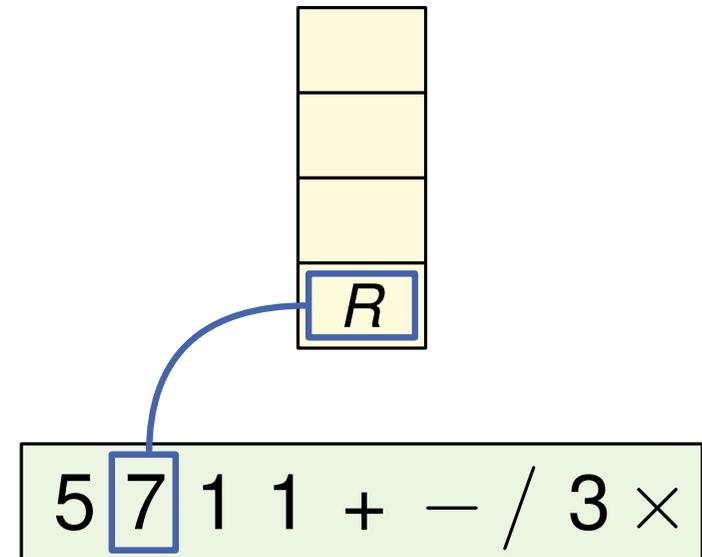
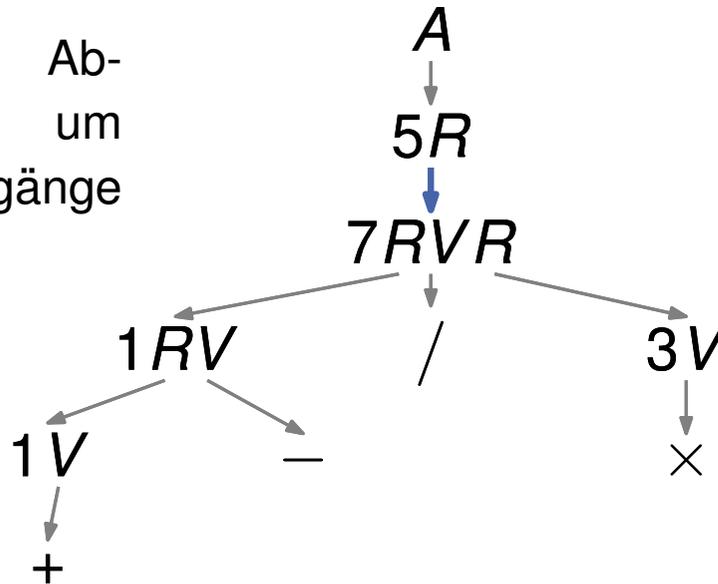


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

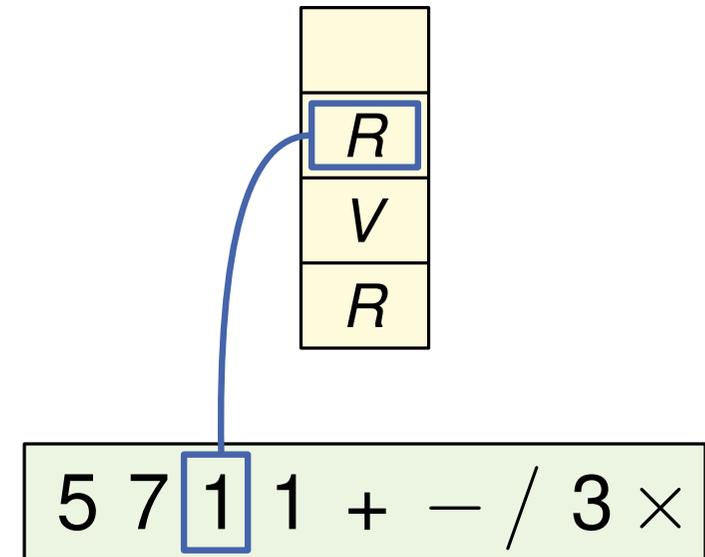
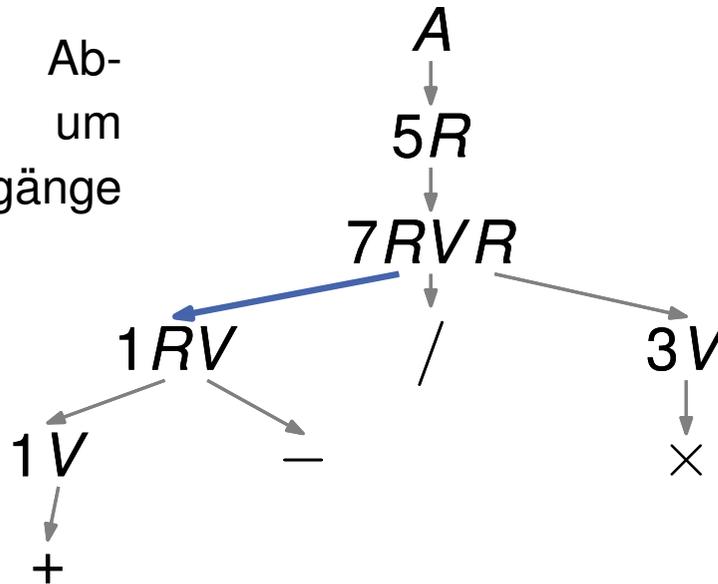


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

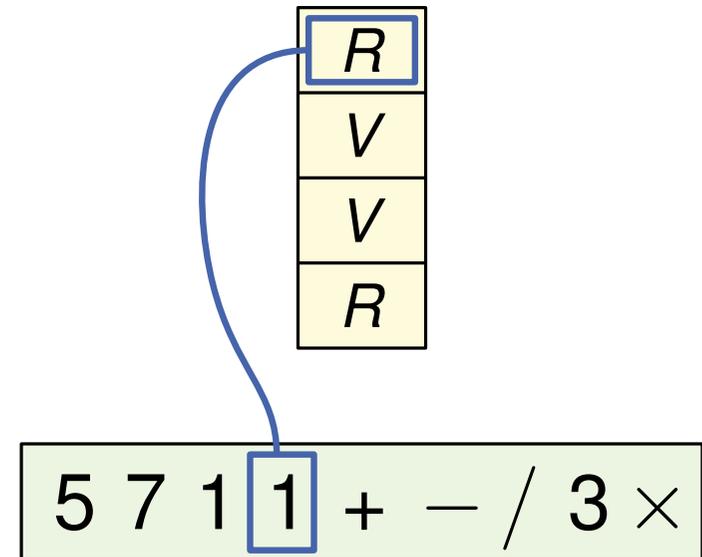
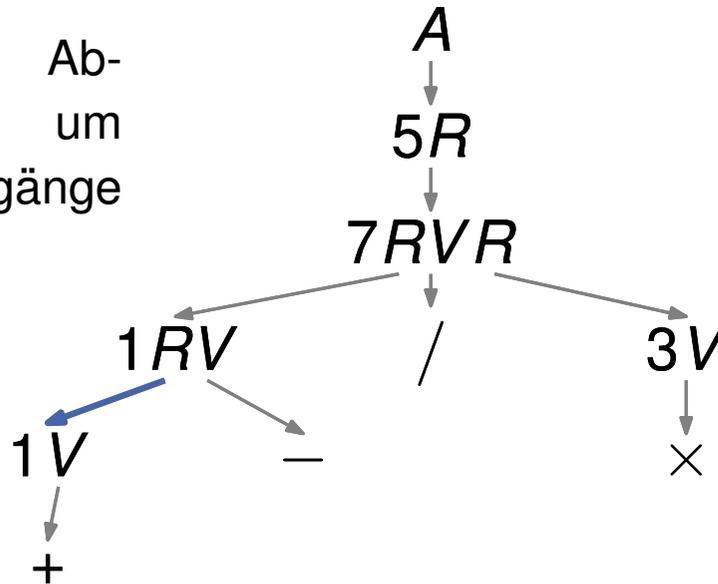


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

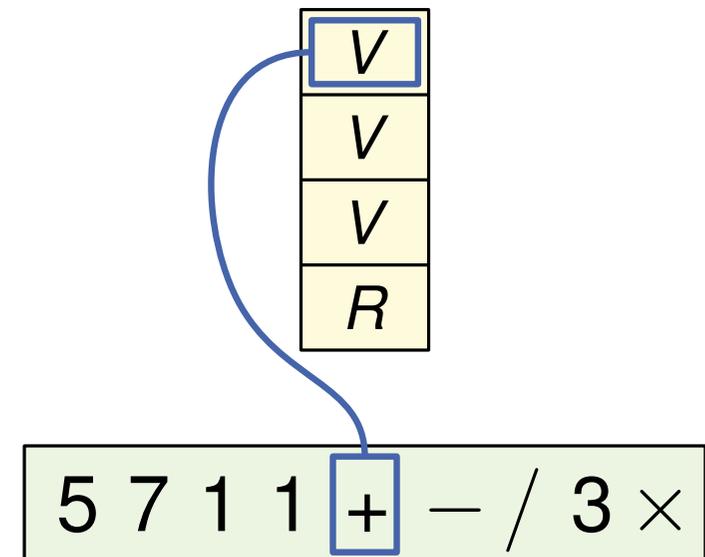
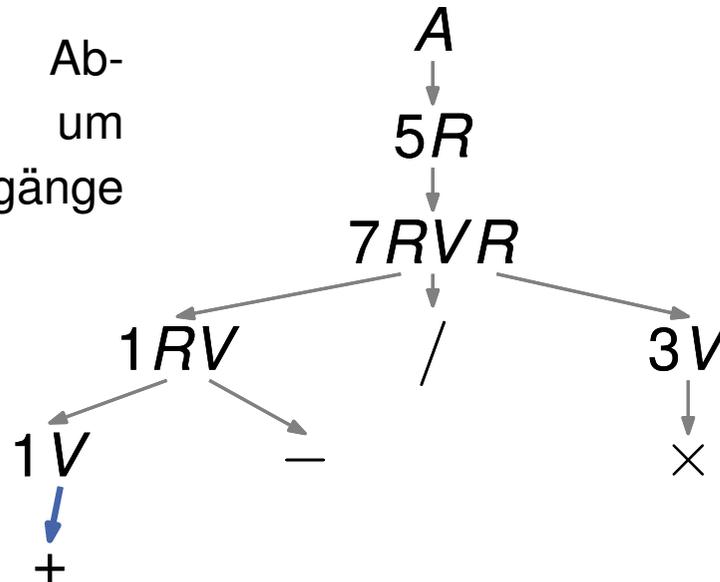


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.



Umgekehrte Polnische Notation

Ausdruck

Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “

Verknüpfung

$A \rightarrow 0 \mid 1 \mid \dots \mid 9$

$R \rightarrow 0V \mid 1V \mid \dots$

$V \rightarrow + \mid - \mid \times \mid /$

$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$

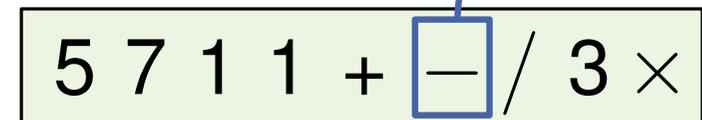
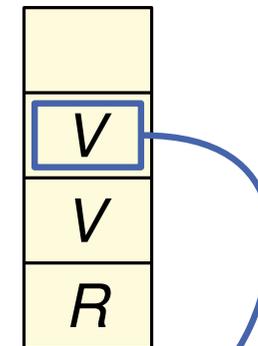
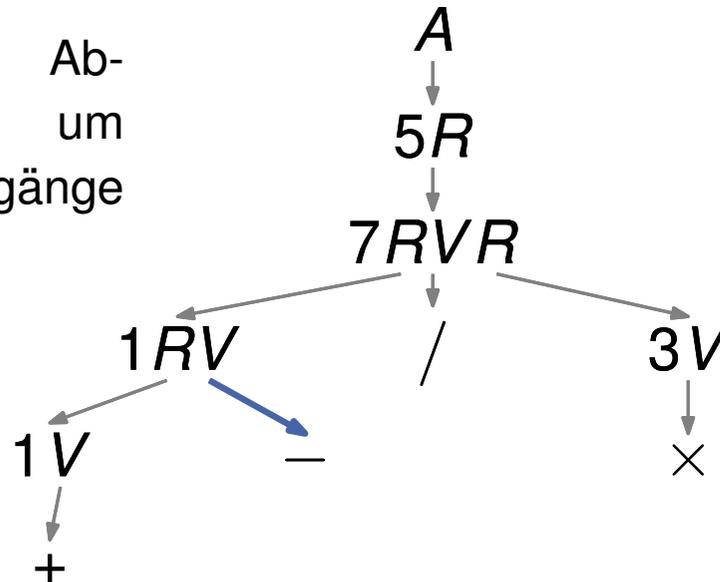
$R \rightarrow 0RV \mid 1RV \mid \dots$

$R \rightarrow 0VR \mid 1VR \mid \dots$

$R \rightarrow 0RVR \mid 1RVR \mid \dots$

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.



Umgekehrte Polnische Notation

Ausdruck

Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “

Verknüpfung

$A \rightarrow 0 \mid 1 \mid \dots \mid 9$

$R \rightarrow 0V \mid 1V \mid \dots$

$V \rightarrow + \mid - \mid \times \mid /$

$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$

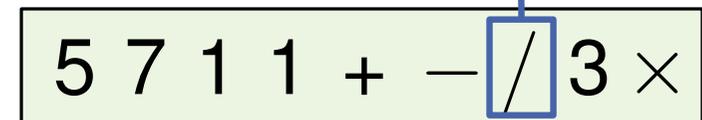
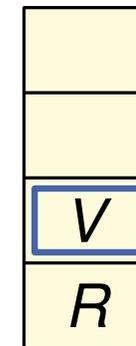
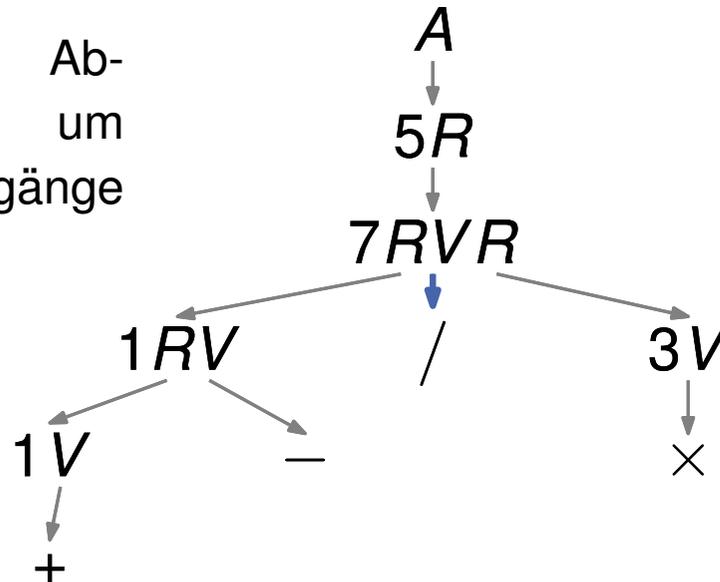
$R \rightarrow 0RV \mid 1RV \mid \dots$

$R \rightarrow 0VR \mid 1VR \mid \dots$

$R \rightarrow 0RVR \mid 1RVR \mid \dots$

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

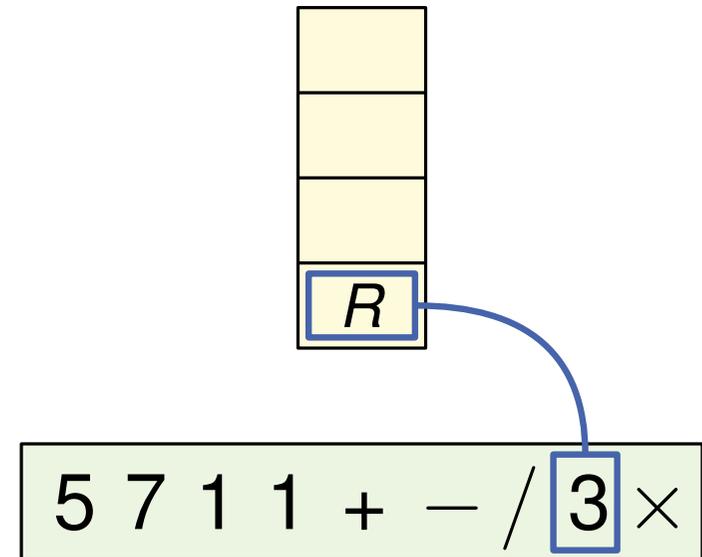
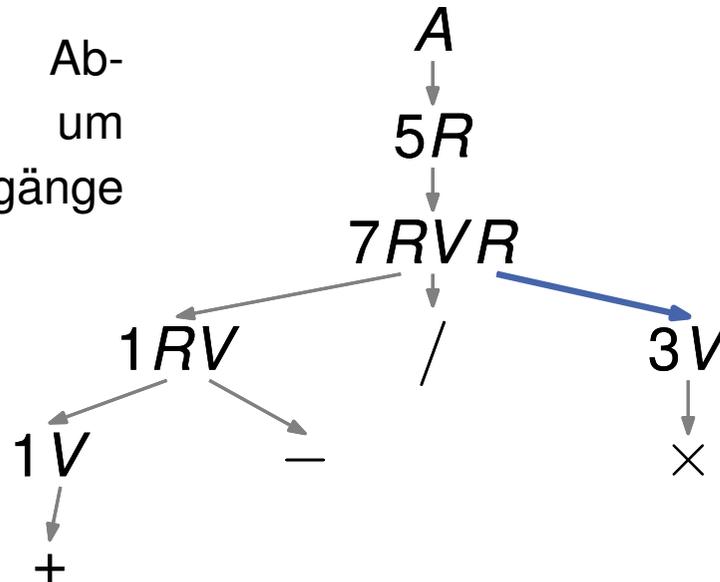


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

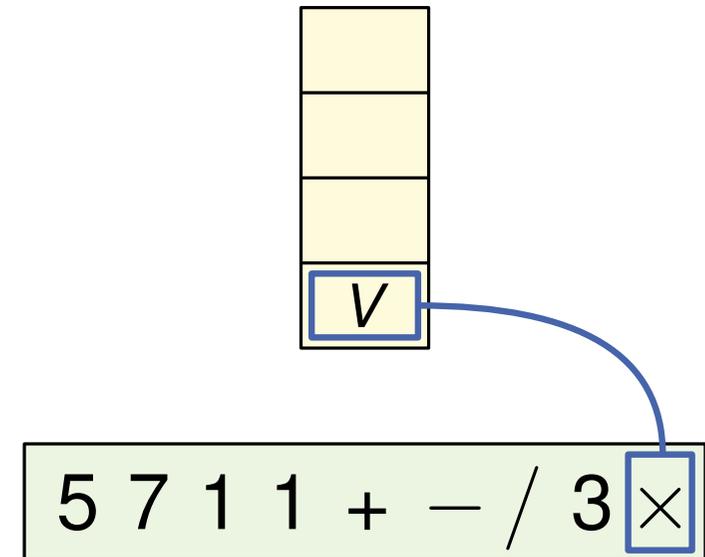
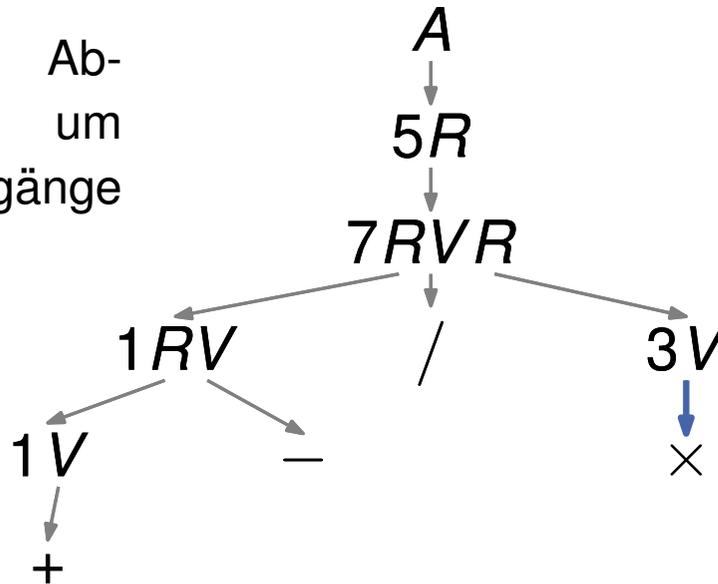


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.

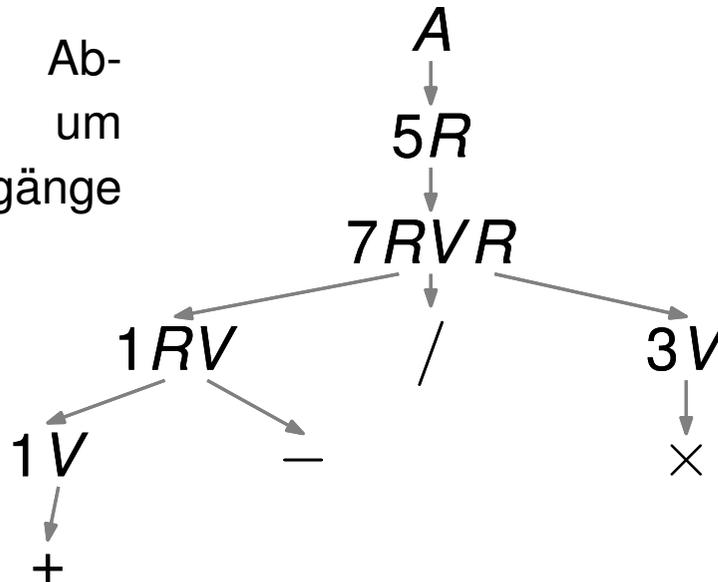


Umgekehrte Polnische Notation

Ausdruck	Rest „ $R \rightarrow AV \mid AVAV \mid \dots$ “	Verknüpfung
$A \rightarrow 0 \mid 1 \mid \dots \mid 9$	$R \rightarrow 0V \mid 1V \mid \dots$	$V \rightarrow + \mid - \mid \times \mid /$
$A \rightarrow 0R \mid 1R \mid \dots \mid 9R$	$R \rightarrow 0RV \mid 1RV \mid \dots$	
	$R \rightarrow 0VR \mid 1VR \mid \dots$	
	$R \rightarrow 0RVR \mid 1RVR \mid \dots$	

Gesucht: PDA-Abarbeitung von $5\ 7\ 1\ 1\ +\ -\ / \ 3\ \times$

Benutze den Ableitungsbaum, um günstige Übergänge zu finden.



Akzeptiert durch leeren Stack

5 7 1 1 + - / 3 x

