# Grundbegriffe der Informatik

## Übung

Simon Wacker

Karlsruher Institut für Technologie

Wintersemester 2015/2016

# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe $0$?

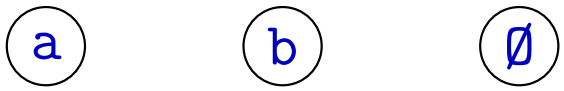
Antwort:

# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe 0?

Antwort: 3



# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe 1?

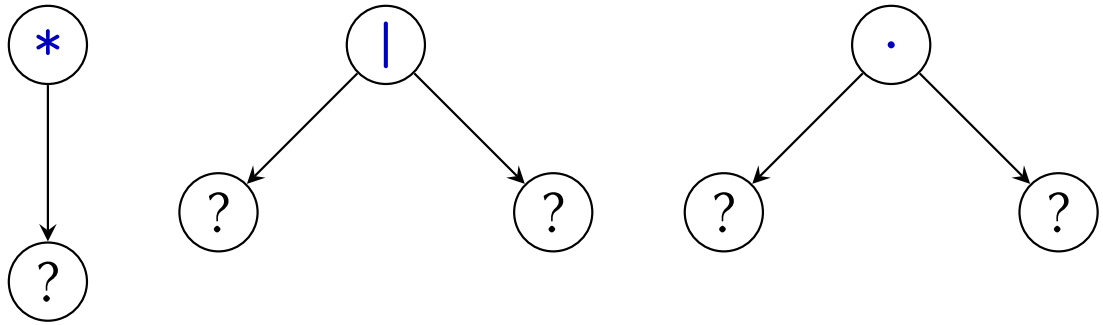
Antwort:

# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe 1?

Antwort: $3 + 3 \cdot 3 + 3 \cdot 3 = 21$



# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe 2?

Antwort:

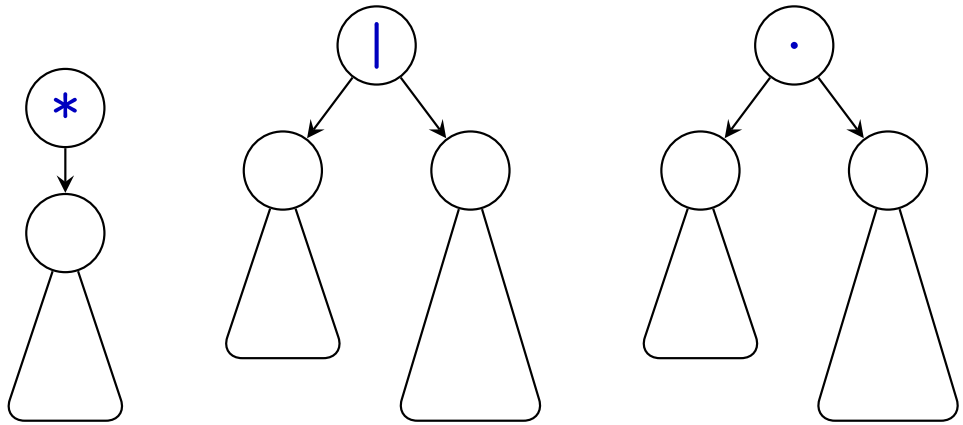
# Regex-Bäume — Anzahl

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Anzahl Regex-Bäume über $A$ der Höhe 2?

Antwort:

$$21 + (21 \cdot 21 + 3 \cdot 21 + 21 \cdot 3) + (21 \cdot 21 + 3 \cdot 21 + 21 \cdot 3) = 1155$$



# Regex-Bäume — Kleinste Anzahl Knoten

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Kleinste Anzahl Knoten von Regex-Bäumen der Höhe $n$?

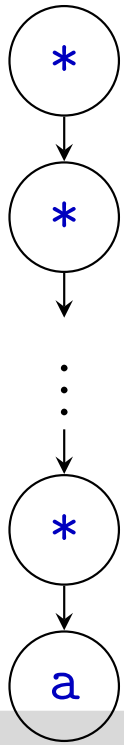
Antwort:

# Regex-Bäume — Kleinste Anzahl Knoten

$A = \{\texttt{a}, \texttt{b}\}$

Frage: Kleinste Anzahl Knoten von Regex-Bäumen der Höhe $n$?

Antwort: $n + 1$



# Regex-Bäume — Größte Anzahl Knoten

$A = \{\texttt{a}, \texttt{b}\}$

Frage: Größte Anzahl Knoten von Regex-Bäumen der Höhe $n$?

Antwort:

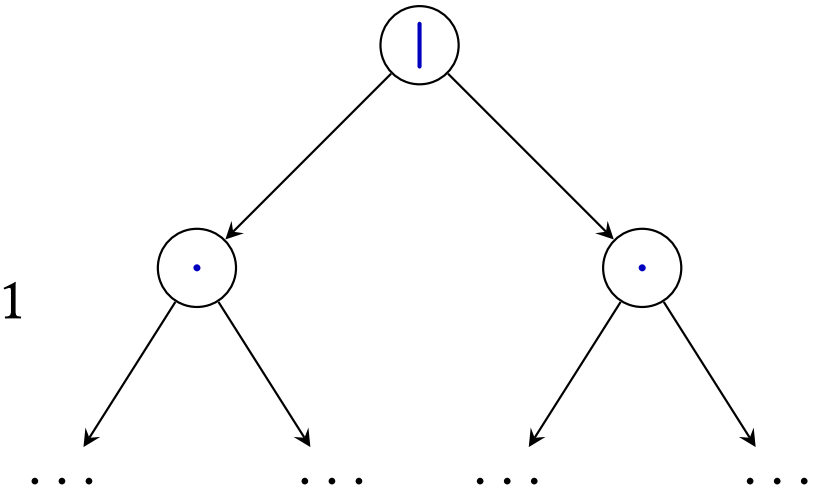
# Regex-Bäume — Größte Anzahl Knoten

$A = \{\mathtt{a}, \mathtt{b}\}$

Frage: Größte Anzahl Knoten von Regex-Bäumen der Höhe $n$?

Antwort:

$$\sum_{i=0}^{n} 2^i = \mathrm{Num}_2(1^{n+1})$$
$$= \mathrm{Num}_2(10^{n+1}) - 1$$
$$= 2^{n+1} - 1$$



# Distributivgesetz

$A$ Alphabet
$R_1, R_2, R_3$ reguläre Ausdrücke über $A$

Behauptung: $\langle (R_1 \mid R_2) R_3 \rangle = \langle R_1 R_3 \mid R_2 R_3 \rangle$

Beweis: Es gilt

$$\begin{aligned}
\langle (R_1 \mid R_2) R_3 \rangle &= \langle (R_1 \mid R_2) \rangle \cdot \langle R_3 \rangle \\
&= (\langle R_1 \rangle \cup \langle R_2 \rangle) \cdot \langle R_3 \rangle \\
&= (\langle R_1 \rangle \cdot \langle R_3 \rangle) \cup (\langle R_2 \rangle \cdot \langle R_3 \rangle) \\
&= \langle R_1 R_3 \rangle \cup \langle R_2 R_3 \rangle \\
&= \langle R_1 R_3 \mid R_2 R_3 \rangle.
\end{aligned}$$

# Regulärer Ausdruck $\rightsquigarrow$ Akzeptor

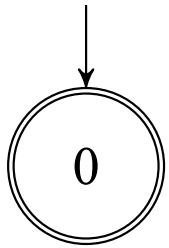
Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

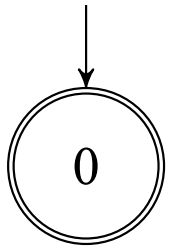



`(...)*` ⇝ Akzeptierender Anfangszustand

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

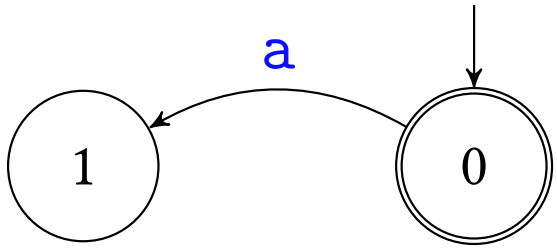



. . . | . . . ⇝ Zwei Teile

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

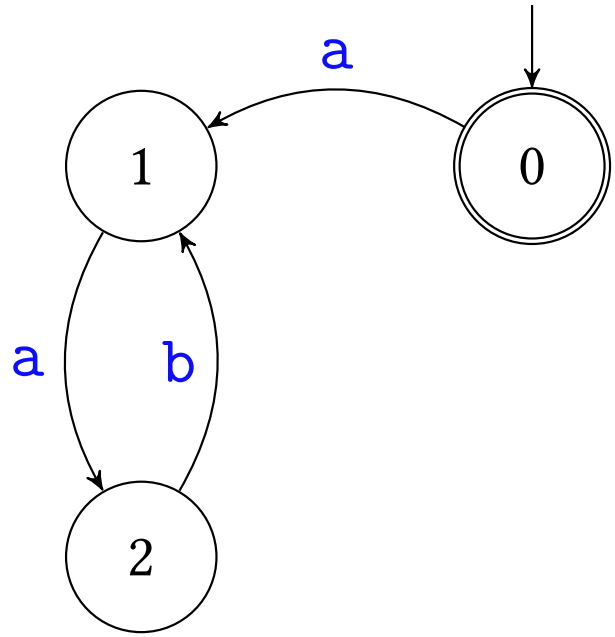



`a...` ⇝ Mit `a` in neuen Zustand

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

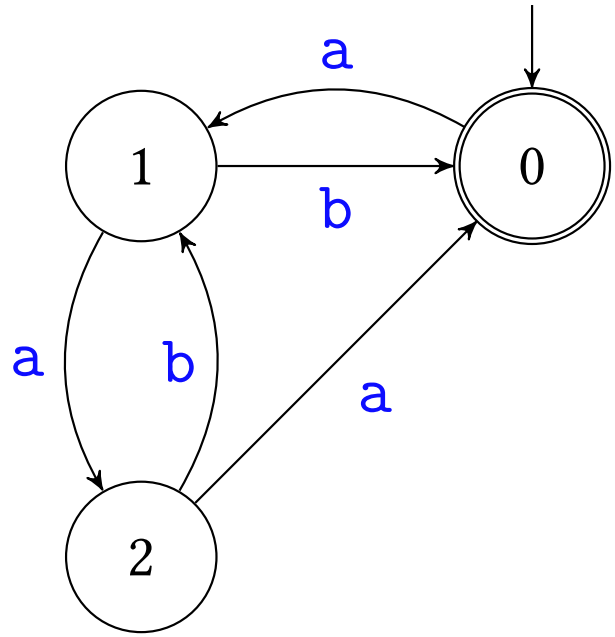



`...(ab)*...` ⇝ Mit `a` in neuen Zustand, mit `b` zurück

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

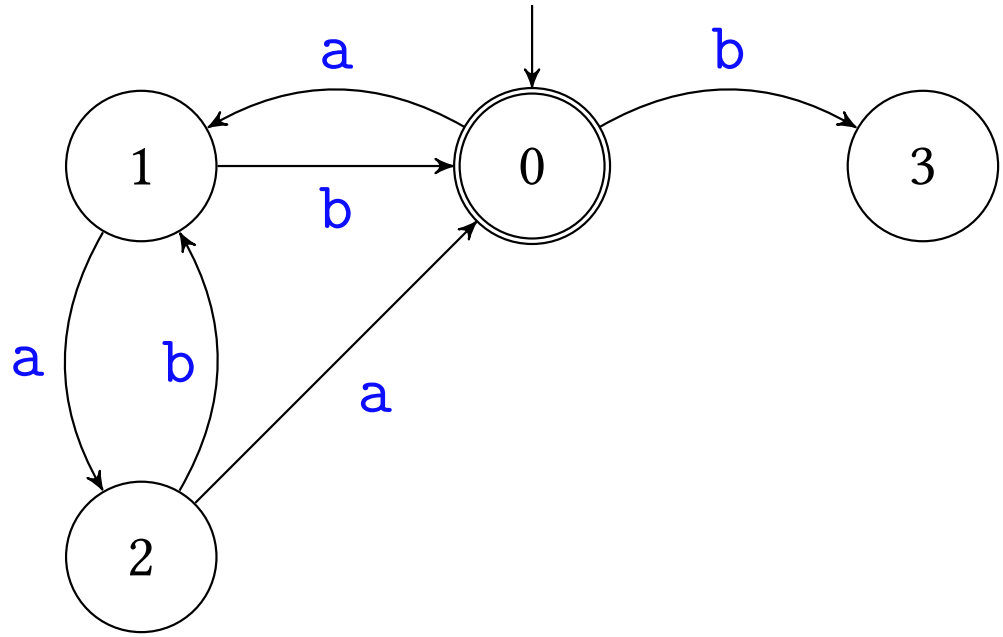



`...(b|aa)` ⇝ Mit `b` oder `aa` zurück zu 0

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

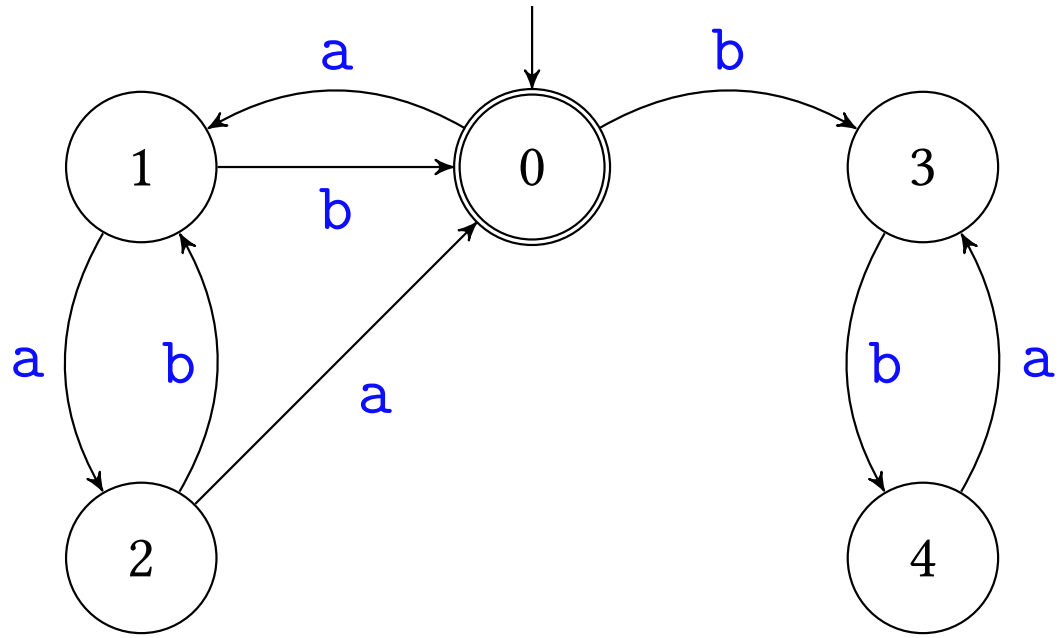



`b...` ⇝ Mit `b` in neuen Zustand

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$

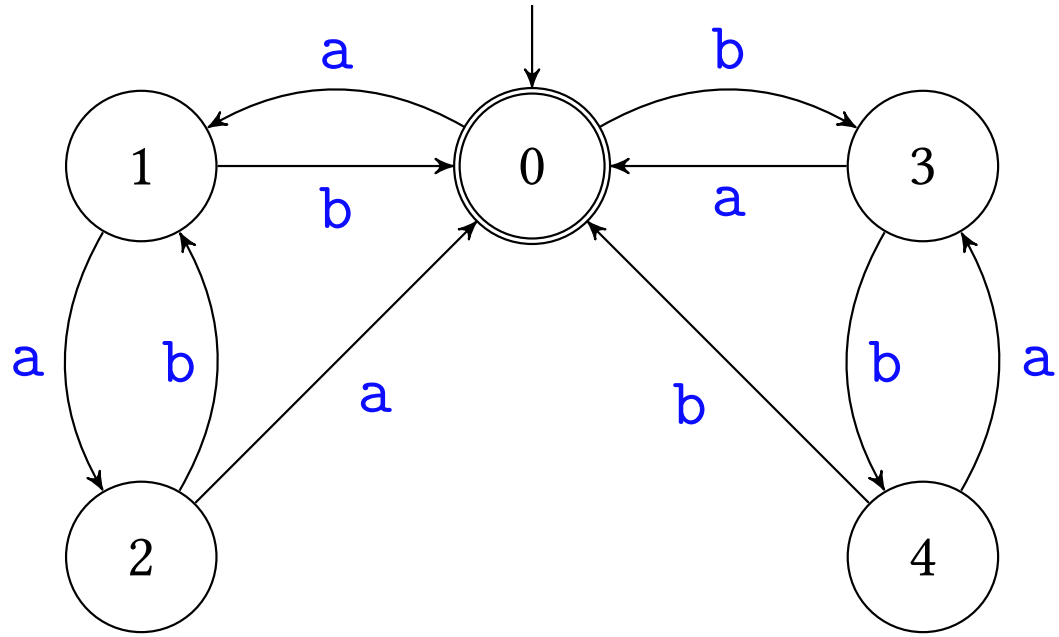



`...(ba)*...` ⇝ Mit `b` in neuen Zustand, mit `a` zurück

# Regulärer Ausdruck ⇝ Akzeptor

Konstruiere endlichen Akzeptor $A$ so, dass

$$L(A) = \langle \texttt{(a(ab)*(b|aa)|b(ba)*(a|bb))*} \rangle$$




. . . `(a|bb)` ⇝ Mit `a` oder `bb` zurück zu 0

# Endlicher Akzeptor $\rightsquigarrow$ Rechtslineare Grammatik

Gegeben: Endlicher Akzeptor $A = (Z, z_0, X, f, F)$
Gesucht: Rechtslineare Grammatik $G$ mit $L(G) = L(A)$

Idee: $G = (Z, X, z_0, P)$ so, dass

$$z_0 \Rightarrow^* x_0 x_1 \ldots x_n z \quad \text{gdw.} \quad f^*(z_0, x_0 x_1 \ldots x_n) = z$$
$$z_0 \Rightarrow^* x_0 x_1 \ldots x_n \quad \text{gdw.} \quad f^*(z_0, x_0 x_1 \ldots x_n) \in F$$

also

$$(z_1 \to x z_2) \in P \quad \text{gdw.} \quad f(z_1, x) = z_2$$
$$(z \to \epsilon) \in P \quad \text{gdw.} \quad z \in F$$

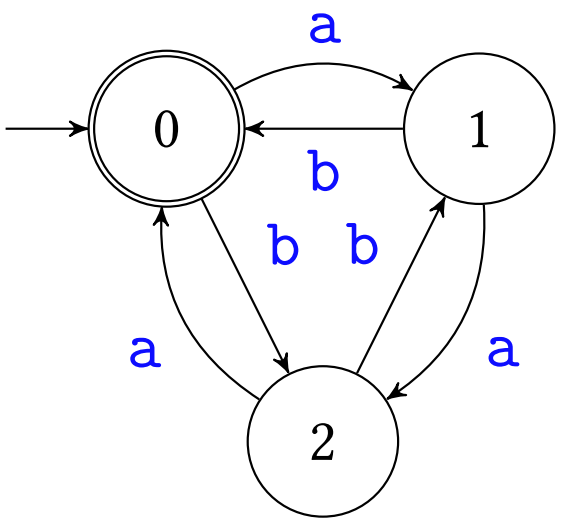
Konkret: $P = \{z \to x f(z, x) \mid z \in Z, x \in X\}$
$$\cup \{z \to \epsilon \mid z \in F\}$$

# Endlicher Akzeptor $\rightsquigarrow$ Rechtslineare Grammatik

Gegeben: Endlicher Akzeptor $A = (Z, z_0, X, f, F)$
Gesucht: Rechtslineare Grammatik $G$ mit $L(G) = L(A)$

Konkret: $P = \{z \to x f(z,x) \mid z \in Z, x \in X\}$
$\cup \{z \to \epsilon \mid z \in F\}$

Beispiel:




$G = (\{0,1,2\}, \{\mathtt{a}, \mathtt{b}\}, 0, P)$ mit

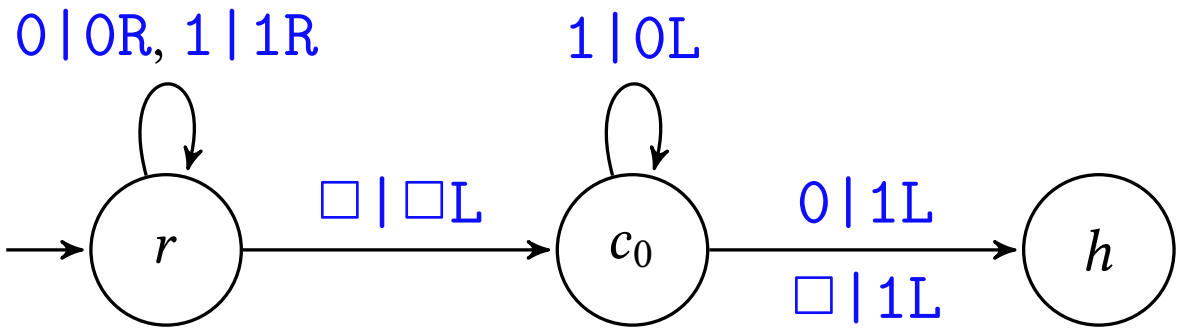
$P = \{0 \to \mathtt{a}1 \mid \mathtt{b}2 \mid \epsilon$
$1 \to \mathtt{a}2 \mid \mathtt{b}0$
$2 \to \mathtt{a}0 \mid \mathtt{b}1\}$

# Zahl in Binärdarstellung um 1 inkrementieren

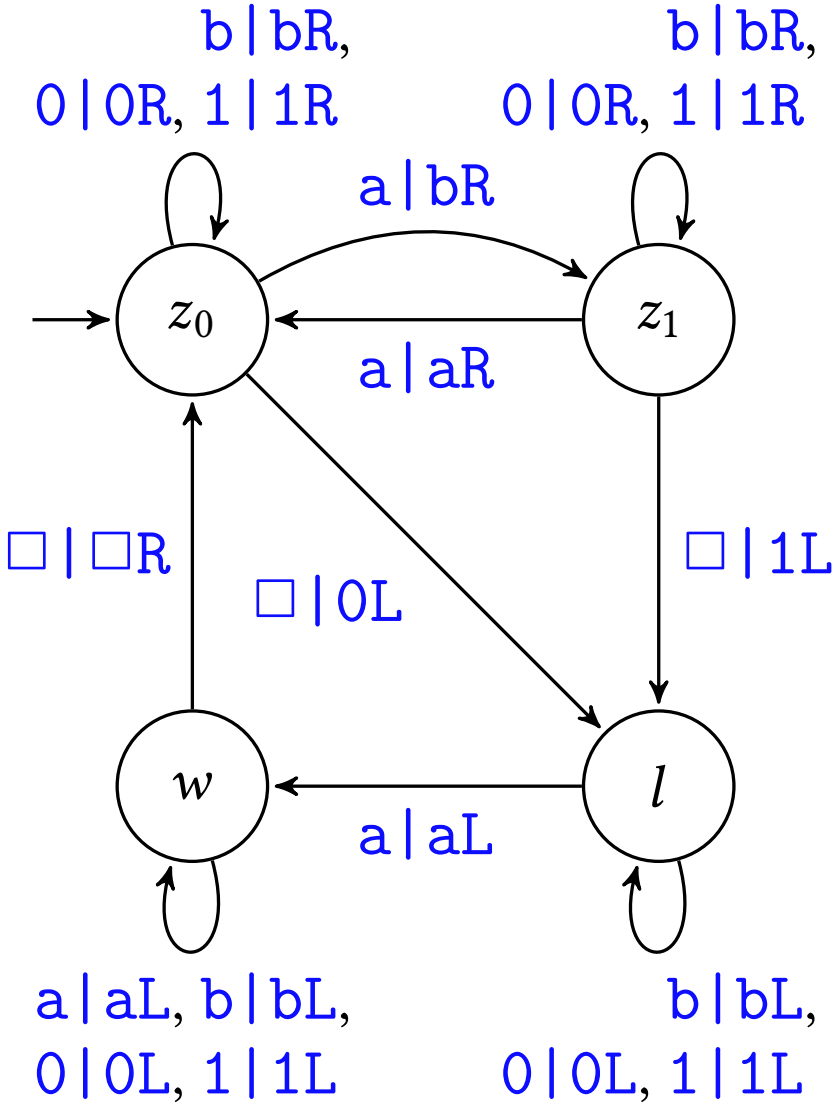
Eingabe: $w \in \{0, 1\}^*$
Ausgabe: $u \in \{0, 1\}^*$ so, dass $\mathrm{Num}_2(w) + 1 = \mathrm{Num}_2(u)$

Beispiel: $\mathrm{Num}_2(100111) + 1 = \mathrm{Num}_2(101000)$

Lösungsidee: Niederwertige Bits bis zur ersten 0 kippen

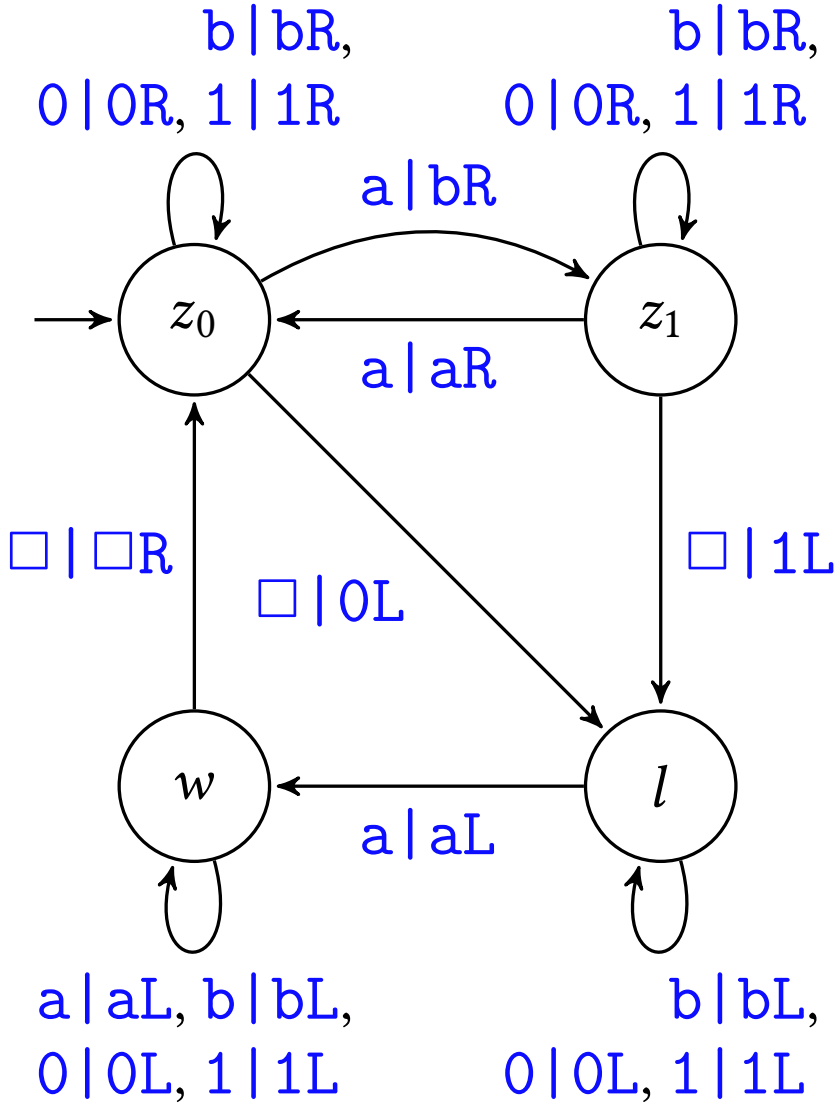


# Turingmaschine



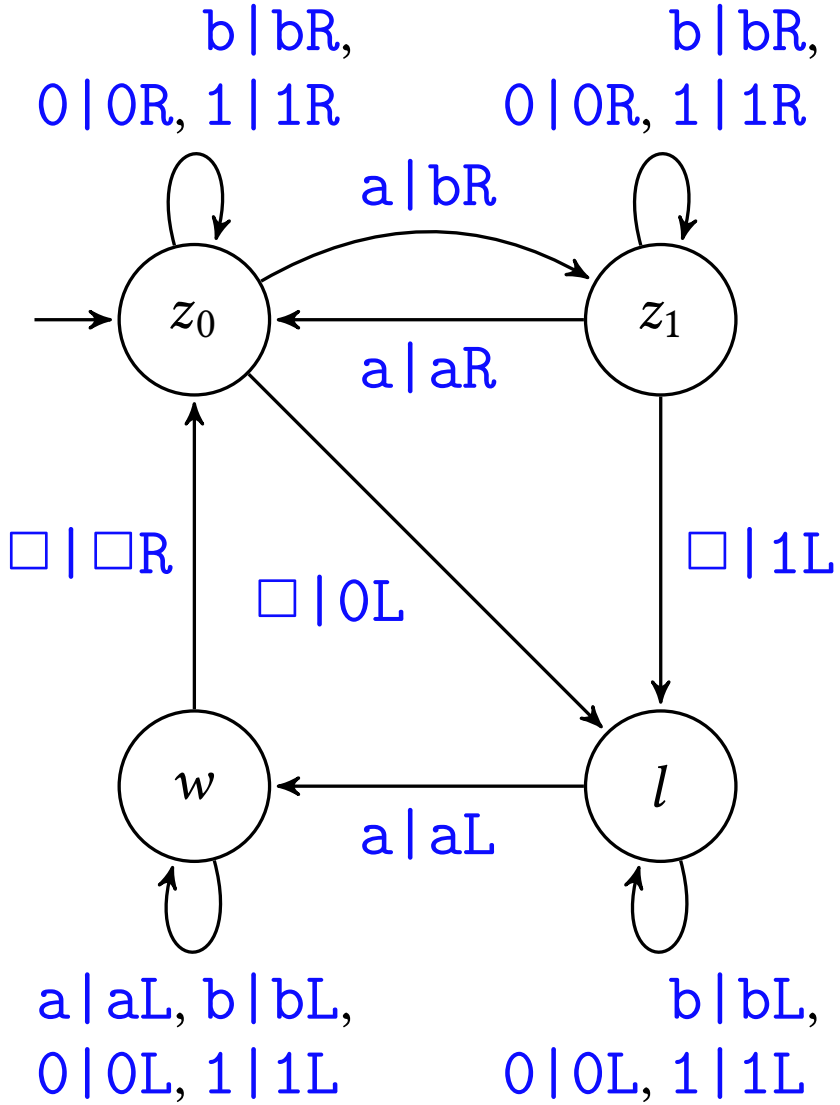

- In $z_k$ außer bei □ Kopf nach rechts
- In $z_0$ bei a schreibe b, gehe in $z_1$
- In $z_1$ bei a gehe in $z_0$
- $k$ in $z_k$ ist Anzahl gelesener a mod 2
- In $z_k$ bei □ schreibe $k$, gehe in $l$
- In $l$ Kopf nach links bis zum ersten a, dann nach $w$, und halt bei □
- In $w$ zum Wortanfang, dann in $z_0$

# Turingmaschine




- Kopf läuft von links nach rechts
- Beginnend mit dem ersten a wird jedes zweite durch b ersetzt
- Bei gerader Anzahl von a wird 0 ans Wortende geschrieben
- Bei ungerader Anzahl 1
- Falls kein a mehr auf dem Band, Ende!
- Ansonsten zurück zum Wortanfang und alles noch einmal

# Turingmaschine




Eingabe $w \in \{\mathtt{a}, \mathtt{b}\}$

Anzahl der $\mathtt{a}$ halbiert sich bei jedem Durchlauf

Ans Wortende wird geschrieben

1. $\mathrm{N}_{\mathtt{a}}(w) \mod 2$
2. $\lfloor \frac{\mathrm{N}_{\mathtt{a}}(w)}{2} \rfloor \mod 2$
3. $\lfloor \frac{\mathrm{N}_{\mathtt{a}}(w)}{4} \rfloor \mod 2$
4. usw.

Binärdarstellung von $\mathrm{N}_{\mathtt{a}}(w)$ wird gespiegelt ans Wortende geschrieben

Am Ende steht auf dem Band: $\mathtt{b}^{|w|}\mathrm{R}(\mathrm{Repr}_2(\mathrm{N}_{\mathtt{a}}(w)))$.