

# Kellerautomaten

# Spracherkennung (Syntaxanalyse)

- Algorithmus gesucht, der für  $L \subseteq T^*$  (möglichst schnell) entscheidet, ob  $w \in L$  (Lösung des Wortproblems)

Grammatik	Automat	Aufwand
rechtslinear	endlich	linear
kontextfrei	?	?

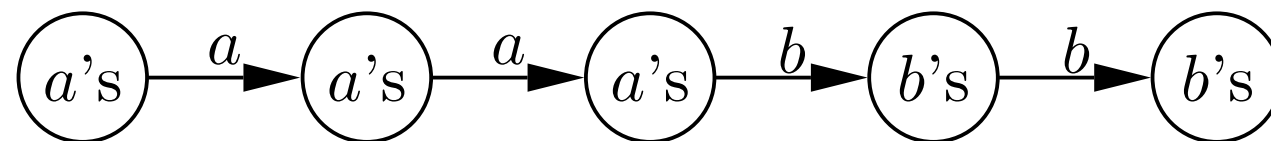
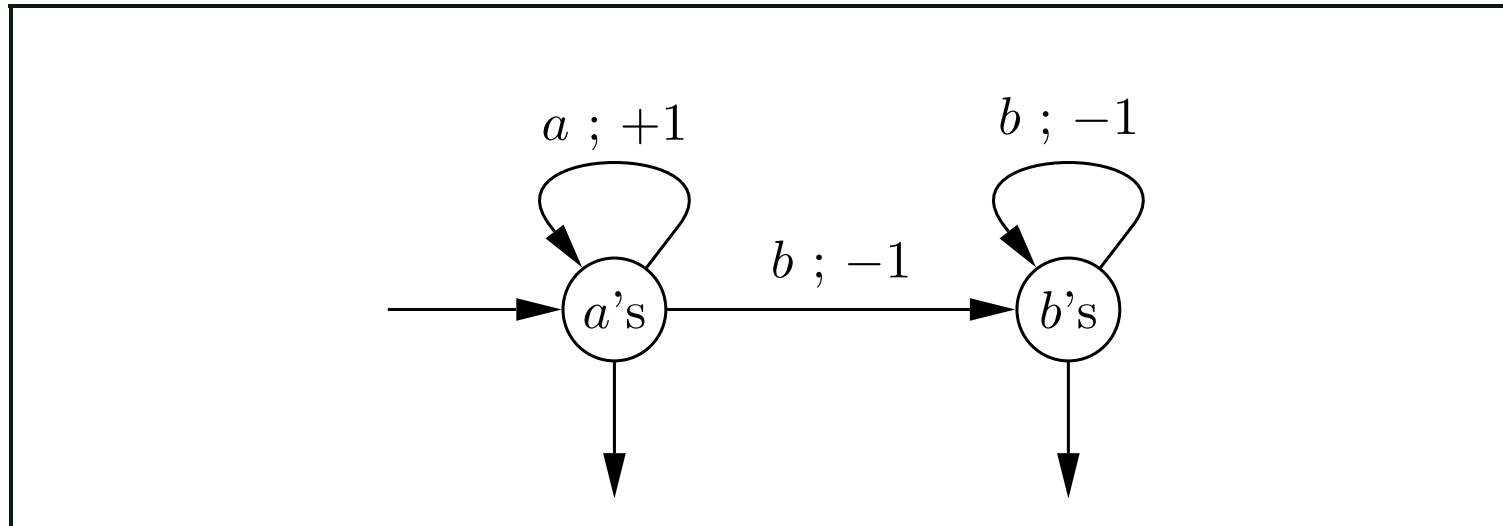
**Problem:** endliche Automaten können nicht unbeschränkt mitzählen und sich keine unbeschränkt langen Teilwörter merken

# Beispiel: Klammerstrukturen

$$L_{bal} = \{a^n b^n \mid n \in \mathbb{N}\}$$

- ▶ nicht regulär
- ▶ kontextfrei erzeugt durch  $S ::= aSb \mid \lambda$

# Idee: Endlicher Automat mit Zähler



Zähler

0

1

2

1

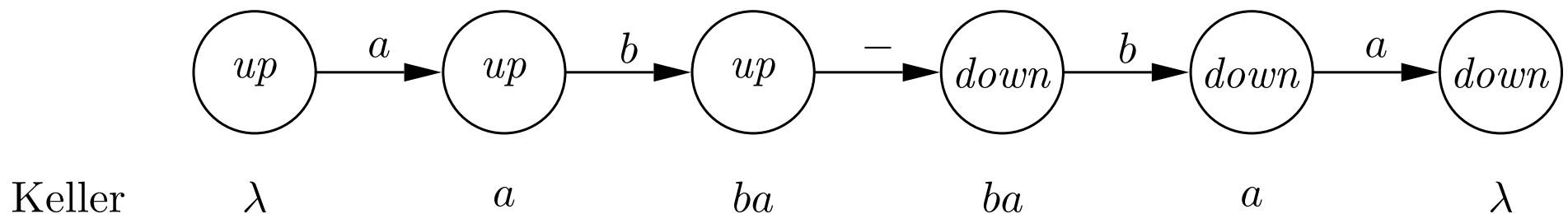
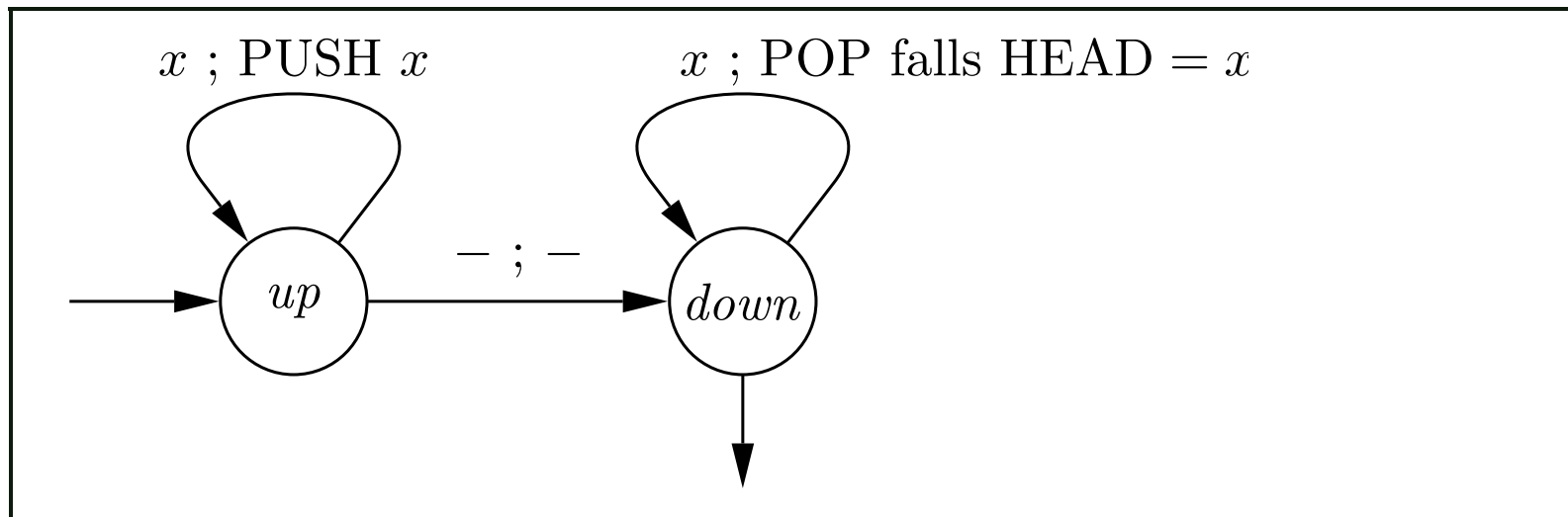
0

# Beispiel: Palindrome

$$L_{pde} = \{wtrans(w)\}$$

- ▶ nicht regulär
- ▶ kontextfrei erzeugt durch  $S ::= xSx \mid \lambda \text{ f.a. } x \in T$

# Endlicher Automat mit Keller/Speicher



# Nichtdeterministischer Kellerautomat

Idee:

endlichen Automaten mit Zusatzspeicher in Form eines Kellers (**Stapel, Stack**) mit Speicheroperationen pro Übergang

Keller über  $X$ :

$X^*$  mit den Operationen **push, head, pop**

(d.h.  $push(x, u) = xu$ ,  $head(xu) = x$ ,  $pop(xu) = u$  für alle  $x \in X$ ,  $u \in X^*$ )

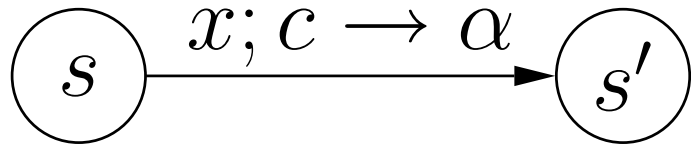
## Nichtdeterministischer Kellerautomat

$K = (Z, I, C, d, s_0, F, c_0)$  mit

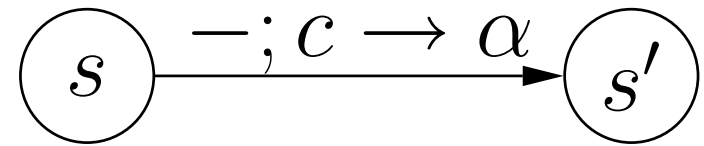
- $Z$ : endliche Menge von **Zuständen**
- $I$ : endliches **Eingabealphabet** mit  $- \notin I$
- $C$ : endliche Menge von **Kellersymbolen**
- $d: Z \times (I \cup \{-\}) \times C \leadsto Z \times C^*$ :  
**Zustandsüberführung**
- $s_0 \in Z$ : **Startzustand**
- $F \subseteq Z$ : **Endzustände**
- $c_0 \in C$ : **initiales Kellersymbol**



# Graphische Darstellung

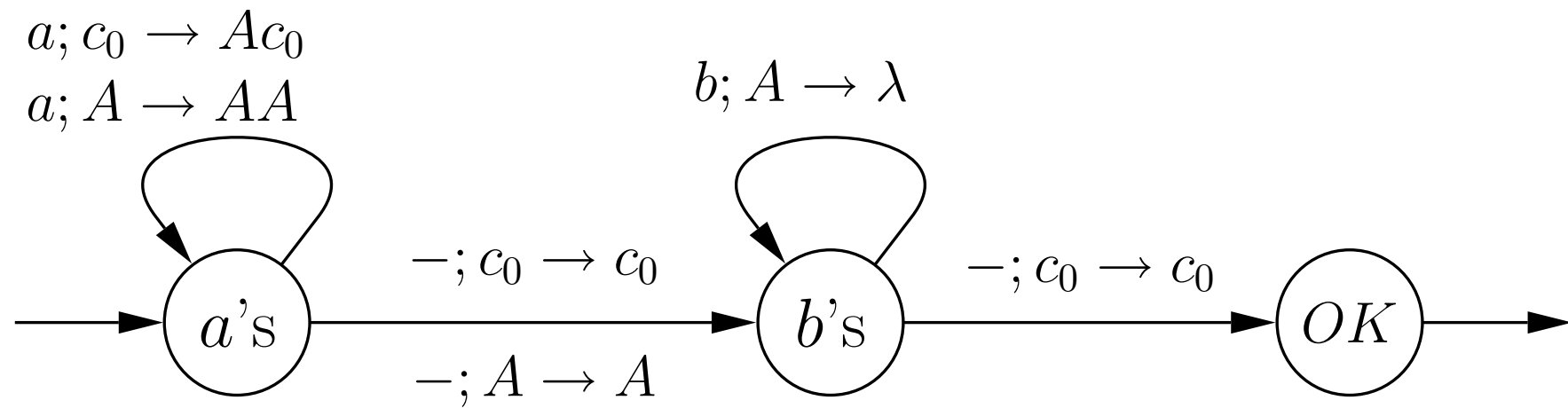


für  $(s', \alpha) \in d(s, x, c)$



für  $(s', \alpha) \in d(s, -, c)$

# Beispiel



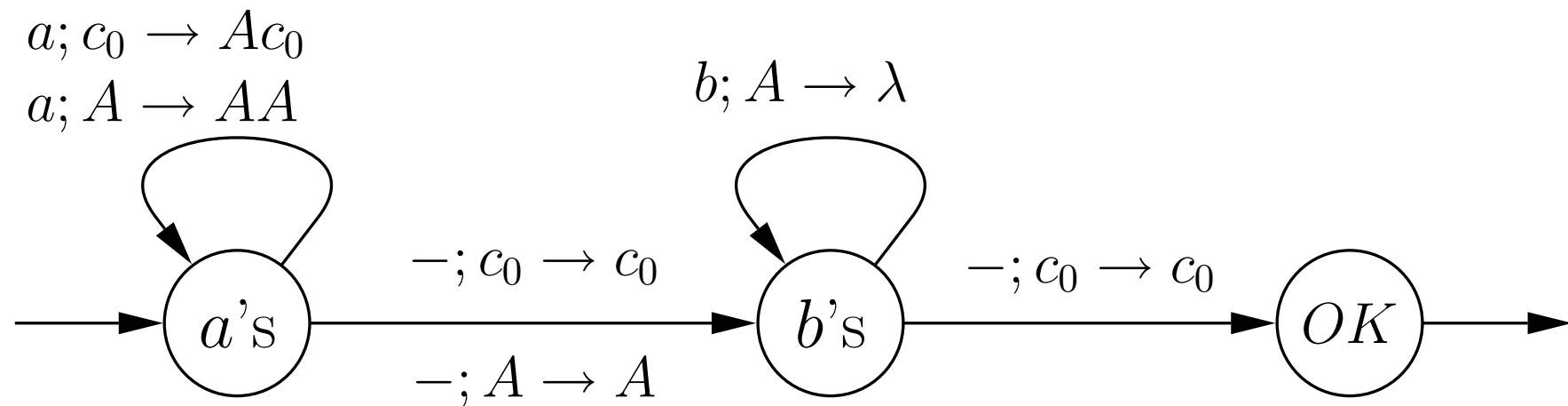
# Konfiguration

Eine **Konfiguration**  $con = (s, w, \gamma)$  besteht aus einem Zustand  $s \in Z$ , einem Wort  $w \in I^*$  und einem Kellerwort  $\gamma \in C^*$ .

## Wichtige Konfigurationen

- **Anfangskonfiguration:**  $con_0 = (s_0, w, c_0)$
- **Endkonfiguration:**  $con_F = (s, \lambda, \gamma)$  mit  $s \in F$

# Beispiel



- **Konfiguration:**  $(b's, bb, AA c_0)$
- **Anfangskonfiguration:**  $(a's, aabb, c_0)$
- **Endkonfiguration:**  $(OK, \lambda, c_0)$

## Folgekonfiguration

$(s, xv, c\gamma) \vdash (s', v, \alpha\gamma)$ , falls  $(s', \alpha) \in d(s, x, c)$

$(s, v, c\gamma) \vdash (s', v, \alpha\gamma)$ , falls  $(s', \alpha) \in d(s, -, c)$

### ► Schreibweise:

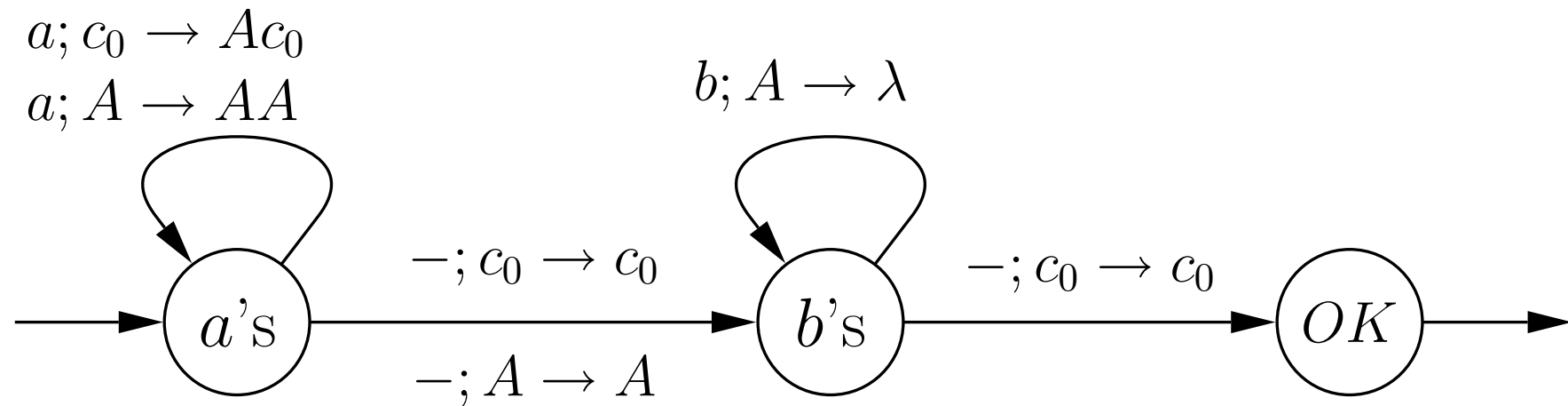
$$con = con_0 \vdash con_1 \vdash \dots \vdash con_n = con'$$

kann durch

$$\boxed{con \xrightarrow{n} con'} \quad \text{oder} \quad \boxed{con \xrightarrow{*} con'}$$

abgekürzt werden.

## Beispiel



### Konfigurationsfolge:

$(a's, abb, Ac_0) \vdash (a's, bb, AA c_0) \vdash (b's, bb, AA c_0) \vdash$   
 $(b's, b, Ac_0)$

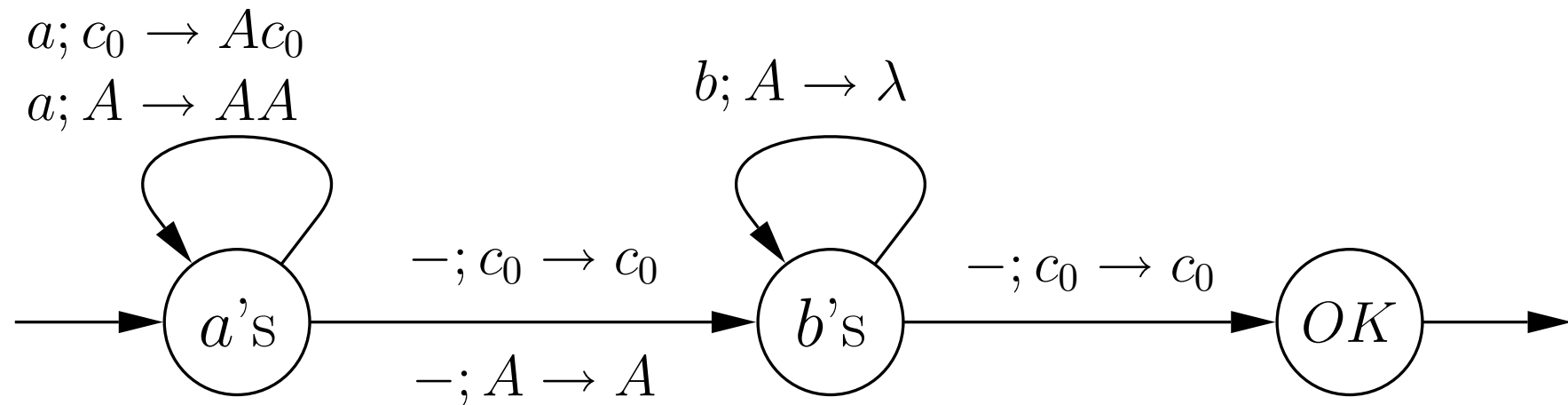
# Erkannte Sprache

Gegeben:  $K = (Z, I, C, d, s_0, F, c_0)$

## Erkannte Sprache

- $K$  **erkennt**  $w \in I^*$ , falls  $(s_0, w, c_0) \xrightarrow{*} (s'', \lambda, \gamma)$  für  $s'' \in F$ .
- Die Menge aller von  $K$  erkannten Wörter bildet die **erkannte Sprache**  $L(K)$ .

## Beispiel



Erkannte Sprache:  $\{a^n b^n \mid n \in \mathbb{N}\}$

Erkennen von  $a^2 b^2$ :

$(a's, aabb, c_0) \vdash (a's, abb, Ac_0) \vdash (a's, bb, AAc_0) \vdash$   
 $(b's, bb, AAc_0) \vdash (b's, b, Ac_0) \vdash (b's, \lambda, c_0) \vdash (OK, \lambda, c_0)$

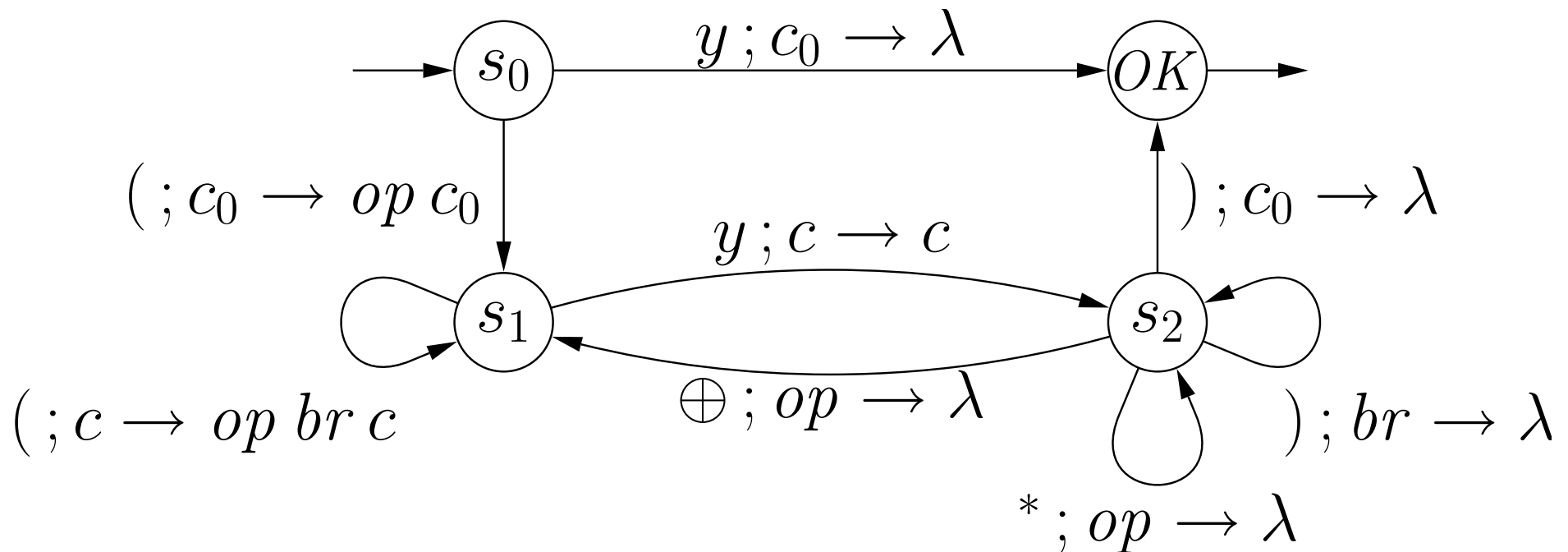


# Deterministischer Kellerautomat

$K = (Z, I, C, d, s_0, F, c_0)$  mit

- $Z$ : endliche Menge von Zuständen
- $I$ : endliches Eingabealphabet mit  $- \notin I$
- $C$ : endliche Menge von Kellersymbolen
- $d: Z \times (I \cup \{-\}) \times C \leadsto Z \times C^*$ : Zustandsüberführung mit  $|d(s, x, c)| + |d(s, -, c)| \leq 1$  für alle  $s \in Z$ ,  $x \in I$ ,  $c \in C$ ,
- $s_0 \in Z$ : Startzustand
- $F \subseteq Z$ : Endzustände
- $c_0 \in C$ : initiales Kellersymbol

# Beispiel: Reguläre Ausdrücke



mit  $y \in I \cup \{empty, lambda\}$ ,  $c \in \{op, br, c_0\}$  und  $\oplus \in \{+, \circ\}$

	$(s_0,$	$((x + \text{empty})^*) \circ \text{lambda}),$	$c_0)$
└──	$(s_1,$	$((x + \text{empty})^*) \circ \text{lambda}),$	$\text{op } c_0)$
└──	$(s_1,$	$(x + \text{empty})^*) \circ \text{lambda}),$	$\text{op br op } c_0)$
└──	$(s_1,$	$x + \text{empty})^*) \circ \text{lambda}),$	$\text{op br op br op } c_0)$
└──	$(s_2,$	$+ \text{empty})^*) \circ \text{lambda}),$	$\text{op br op br op } c_0)$
└──	$(s_1,$	$\text{empty})^*) \circ \text{lambda}),$	$\text{br op br op } c_0)$
└──	$(s_2,$	$)^*) \circ \text{lambda}),$	$\text{br op br op } c_0)$
└──	$(s_2,$	$^*) \circ \text{lambda}),$	$\text{op br op } c_0)$
└──	$(s_2,$	$) \circ \text{lambda}),$	$\text{br op } c_0)$
└──	$(s_2,$	$\circ \text{lambda}),$	$\text{op } c_0)$
└──	$(s_1,$	$\text{lambda}),$	$c_0)$
└──	$(s_2,$	$),$	$c_0)$
└──	$(OK,$	$\lambda,$	$)$

$$\begin{array}{lcl}
 & (s_0, & ((x + \text{empty})^*) \circ \text{lambda}), \quad c_0) \\
 \vdash & (s_1, & (x + \text{empty})^*) \circ \text{lambda}), \quad \text{op } c_0) \\
 \vdash & (s_1, & x + \text{empty})^*) \circ \text{lambda}), \text{op br op } c_0) \\
 \vdash & (s_2, & + \text{empty})^*) \circ \text{lambda}), \text{op br op } c_0) \\
 \vdash & (s_1, & \text{empty})^*) \circ \text{lambda}), \quad \text{br op } c_0) \\
 \vdash & (s_2, & )^*) \circ \text{lambda}), \quad \text{br op } c_0) \\
 \vdash & (s_2, & ^*) \circ \text{lambda}), \quad \text{op } c_0) \\
 \vdash & (s_2, & ) \circ \text{lambda}), \quad c_0) \\
 \vdash & (OK, & \circ \text{lambda}), \quad )
 \end{array}$$

## Mächtigkeit

- $\mathcal{L}_{DKA}$ : Klasse aller von deterministischen Kellerautomaten erkannten Sprachen
- $\mathcal{L}_{NKA}$ : Klasse aller von nichtdeterministischen Kellerautomaten erkannten Sprachen

Es gibt keinen deterministischen Kellerautomaten, der

$$\{wtrans(w) \mid w \in \{a, b\}^*\}$$

erkennt.

Also gilt

$$\mathcal{L}_{DKA} \subset \mathcal{L}_{NKA}$$

# Übersetzung kontextfreier Grammatiken in Kellerautomaten

Gegeben: kontextfreie Grammatik  $G = (N, T, P, S)$  mit  $c_0, c_{OK} \notin N \cup T$ .

$$\begin{aligned} PDA(G) = & \\ & (\{s_0, OK\}, T, N \cup T \cup \{c_0, c_{OK}\}, d_G, s_0, \{OK\}, c_0) \text{ mit} \\ & \bullet d_G(s_0, -, c_0) = \{(s_0, S c_{OK})\}, \\ & \bullet d_G(s_0, -, A) = \{(s_0, u) \mid A ::= u \in P\}, \\ & \bullet d_G(s_0, x, x) = \{(s_0, \lambda)\} \text{ für alle } x \in T \text{ und} \\ & \bullet d_G(s_0, -, c_{OK}) = \{(OK, \lambda)\}. \end{aligned}$$

## Beispiel: Klammergebirge

$$G = (\{A\}, \{[, ]\}, \{A ::= AA \mid [A] \mid \lambda\}, A)$$

$L(G)$ : alle korrekten Klammerungen über dem Klammerpaar  $[$  und  $]$ .

