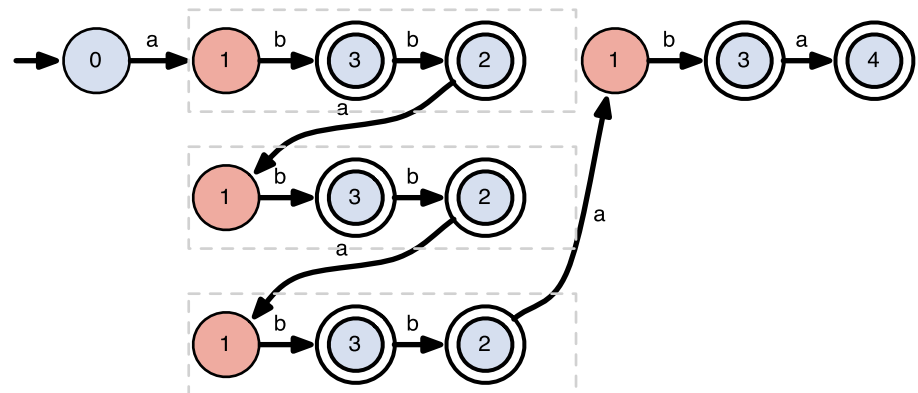


# Berechenbarkeitstheorie

## 4. Vorlesung



Dr. Franziska Jahnke

Institut für Mathematische Logik und Grundlagenforschung

WWU Münster

# Reguläre Ausdrücke

2

4. Vorlesung

# Reguläre Ausdrücke

2

- Formalismus zum Beschreiben von formalen Sprachen

# Reguläre Ausdrücke

- Formalismus zum Beschreiben von formalen Sprachen
- ein **Regulärer Ausdruck (RA)** ist ein Wort über dem Alphabet  $\Sigma \cup \{\emptyset, \varepsilon, (, ), +, \cdot, *\}$

# Reguläre Ausdrücke

- Formalismus zum Beschreiben von formalen Sprachen
- ein **Regulärer Ausdruck (RA)** ist ein Wort über dem Alphabet  $\Sigma \cup \{\emptyset, \varepsilon, (, ), +, \cdot, *\}$

## Syntax

# Reguläre Ausdrücke

- Formalismus zum Beschreiben von formalen Sprachen
- ein **Regulärer Ausdruck (RA)** ist ein Wort über dem Alphabet  $\Sigma \cup \{\emptyset, \varepsilon, (, ), +, \cdot, *\}$

- Syntax**
- (1)  $\forall a \in \Sigma \cup \{\emptyset, \varepsilon\}$ :  $a$  ist RA
  - (2) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $(A + B)$
  - (3) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $A \cdot B$  (kurz  $AB$ )
  - (4) Wenn  $A$  Reg. Ausdruck, dann auch  $A^*$
  - (5) Wenn  $A$  Reg. Ausdruck, dann auch  $(A)$

# Reguläre Ausdrücke

- Formalismus zum Beschreiben von formalen Sprachen
- ein **Regulärer Ausdruck (RA)** ist ein Wort über dem Alphabet  $\Sigma \cup \{\emptyset, \varepsilon, (, ), +, \cdot, *\}$

- Syntax**
- (1)  $\forall a \in \Sigma \cup \{\emptyset, \varepsilon\}$ :  $a$  ist RA
  - (2) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $(A + B)$
  - (3) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $A \cdot B$  (kurz  $AB$ )
  - (4) Wenn  $A$  Reg. Ausdruck, dann auch  $A^*$
  - (5) Wenn  $A$  Reg. Ausdruck, dann auch  $(A)$

- Bsp**
- $((aba^* + abb)^* ab + \varepsilon)$  ist RA
  - $(a + b)b(*b)$  ist kein RA

# Reguläre Ausdrücke

- Formalismus zum Beschreiben von formalen Sprachen
- ein **Regulärer Ausdruck (RA)** ist ein Wort über dem Alphabet  $\Sigma \cup \{\emptyset, \varepsilon, (, ), +, \cdot, *\}$

- Syntax**
- (1)  $\forall a \in \Sigma \cup \{\emptyset, \varepsilon\}$ :  $a$  ist RA
  - (2) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $(A + B)$
  - (3) Wenn  $A, B$  Reg. Ausdrücke, dann auch  $A \cdot B$  (kurz  $AB$ )
  - (4) Wenn  $A$  Reg. Ausdruck, dann auch  $A^*$
  - (5) Wenn  $A$  Reg. Ausdruck, dann auch  $(A)$

- Bsp**
- $((aba^* + abb)^* ab + \varepsilon)$  ist RA
  - $(a + b)b(*b)$  ist kein RA

Induktive Definition korrekt, da RA durch die Anwendung der (umgekehrten) Regeln immer kürzer werden



# Semantik Regulärer Ausdrücke

3

# Semantik Regulärer Ausdrücke

3

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

# Semantik Regulärer Ausdrücke

3

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
- $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$

# Semantik Regulärer Ausdrücke

3

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$

# Semantik Regulärer Ausdrücke

3

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$
- (4)  $R = A^*$  und  $A$  ist RA, dann  $L(R) = L(A)^*$

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$
- (4)  $R = A^*$  und  $A$  ist RA, dann  $L(R) = L(A)^*$
- (5)  $R = (A)$  und  $A$  ist RA, dann  $L(R) = L(A)$

# Semantik Regulärer Ausdrücke

3

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$
- (4)  $R = A^*$  und  $A$  ist RA, dann  $L(R) = L(A)^*$
- (5)  $R = (A)$  und  $A$  ist RA, dann  $L(R) = L(A)$

**Reihenfolge der Operatoren:  $*$  vor  $\cdot$  vor  $+$**



Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$
- (4)  $R = A^*$  und  $A$  ist RA, dann  $L(R) = L(A)^*$
- (5)  $R = (A)$  und  $A$  ist RA, dann  $L(R) = L(A)$

**Reihenfolge der Operatoren:  $*$  vor  $\cdot$  vor  $+$**

**Bsp.**  $R = (a + b)^* aba(a + b)^*$

Für jeden RA  $R$  definieren wir induktiv eine Sprache  $L(R)$ :

- (1)
  - $R = \emptyset$ , dann  $L(R) = \emptyset$
  - $R = \varepsilon$ , dann  $L(R) = \{\varepsilon\}$
  - $R = a$ , und  $a \in \Sigma$ , dann  $L(R) = \{a\}$
- (2)  $R = (A + B)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \cup L(B)$
- (3)  $R = (AB)$  und  $A, B$  sind RA, dann  $L(R) = L(A) \circ L(B)$
- (4)  $R = A^*$  und  $A$  ist RA, dann  $L(R) = L(A)^*$
- (5)  $R = (A)$  und  $A$  ist RA, dann  $L(R) = L(A)$

**Reihenfolge der Operatoren:  $*$  vor  $\cdot$  vor  $+$**

**Bsp.**  $R = (a + b)^* aba(a + b)^*$

$L(R) = \{w \in \{a, b\}^* \mid w \text{ enthält Teilwort } aba\}$

# Satz von Kleene

4

# Satz von Kleene

4

4. Vorlesung

Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

## Beweis (Organisation)

## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

### Beweis (Organisation)

- 1. Teil:** Wir konstruieren zu einem RA  $R$  einen NEA  $N$  mit  $L(R) = L(N)$

## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

### Beweis (Organisation)

- 1. Teil:** Wir konstruieren zu einem RA  $R$  einen NEA  $N$  mit  $L(R) = L(N)$   
das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \subseteq \text{REG}$

## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

### Beweis (Organisation)

- 1. Teil:** Wir konstruieren zu einem RA  $R$  einen NEA  $N$  mit  $L(R) = L(N)$   
das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \subseteq \text{REG}$
- 2. Teil:** Wir konstruieren zu einem DEA  $M$  einen RA  $R$  mit  $L(M) = L(R)$



## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

### Beweis (Organisation)

**1. Teil:** Wir konstruieren zu einem RA  $R$  einen NEA  $N$  mit  $L(R) = L(N)$

das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \subseteq \text{REG}$


**2. Teil:** Wir konstruieren zu einem DEA  $M$  einen RA  $R$  mit  $L(M) = L(R)$

das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \supseteq \text{REG}$

## Satz 5

$$\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} = \text{REG}$$

### Beweis (Organisation)

- 1. Teil:**  Wir konstruieren zu einem RA  $R$  einen NEA  $N$  mit  $L(R) = L(N)$   
das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \subseteq \text{REG}$
- 2. Teil:** **heute** Wir konstruieren zu einem DEA  $M$  einen RA  $R$  mit  $L(M) = L(R)$   
das zeigt  $\{L \mid \exists \text{ RA } R \text{ mit } L(R) = L\} \supseteq \text{REG}$

# Vom DEA zum RA

5

4. Vorlesung

# Vom DEA zum RA

5

**Ansatz** • Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.

# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$

# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



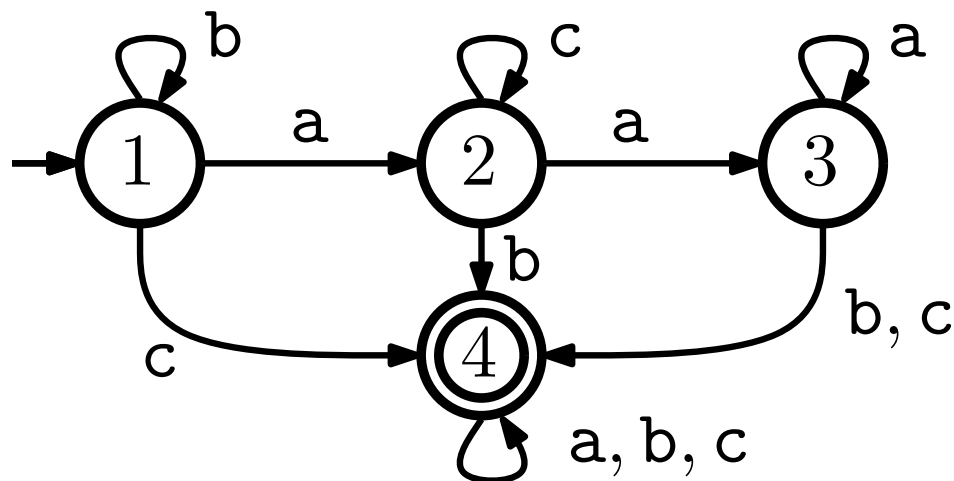
# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



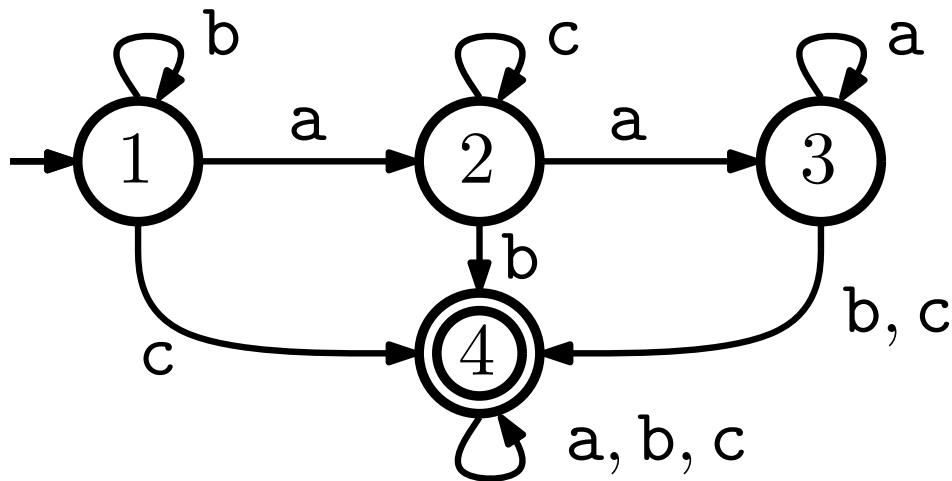
**Bsp**

# Vom DEA zum RA

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



**Bsp**  $R_{22}^0 =$

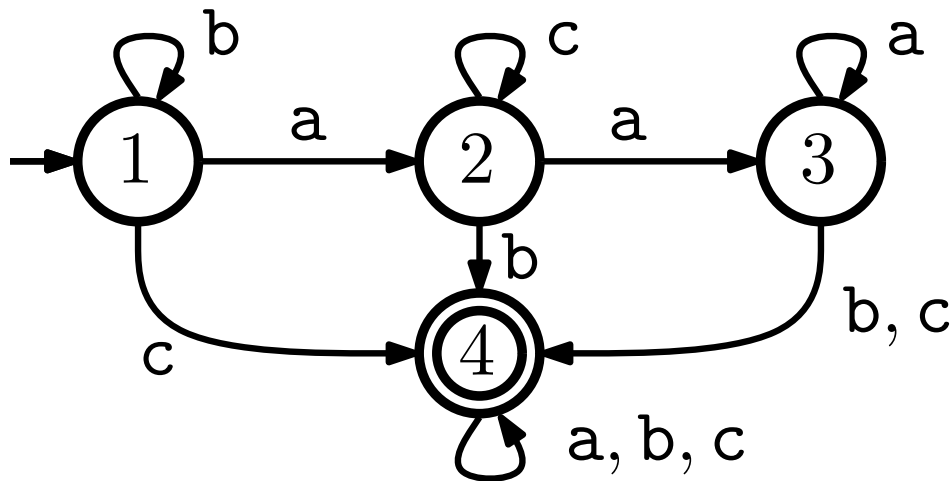
# Vom DEA zum RA

5

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



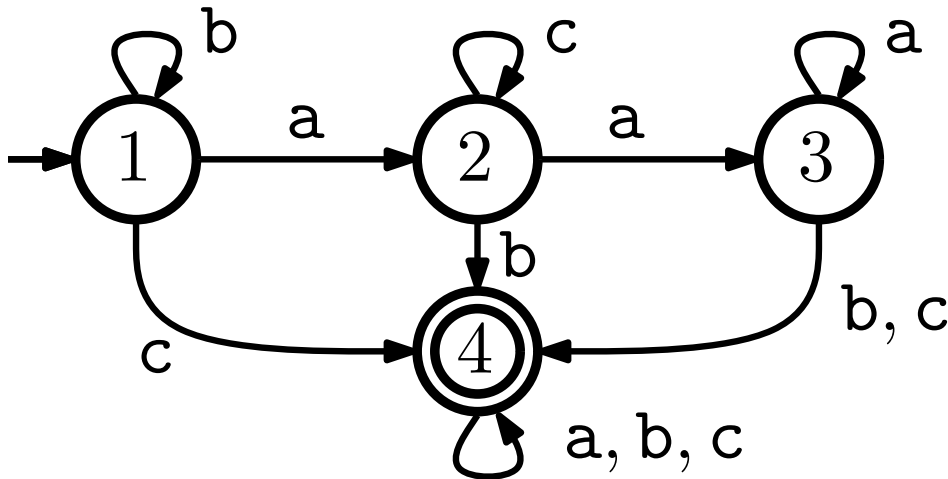
**Bsp**  $R_{22}^0 = c + \varepsilon$

# Vom DEA zum RA

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



**Bsp**  $R_{22}^0 = c + \varepsilon$

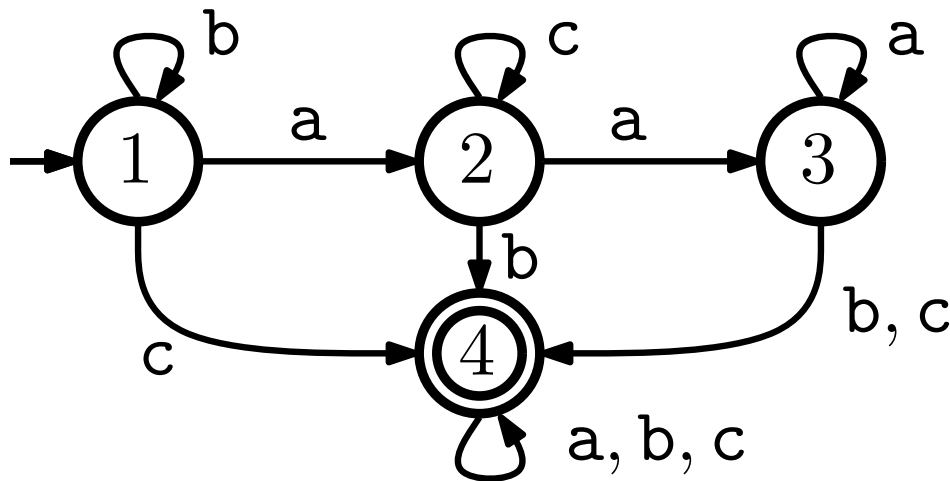
$R_{24}^2 =$

# Vom DEA zum RA

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



**Bsp**  $R_{22}^0 = c + \varepsilon$

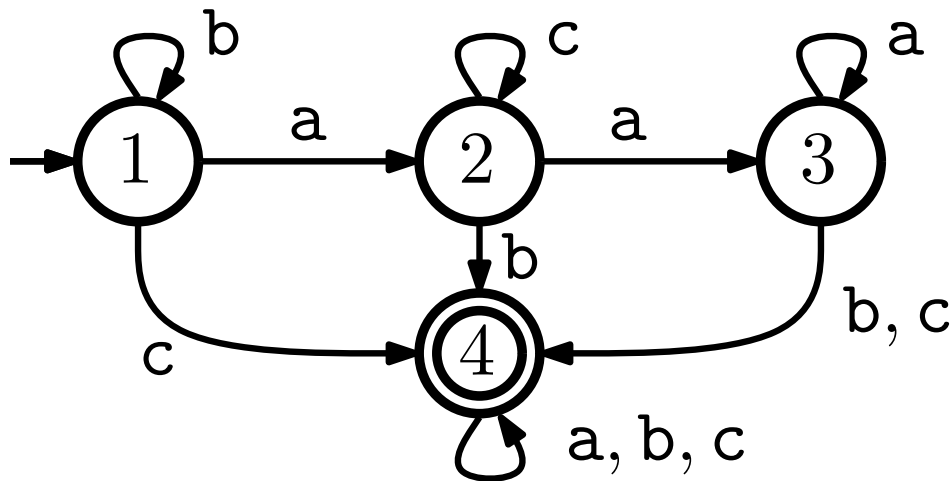
$R_{24}^2 = c^*b$

# Vom DEA zum RA

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



**Bsp**  $R_{22}^0 = c + \varepsilon$

$$R_{24}^2 = c^*b$$

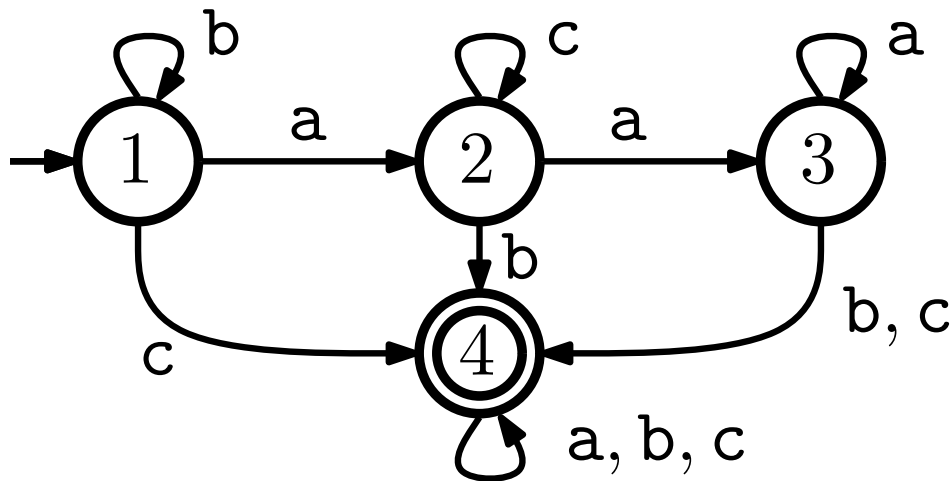
$$R_{24}^3 =$$

# Vom DEA zum RA

- Ansatz**
- Wenn  $L \in \text{REG}$ , dann existiert ein DEA  $M$  der  $L$  erkennt.
  - $M = (Q, \Sigma, \delta, 1, F)$  mit  $Q = \{1, 2, \dots, n\}$
  - **Ziel:** Konstruiere RA  $R$  mit  $L(R) = L$

$R_{ij}^k$  := RA für alle Wörter die von Zustand  $i$  nach  $j$  führen ohne einen Zustand  $> k$  zu benutzen.

Anfangs- und Endzustand  $i, j$  dürfen  $> k$  sein!



**Bsp**  $R_{22}^0 = c + \varepsilon$

$$R_{24}^2 = c^*b$$

$$R_{24}^3 = (c^*b + c^*aa^*(b + c))$$

# Von den Termen zum RA

6



# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

$\rightarrow w \in L(M)$  genau dann, wenn  $\delta^*(1, w) = j$  und  $j \in F$

# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

$\rightarrow w \in L(M)$  genau dann, wenn  $\underbrace{\delta^*(1, w)}_{w \in R_{1j}^n} = j$  und  $j \in F$

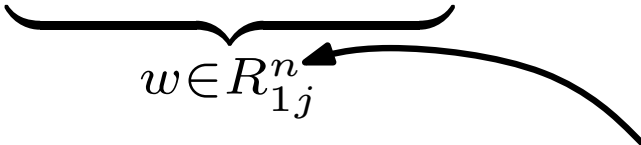
# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

$\rightarrow w \in L(M)$  genau dann, wenn  $\delta^*(1, w) = j$  und  $j \in F$

$w \in R_{1j}^n$



oberer Index  $n$  heißt keine Beschränkung

# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

→  $w \in L(M)$  genau dann, wenn  $\delta^*(1, w) = j$  und  $j \in F$

$w \in R_{1j}^n$

oberer Index  $n$  heißt keine Beschränkung

→ verbinde für alle akzeptierende Zustände die  $R_{ij}^k$  Terme

# Von den Termen zum RA

6

**Annahme:** Ich kenne alle Terme  $R_{ij}^k$

→  $w \in L(M)$  genau dann, wenn  $\delta^*(1, w) = j$  und  $j \in F$

$w \in R_{1j}^n$

oberer Index  $n$  heißt keine Beschränkung

→ verbinde für alle akzeptierende Zustände die  $R_{ij}^k$  Terme

$$R := R_{1f_1}^n + R_{1f_2}^n + \cdots + R_{1f_k}^n$$

$$\text{wobei } F = \{f_1, f_2, \dots, f_k\}$$

# Konstruktion der $R_{ij}^k$ Terme

7

# Konstruktion der $R_{ij}^k$ Terme

## Rekursive Konstruktion

7



# Konstruktion der $R_{ij}^k$ Terme

7

## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

# Konstruktion der $R_{ij}^k$ Terme

7

## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

# Konstruktion der $R_{ij}^k$ Terme

7

## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ):

# Konstruktion der $R_{ij}^k$ Terme

7

## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

# Konstruktion der $R_{ij}^k$ Terme

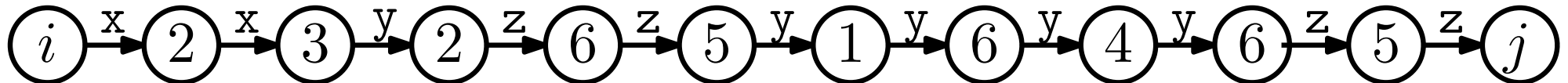
## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

**Bsp.** von einem "Lauf" aus  $R_{ij}^6$



# Konstruktion der $R_{ij}^k$ Terme

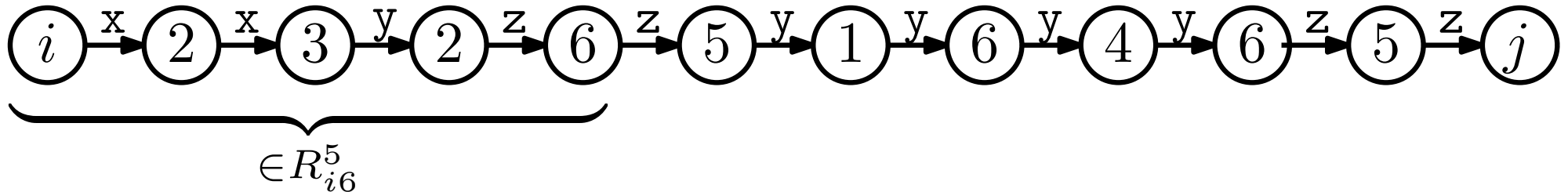
## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

**Bsp.** von einem "Lauf" aus  $R_{ij}^6$



# Konstruktion der $R_{ij}^k$ Terme

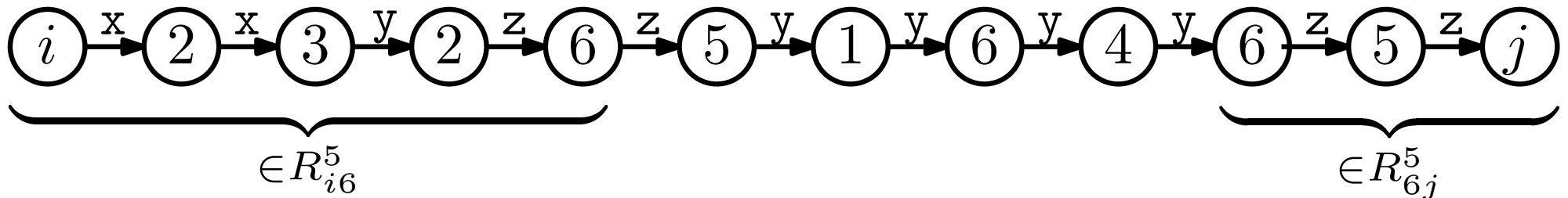
## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

**Bsp.** von einem "Lauf" aus  $R_{ij}^6$



# Konstruktion der $R_{ij}^k$ Terme

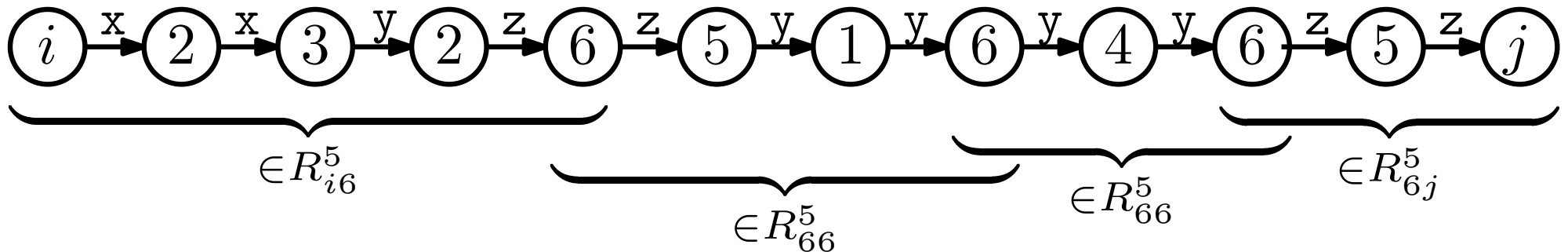
## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

**Bsp.** von einem "Lauf" aus  $R_{ij}^6$





# Konstruktion der $R_{ij}^k$ Terme

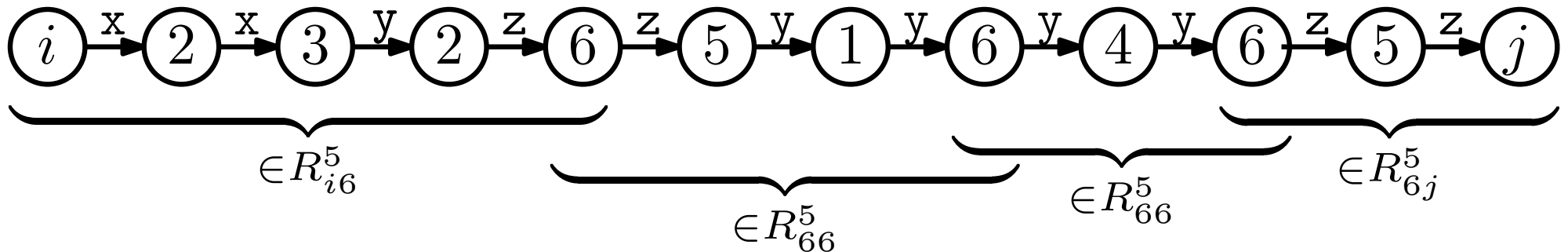
## Rekursive Konstruktion

**Basisfall** ( $R_{ij}^0$ ): nur direkte Übergänge

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

**Rekursion** ( $R_{ij}^k$ ): ich kenne bereits alle  $R_{ij}^{k-1}$

**Bsp.** von einem "Lauf" aus  $R_{ij}^6$



$\rightarrow R_{**}^k$  kann ich mit Termen  $R_{**}^{k-1}$  beschreiben

# Rekursion für $R_{ij}^k$

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$

$$\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$$

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- es gibt keine andere Möglichkeit, deshalb

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- es gibt keine andere Möglichkeit, deshalb

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$



# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- es gibt keine andere Möglichkeit, deshalb

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

Berechnung mit ansteigendem  $k$  ( $k = 0, \dots, n$ ) aller  $R_{ij}^k$  Terme

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- es gibt keine andere Möglichkeit, deshalb

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

Berechnung mit ansteigendem  $k$  ( $k = 0, \dots, n$ ) aller  $R_{ij}^k$  Terme □

# Rekursion für $R_{ij}^k$

- 1. Möglichkeit: der *Lauf* eines Wortes besucht **nicht** den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ij}^{k-1}$
- 2. Möglichkeit: der *Lauf* eines Wortes besucht einmal oder mehrmals den Zustand  $k$   
 $\rightarrow R_{ij}^k \supseteq R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$
- es gibt keine andere Möglichkeit, deshalb

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

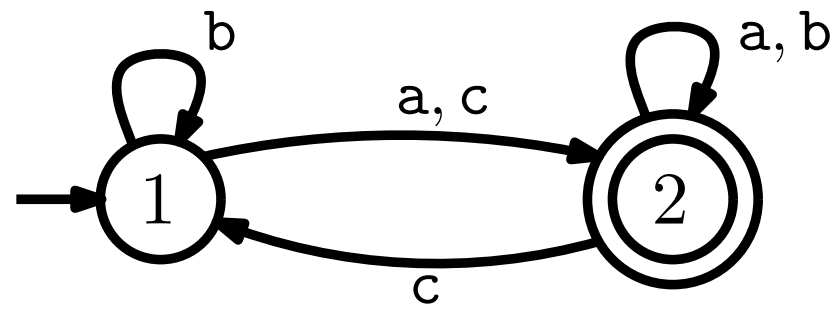
Berechnung mit ansteigendem  $k$  ( $k = 0, \dots, n$ ) aller  $R_{ij}^k$  Terme

## Dynamisches Programmieren

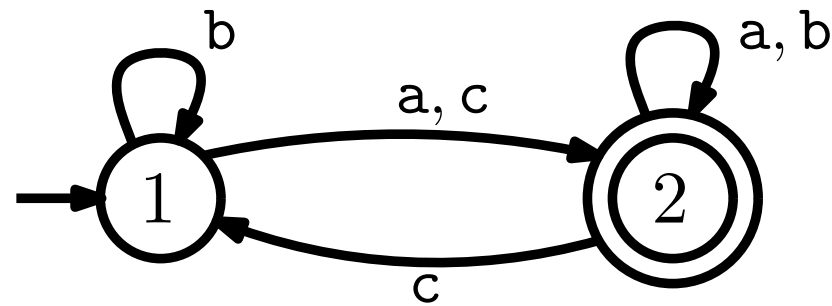


# Beispiel

# Beispiel



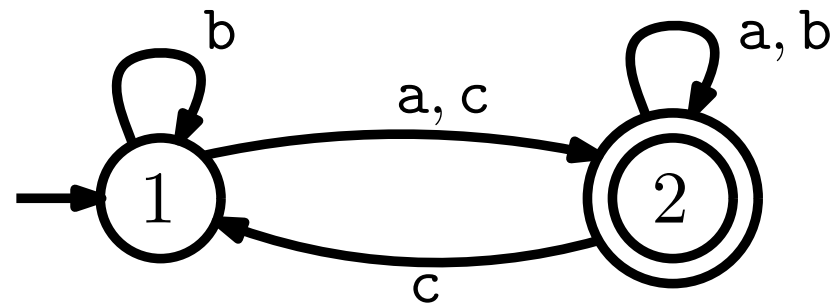
# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

# Beispiel

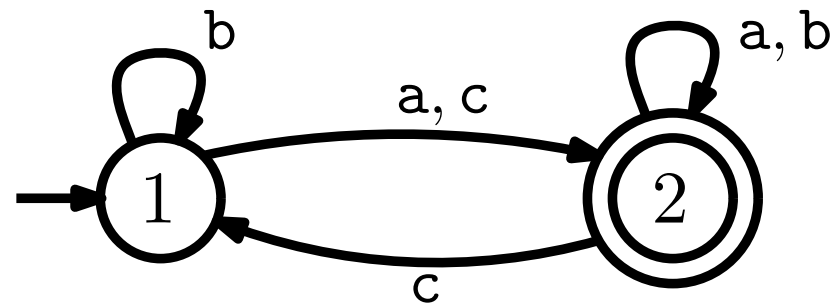


9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

# Beispiel



9

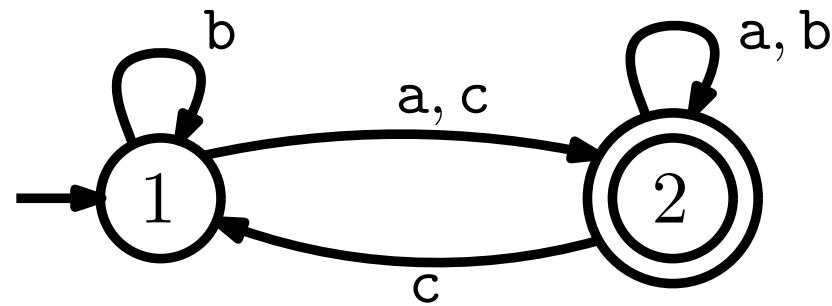
$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = b + \varepsilon \quad R_{12}^0 = a + c \quad R_{21}^0 = c \quad R_{22}^0 = (a + b + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$



# Beispiel



9

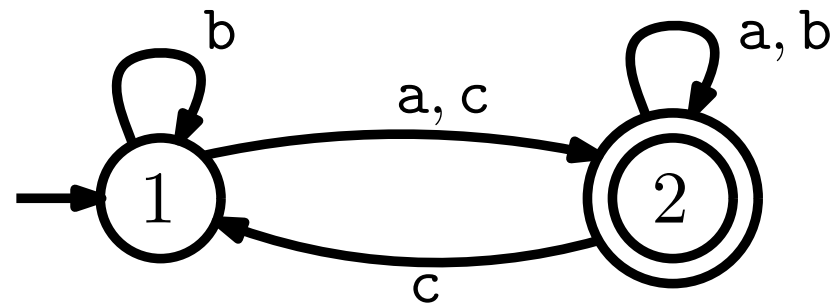
$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 =$$

# Beispiel



9

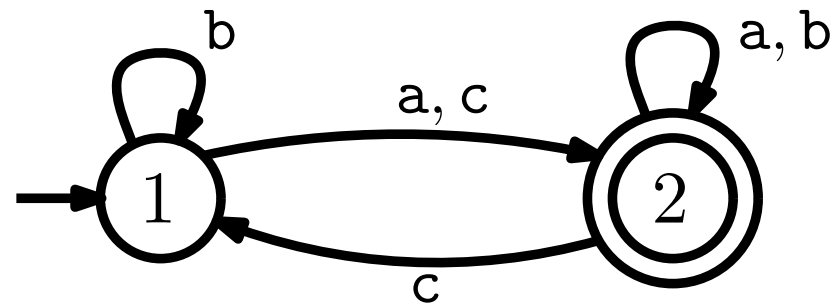
$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

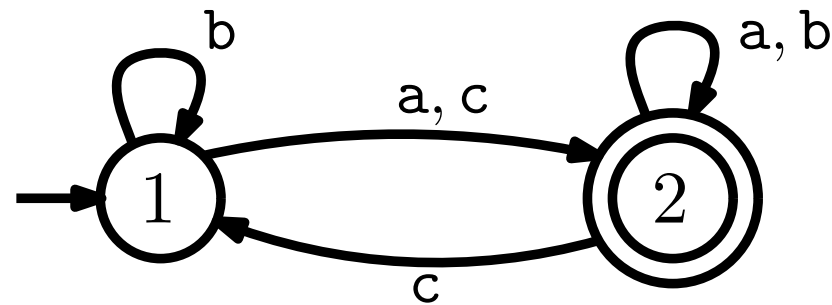
$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

$$R_{12}^1 =$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

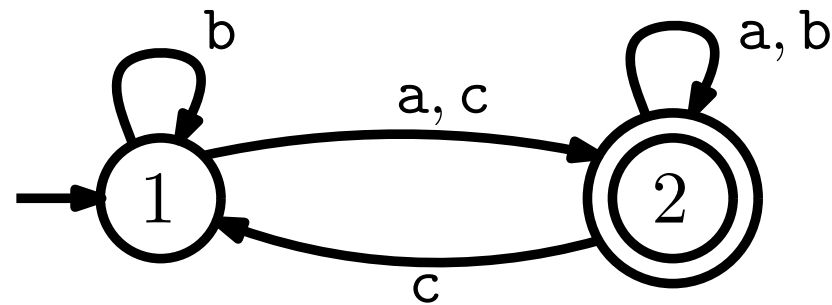
$$R_{11}^0 = b + \varepsilon \quad R_{12}^0 = a + c \quad R_{21}^0 = c \quad R_{22}^0 = (a + b + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (b + \varepsilon) + (b + \varepsilon)(b + \varepsilon)^*(b + \varepsilon) = b^*$$

$$R_{12}^1 = (a + c) + (b + \varepsilon)(b + \varepsilon)^*(a + c) = b^*(a + c)$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

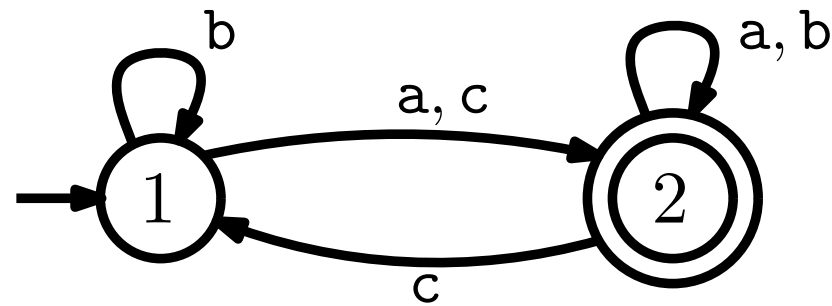
$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

$$R_{12}^1 = (\mathbf{a} + \mathbf{c}) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = \mathbf{b}^*(\mathbf{a} + \mathbf{c})$$

$$R_{21}^1 =$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = b + \varepsilon \quad R_{12}^0 = a + c \quad R_{21}^0 = c \quad R_{22}^0 = (a + b + \varepsilon)$$

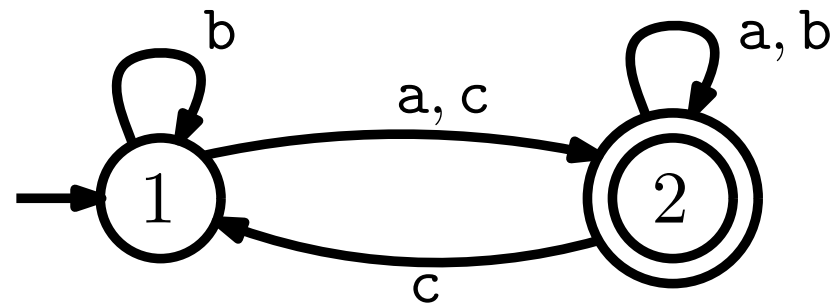
$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (b + \varepsilon) + (b + \varepsilon)(b + \varepsilon)^*(b + \varepsilon) = b^*$$

$$R_{12}^1 = (a + c) + (b + \varepsilon)(b + \varepsilon)^*(a + c) = b^*(a + c)$$

$$R_{21}^1 = c + c(b + \varepsilon)^*(b + \varepsilon) = cb^*$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

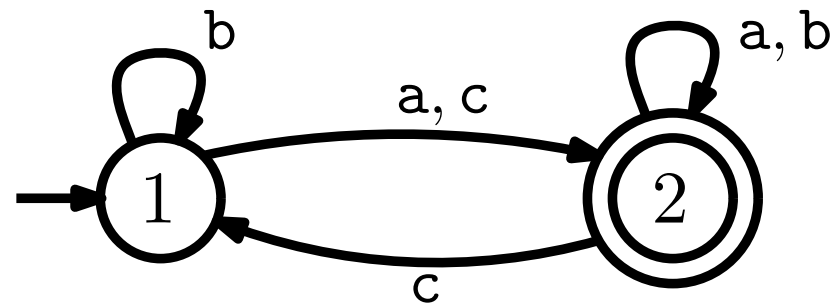
$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

$$R_{12}^1 = (\mathbf{a} + \mathbf{c}) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = \mathbf{b}^*(\mathbf{a} + \mathbf{c})$$

$$R_{21}^1 = \mathbf{c} + \mathbf{c}(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{cb}^*$$

$$R_{22}^1 =$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

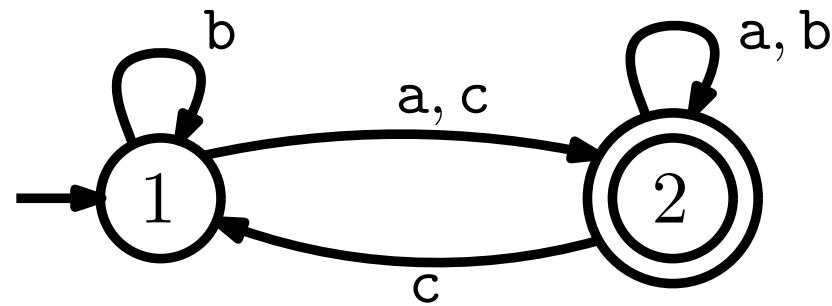
$$R_{12}^1 = (\mathbf{a} + \mathbf{c}) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = \mathbf{b}^*(\mathbf{a} + \mathbf{c})$$

$$R_{21}^1 = \mathbf{c} + \mathbf{c}(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{cb}^*$$

$$R_{22}^1 = (\mathbf{a} + \mathbf{b} + \varepsilon) + \mathbf{c}(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = (\mathbf{a} + \mathbf{b} + \varepsilon) + \mathbf{cb}^*(\mathbf{a} + \mathbf{c})$$



# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = \mathbf{b} + \varepsilon \quad R_{12}^0 = \mathbf{a} + \mathbf{c} \quad R_{21}^0 = \mathbf{c} \quad R_{22}^0 = (\mathbf{a} + \mathbf{b} + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (\mathbf{b} + \varepsilon) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{b}^*$$

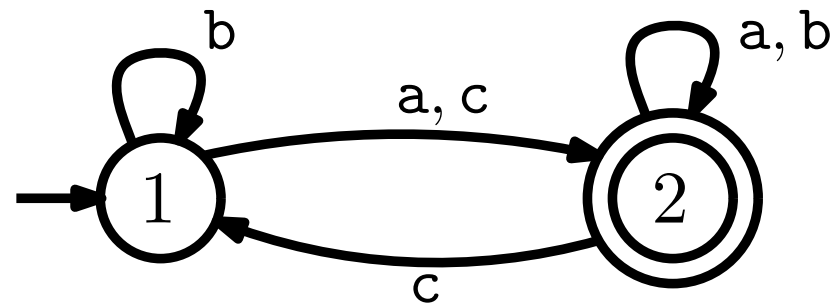
$$R_{12}^1 = (\mathbf{a} + \mathbf{c}) + (\mathbf{b} + \varepsilon)(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = \mathbf{b}^*(\mathbf{a} + \mathbf{c})$$

$$R_{21}^1 = \mathbf{c} + \mathbf{c}(\mathbf{b} + \varepsilon)^*(\mathbf{b} + \varepsilon) = \mathbf{cb}^*$$

$$R_{22}^1 = (\mathbf{a} + \mathbf{b} + \varepsilon) + \mathbf{c}(\mathbf{b} + \varepsilon)^*(\mathbf{a} + \mathbf{c}) = (\mathbf{a} + \mathbf{b} + \varepsilon) + \mathbf{cb}^*(\mathbf{a} + \mathbf{c})$$

$$R = R_{12}^2 =$$

# Beispiel



9

$$R_{ij}^0 := \begin{cases} \emptyset + a_1 + a_2 + \dots & \text{falls } i \neq j \text{ und } a_i \in \{a \mid \delta(i, a) = j\} \\ \varepsilon + a_1 + a_2 + \dots & \text{falls } i = j \text{ und } a_i \in \{a \mid \delta(i, a) = i\} \end{cases}$$

$$R_{11}^0 = b + \varepsilon \quad R_{12}^0 = a + c \quad R_{21}^0 = c \quad R_{22}^0 = (a + b + \varepsilon)$$

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{kj}^{k-1}$$

$$R_{11}^1 = (b + \varepsilon) + (b + \varepsilon)(b + \varepsilon)^*(b + \varepsilon) = b^*$$

$$R_{12}^1 = (a + c) + (b + \varepsilon)(b + \varepsilon)^*(a + c) = b^*(a + c)$$

$$R_{21}^1 = c + c(b + \varepsilon)^*(b + \varepsilon) = cb^*$$

$$R_{22}^1 = (a + b + \varepsilon) + c(b + \varepsilon)^*(a + c) = (a + b + \varepsilon) + cb^*(a + c)$$

$$R = R_{12}^2 =$$

$$b^*(a + c) + b^*(a + c) [(a + b + \varepsilon) + cb^*(a + c)]^* [(a + b + \varepsilon) + cb^*(a + c)]$$

# Fazit

10

## Reguläre Sprachen

**Reguläre Sprachen** vom **DEA** erkannt (nach Def.)



**Reguläre Sprachen**

vom **DEA** erkannt (nach Def.)

äquivalent zum **NEA**

Berechnungsmodell mit

- Nichtdeterminismus
- $\varepsilon$ -Übergängen

**Reguläre Sprachen**

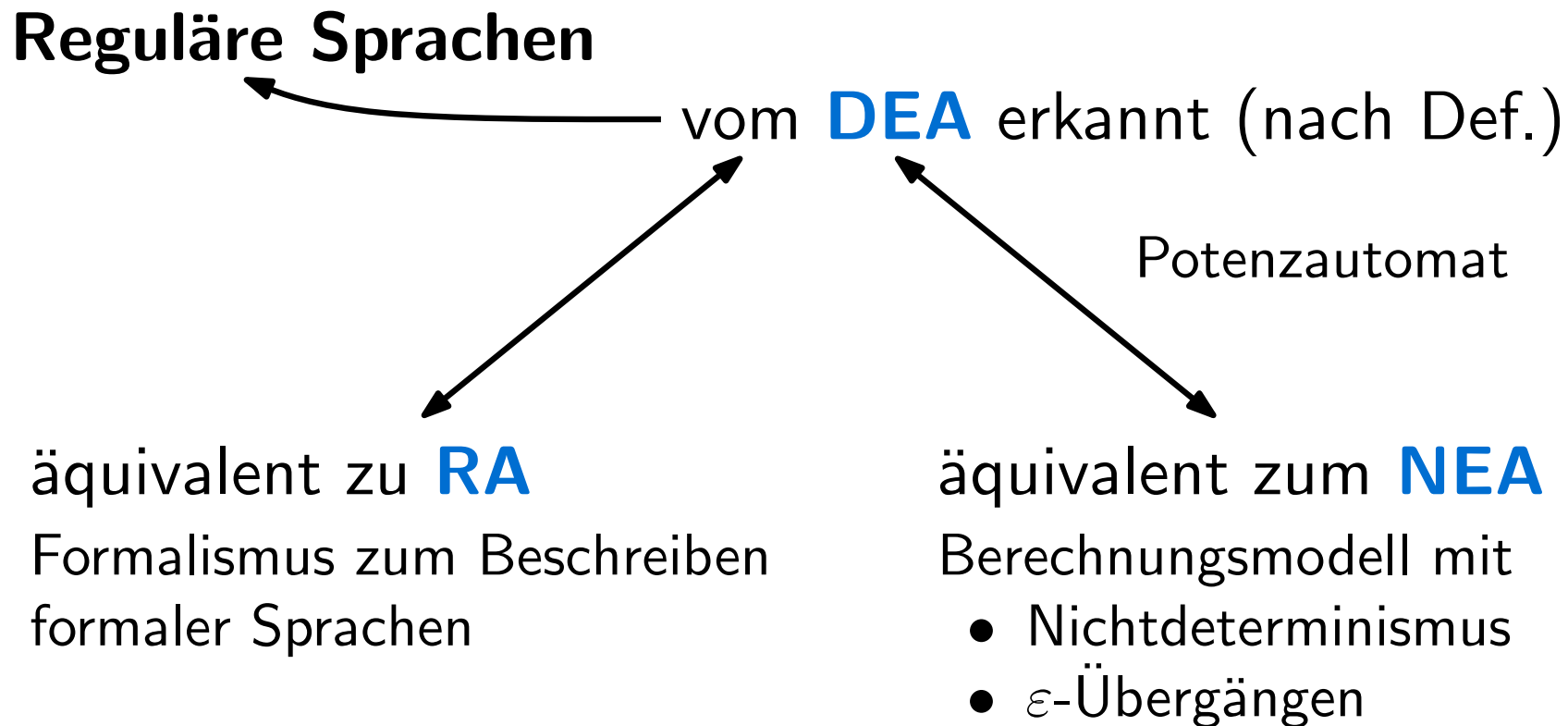
vom **DEA** erkannt (nach Def.)

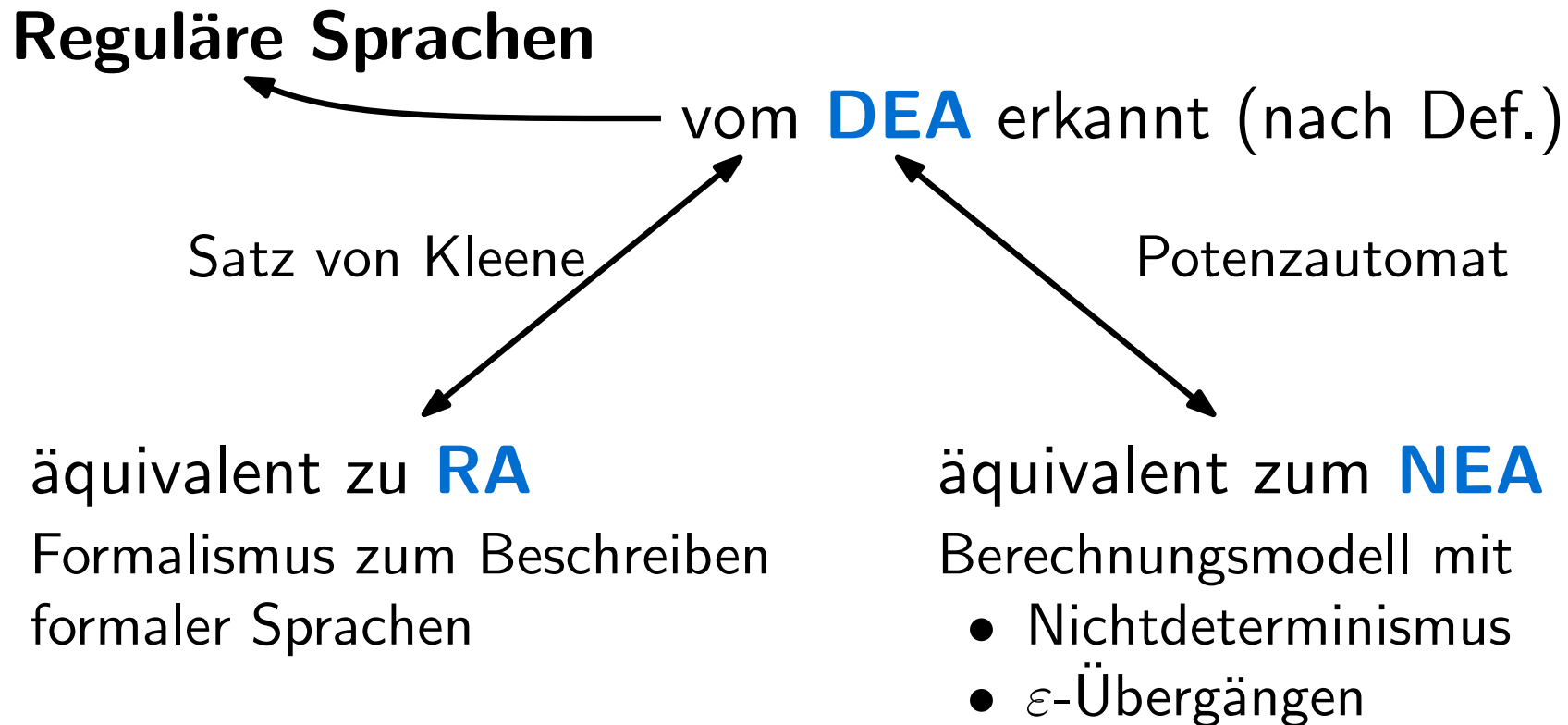
Potenzautomat

äquivalent zum **NEA**

Berechnungsmodell mit

- Nichtdeterminismus
- $\varepsilon$ -Übergängen







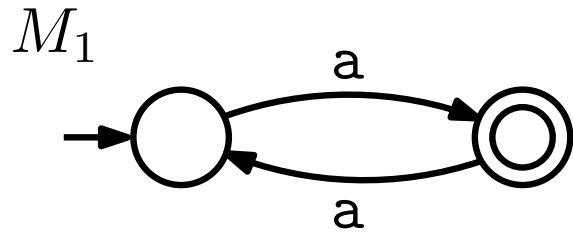
# Minimierung von DEAs

11

4. Vorlesung

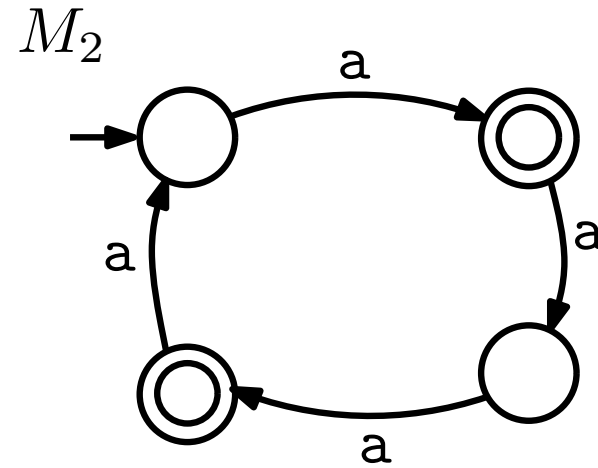
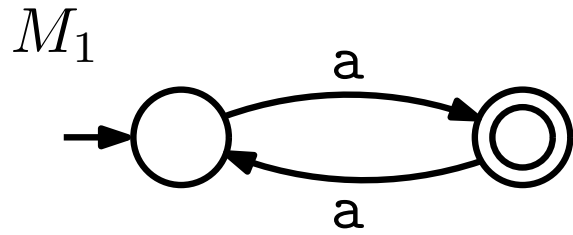
# Minimierung von DEAs

11



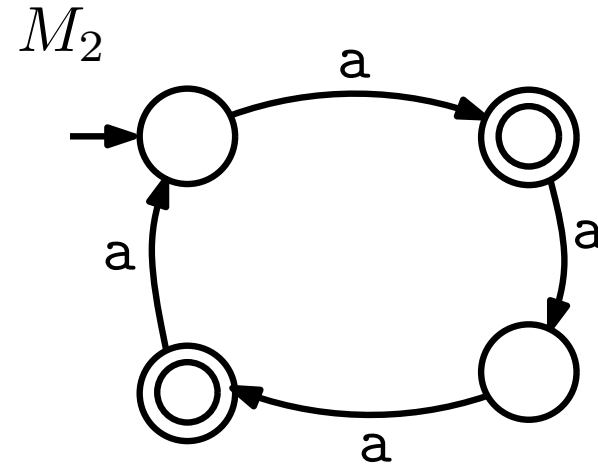
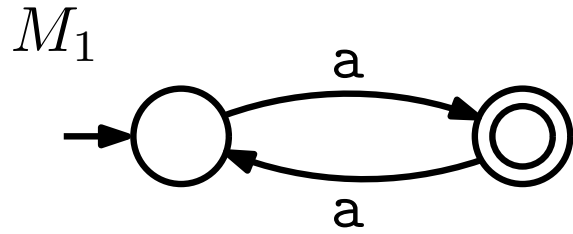
# Minimierung von DEAs

11



# Minimierung von DEAs

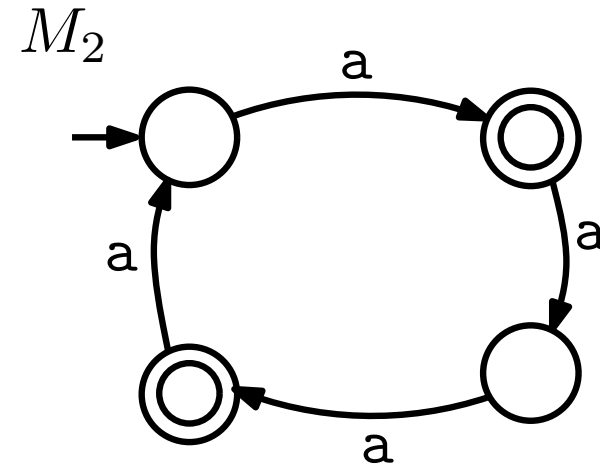
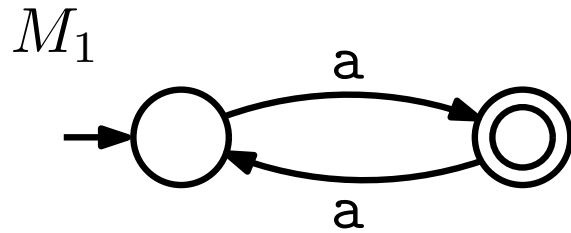
11



$$L(M_1) = \{a^k \mid k \text{ ist ungerade}\}$$

# Minimierung von DEAs

11

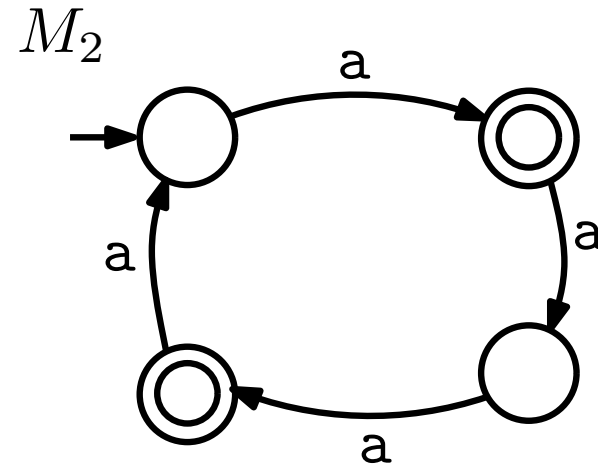
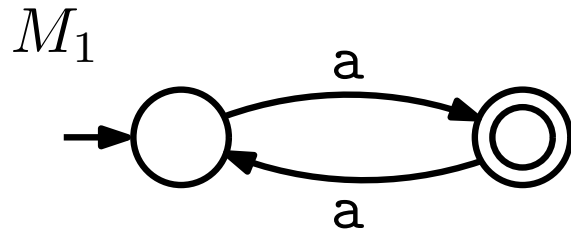


$$L(M_1) = \{a^k \mid k \text{ ist ungerade}\}$$

$$L(M_2) = \{a^k \mid k \text{ ist ungerade}\}$$

# Minimierung von DEAs

11



$$L(M_1) = \{a^k \mid k \text{ ist ungerade}\}$$

$$L(M_2) = \{a^k \mid k \text{ ist ungerade}\}$$

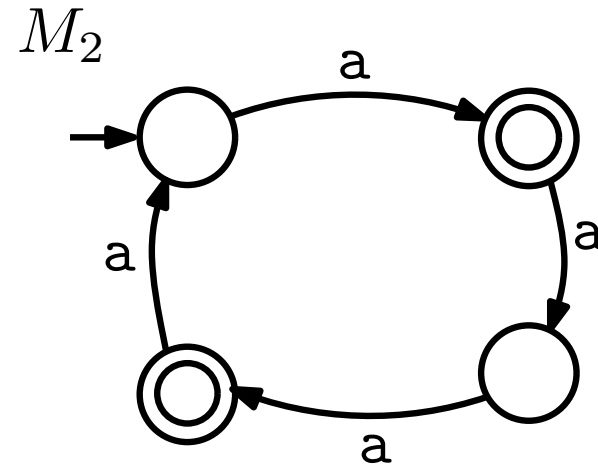
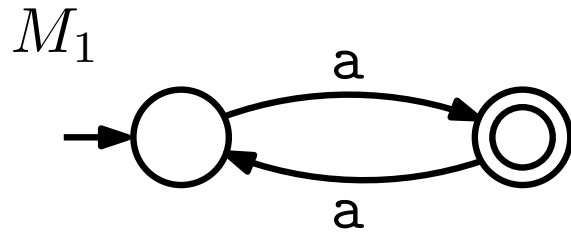
## Definition

Zwei DEAs  $M_1$  und  $M_2$  heißen **äquivalent**, gdw.

$$L(M_1) = L(M_2)$$

# Minimierung von DEAs

11



$$L(M_1) = \{a^k \mid k \text{ ist ungerade}\}$$

$$L(M_2) = \{a^k \mid k \text{ ist ungerade}\}$$

## Definition

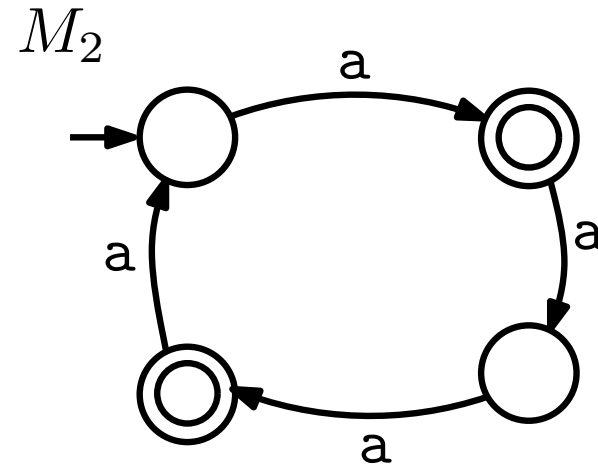
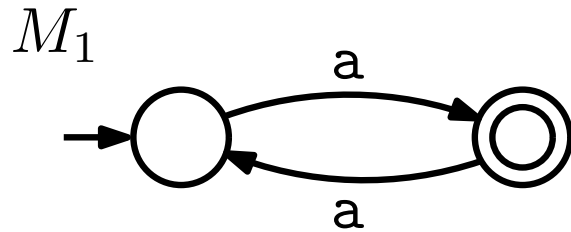
Zwei DEAs  $M_1$  und  $M_2$  heißen **äquivalent**, gdw.

$$L(M_1) = L(M_2)$$

**Frage:** Welches ist der kleinste DEA für eine Sprache  $L$ ?

# Minimierung von DEAs

11



$$L(M_1) = \{a^k \mid k \text{ ist ungerade}\}$$

$$L(M_2) = \{a^k \mid k \text{ ist ungerade}\}$$

## Definition

Zwei DEAs  $M_1$  und  $M_2$  heißen **äquivalent**, gdw.

$$L(M_1) = L(M_2)$$

**Frage:** Welches ist der kleinste DEA für eine Sprache  $L$ ?

minimale Anzahl von Zuständen



# Äquivalenz von Zuständen

12

4. Vorlesung

## Definition

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein DEA, dann heißen zwei Zustände  $p, q \in Q$  **äquivalent** (Schreibweise  $p \approx q$ ) gdw.

$$\forall z \in \Sigma^* : \quad \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$$

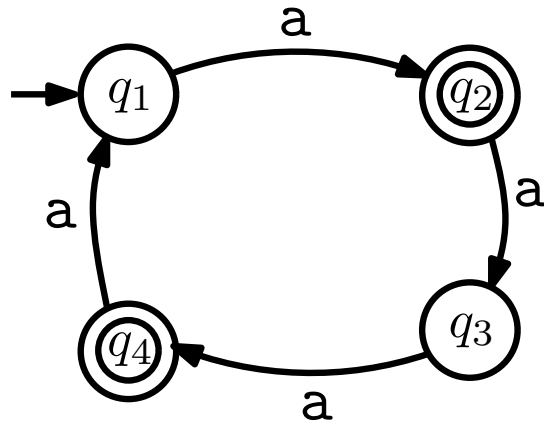
## Definition

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein DEA, dann heißen zwei Zustände  $p, q \in Q$  **äquivalent** (Schreibweise  $p \approx q$ ) gdw.

$$\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$$

Zwei nicht äquivalente Zustände nennen wir auch **trennbar**.

**Bsp.**



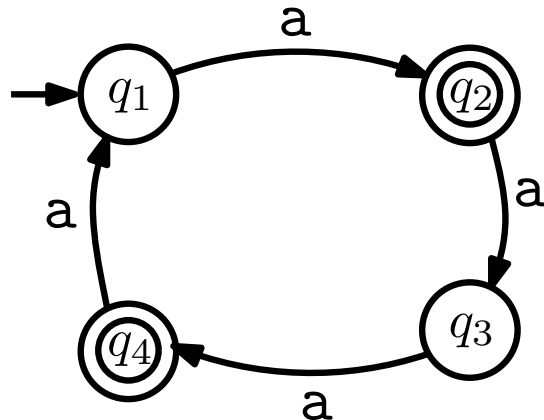
## Definition

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein DEA, dann heißen zwei Zustände  $p, q \in Q$  **äquivalent** (Schreibweise  $p \approx q$ ) gdw.

$$\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$$

Zwei nicht äquivalente Zustände nennen wir auch **trennbar**.

**Bsp.**



- $q_1$  und  $q_2$  sind trennbar (z.B. durch das **Trennwort** a)

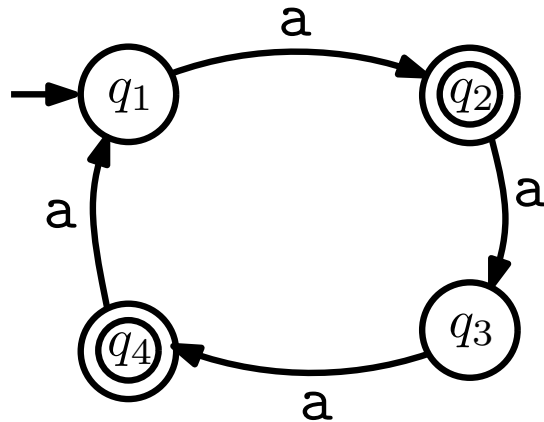
## Definition

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein DEA, dann heißen zwei Zustände  $p, q \in Q$  **äquivalent** (Schreibweise  $p \approx q$ ) gdw.

$$\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$$

Zwei nicht äquivalente Zustände nennen wir auch **trennbar**.

**Bsp.**



- $q_1$  und  $q_2$  sind trennbar (z.B. durch das **Trennwort** a)
- $q_1$  und  $q_3$  sind äquivalent

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$



## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

Transitivität

$$\left[ \begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(r, z) \in F \end{array} \right] \Rightarrow \delta^*(p, z) \in F \Leftrightarrow \delta^*(r, z) \in F$$

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

Transitivität

$$\left[ \begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(r, z) \in F \end{array} \right] \Rightarrow \delta^*(p, z) \in F \Leftrightarrow \delta^*(r, z) \in F \quad \square$$

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

Transitivität

$$\left[ \begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(r, z) \in F \end{array} \right] \Rightarrow \delta^*(p, z) \in F \Leftrightarrow \delta^*(r, z) \in F \quad \square$$

→ die Zustandsmenge  $Q$  zerfällt in Äquivalenzklassen

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

Transitivität

$$\left[ \begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(r, z) \in F \end{array} \right] \Rightarrow \delta^*(p, z) \in F \Leftrightarrow \delta^*(r, z) \in F \quad \square$$

→ die Zustandsmenge  $Q$  zerfällt in Äquivalenzklassen

→ Äquivalenzklasse die  $q \in Q$  enthält  $[q] := \{p \in Q \mid p \approx q\}$

## Lemma 2

Die Relation  $\approx$  ist eine Äquivalenzrelation.

Erinnerung:  $p \approx q$  wenn  $\forall z \in \Sigma^* : \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F$

Symmetrie

$$\begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(p, z) \in F \end{array}$$

Reflexivität

$$\forall p \in Q : \delta^*(p, z) \in F \Leftrightarrow \delta^*(p, z) \in F$$

Transitivität

$$\left[ \begin{array}{l} \delta^*(p, z) \in F \Leftrightarrow \delta^*(q, z) \in F \\ \delta^*(q, z) \in F \Leftrightarrow \delta^*(r, z) \in F \end{array} \right] \Rightarrow \delta^*(p, z) \in F \Leftrightarrow \delta^*(r, z) \in F \quad \square$$

→ die Zustandsmenge  $Q$  zerfällt in Äquivalenzklassen

→ Äquivalenzklasse die  $q \in Q$  enthält  $[q] := \{p \in Q \mid p \approx q\}$

Für zwei Zustände  $p, q$  aus einer Klasse gilt

1.  $p \in F \Leftrightarrow q \in F$ ,
2.  $\forall a \in \Sigma$ , dass  $\delta(p, a) \approx \delta(q, a)$ .

# Der kollabierte Automat

14

4. Vorlesung

## Definition

Der **kollabierte Automat**  $M' = (Q', \Sigma, \delta', q'_0, F')$  ist ein DEA mit

- $Q' = \{[q] \mid q \in Q\}$ ,
- $\delta'([q], a) = [\delta(q, a)]$ ,
- $q'_0 = [q_0]$ ,
- $F' = \{[q] \mid q \in F\}$ .

Außerdem werden alle nicht erreichbaren Zustände gestrichen.



## Definition

Der **kollabierte Automat**  $M' = (Q', \Sigma, \delta', q'_0, F')$  ist ein DEA mit

- $Q' = \{[q] \mid q \in Q\}$ ,
- $\delta'([q], a) = [\delta(q, a)]$ ,
- $q'_0 = [q_0]$ ,
- $F' = \{[q] \mid q \in F\}$ .

Außerdem werden alle nicht erreichbaren Zustände gestrichen.

Definition ist korrekt, denn

## Definition

Der **kollabierte Automat**  $M' = (Q', \Sigma, \delta', q'_0, F')$  ist ein DEA mit

- $Q' = \{[q] \mid q \in Q\}$ ,
- $\delta'([q], a) = [\delta(q, a)]$ ,
- $q'_0 = [q_0]$ ,
- $F' = \{[q] \mid q \in F\}$ .

Außerdem werden alle nicht erreichbaren Zustände gestrichen.

Definition ist korrekt, denn

- Alle Elemente aus  $[q]$  beschreiben den gleichen Folgezustand bzgl.  $\delta'$   
$$p \approx q \quad \Rightarrow \quad \delta(p, a) \approx \delta(q, a)$$

## Definition

Der **kollabierte Automat**  $M' = (Q', \Sigma, \delta', q'_0, F')$  ist ein DEA mit

- $Q' = \{[q] \mid q \in Q\}$ ,
- $\delta'([q], a) = [\delta(q, a)]$ ,
- $q'_0 = [q_0]$ ,
- $F' = \{[q] \mid q \in F\}$ .

Außerdem werden alle nicht erreichbaren Zustände gestrichen.

Definition ist korrekt, denn

- Alle Elemente aus  $[q]$  beschreiben den gleichen Folgezustand bzgl.  $\delta'$

$$p \approx q \quad \Rightarrow \quad \delta(p, a) \approx \delta(q, a)$$

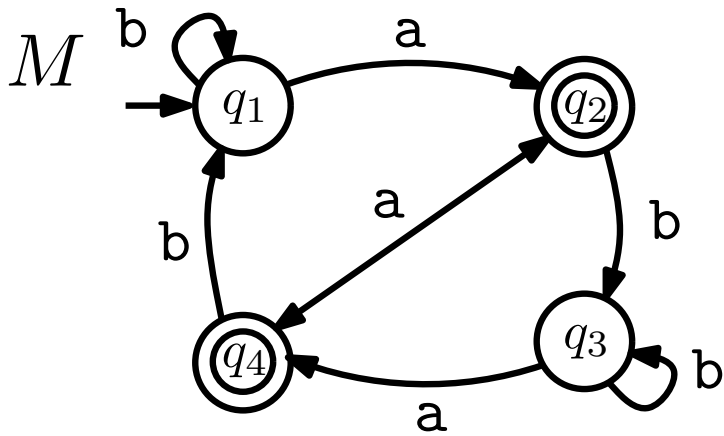
- Akzeptierte Klassen sind wohldefiniert

$$p_1, p_2 \in [q] \quad \Rightarrow \quad \delta(p_1, \varepsilon) \in F \Leftrightarrow \delta(p_2, \varepsilon) \in F$$

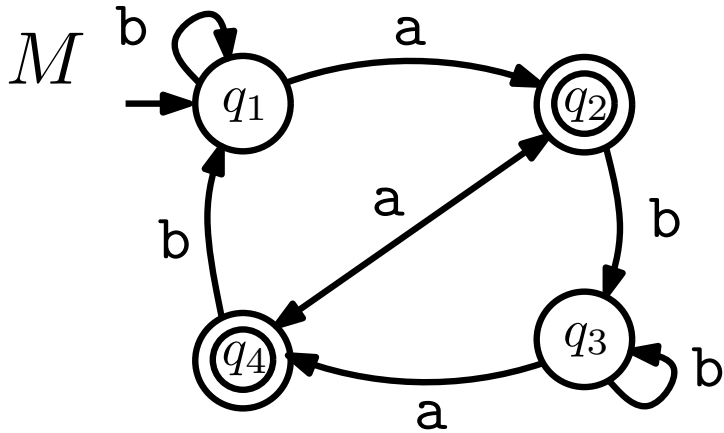
$$\Rightarrow \quad p_1 \in F \Leftrightarrow p_2 \in F$$

# Beispiel: Der kollabierte Automat <sup>15</sup>

# Beispiel: Der kollabierte Automat <sup>15</sup>



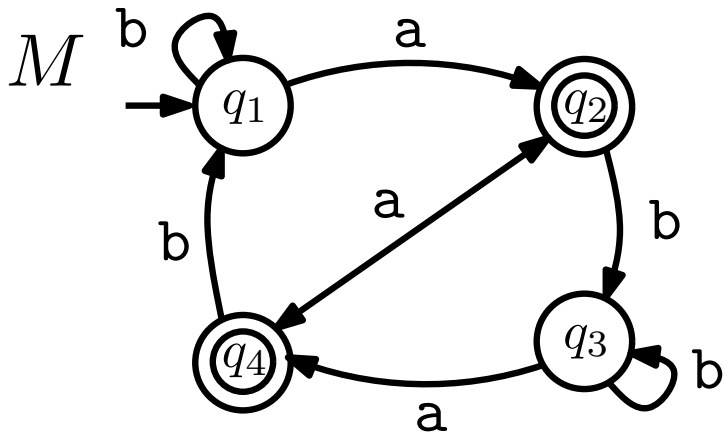
# Beispiel: Der kollabierte Automat <sup>15</sup>



Alle Zustandspaare trennbar bis auf

- $q_1 \approx q_3$ ,
- $q_2 \approx q_4$ .

# Beispiel: Der kollabierte Automat <sup>15</sup>

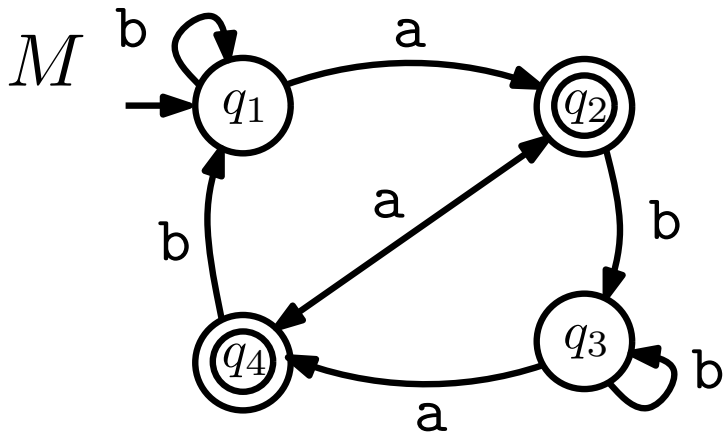


Alle Zustandspaare trennbar bis auf

- $q_1 \approx q_3$ ,
- $q_2 \approx q_4$ .

- zwei Klassen:  $[q_1] = \{q_1, q_3\}$  und  $[q_2] = \{q_2, q_4\}$ , d.h.  
 $Q' = \{[q_1], [q_2]\}$

# Beispiel: Der kollabierte Automat <sup>15</sup>



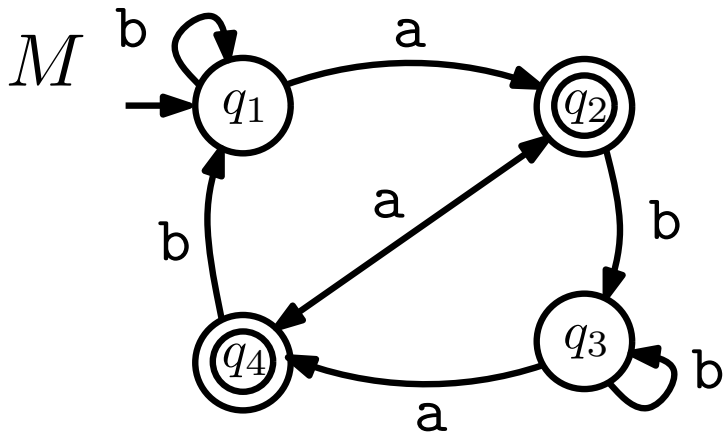
Alle Zustandspaare trennbar bis auf

- $q_1 \approx q_3$ ,
- $q_2 \approx q_4$ .

- zwei Klassen:  $[q_1] = \{q_1, q_3\}$  und  $[q_2] = \{q_2, q_4\}$ , d.h.  
 $Q' = \{[q_1], [q_2]\}$
- nur  $[q_2]$  enthält akz. Zustände, deshalb  $F' = \{[q_2]\}$



# Beispiel: Der kollabierte Automat <sup>15</sup>

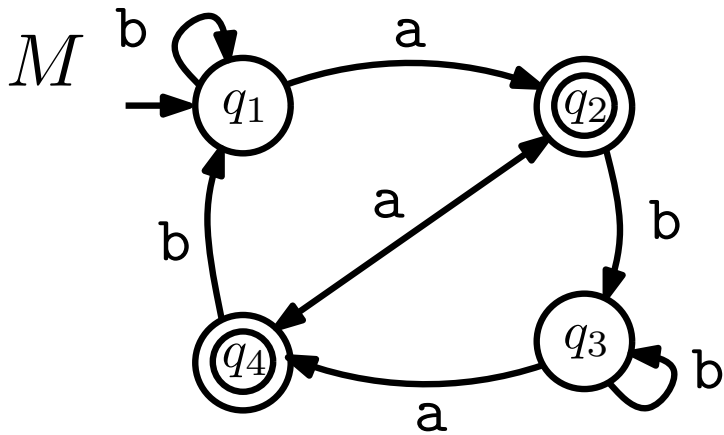


Alle Zustandspaare trennbar bis auf

- $q_1 \approx q_3$ ,
- $q_2 \approx q_4$ .

- zwei Klassen:  $[q_1] = \{q_1, q_3\}$  und  $[q_2] = \{q_2, q_4\}$ , d.h.  $Q' = \{[q_1], [q_2]\}$
- nur  $[q_2]$  enthält akz. Zustände, deshalb  $F' = \{[q_2]\}$
- $[q_1]$  enthält Startzustand, deshalb  $q'_0 = [q_1]$

# Beispiel: Der kollabierte Automat <sup>15</sup>

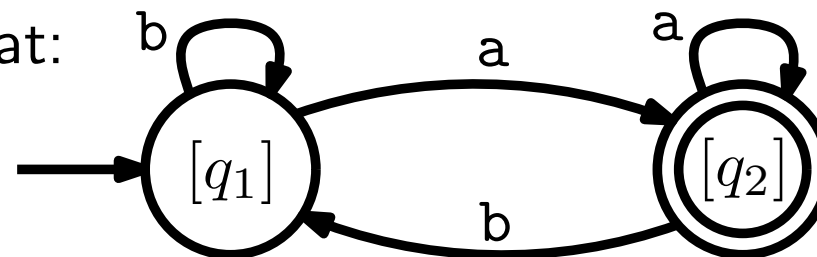


Alle Zustandspaare trennbar bis auf

- $q_1 \approx q_3$ ,
- $q_2 \approx q_4$ .

- zwei Klassen:  $[q_1] = \{q_1, q_3\}$  und  $[q_2] = \{q_2, q_4\}$ , d.h.  $Q' = \{[q_1], [q_2]\}$
- nur  $[q_2]$  enthält akz. Zustände, deshalb  $F' = \{[q_2]\}$
- $[q_1]$  enthält Startzustand, deshalb  $q'_0 = [q_1]$

Kollabierter Automat:



## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

**Beweis**

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

Zur Erinnerung

1.  $q_0 = s_1$ ,
2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

Zur Erinnerung

1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn



## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

Zur Erinnerung

1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn
    1.  $q'_0 = [q_0] = [s_1]$ ,
    2.  $\delta'([s_i], x_i) = [\delta(s_i, x_i)] = [s_{i+1}]$  für alle  $1 \leq i \leq k$ .

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

Zur Erinnerung

1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn
    1.  $q'_0 = [q_0] = [s_1]$ ,
    2.  $\delta'([s_i], x_i) = [\delta(s_i, x_i)] = [s_{i+1}]$  für alle  $1 \leq i \leq k$ .
  - $[s_{k+1}] \in F' \iff s_{k+1} \in F$

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
  - sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )
- Zur Erinnerung
1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn
    1.  $q'_0 = [q_0] = [s_1]$ ,
    2.  $\delta'([s_i], x_i) = [\delta(s_i, x_i)] = [s_{i+1}]$  für alle  $1 \leq i \leq k$ .
  - $[s_{k+1}] \in F' \iff s_{k+1} \in F$
  - $w$ -Lauf akzeptierend in  $M$  gdw.  $w$ -Lauf akzeptierend in  $M'$

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
  - sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )
- Zur Erinnerung
1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn
    1.  $q'_0 = [q_0] = [s_1]$ ,
    2.  $\delta'([s_i], x_i) = [\delta(s_i, x_i)] = [s_{i+1}]$  für alle  $1 \leq i \leq k$ .
  - $[s_{k+1}] \in F' \iff s_{k+1} \in F$
  - $w$ -Lauf akzeptierend in  $M$  gdw.  $w$ -Lauf akzeptierend in  $M'$
  - $w \in L(M) \iff w \in L(M')$

## Lemma 2

Sei  $M'$  der kollabierte Automat von  $M$ , dann gilt

$$L(M) = L(M').$$

### Beweis

- wir betrachten ein  $w \in \Sigma^*$ ,  $w = x_1x_2 \cdots x_k$ , mit  $x_i \in \Sigma$
- sei  $(s_1, s_2, \dots, s_{k+1})$  der  $w$ -Lauf in  $M$  ( $s_i \in Q$ )

Zur Erinnerung

1.  $q_0 = s_1$ ,
  2.  $\delta(s_i, x_i) = s_{i+1}$  für alle  $1 \leq i \leq k$ .
- $([s_1], [s_2], \dots, [s_{k+1}])$  ist der  $w$ -Lauf in  $M'$ , denn
    1.  $q'_0 = [q_0] = [s_1]$ ,
    2.  $\delta'([s_i], x_i) = [\delta(s_i, x_i)] = [s_{i+1}]$  für alle  $1 \leq i \leq k$ .
  - $[s_{k+1}] \in F' \iff s_{k+1} \in F$
  - $w$ -Lauf akzeptierend in  $M$  gdw.  $w$ -Lauf akzeptierend in  $M'$
  - $w \in L(M) \iff w \in L(M')$



# Table-Filling Algorithmus

17

4. Vorlesung

# Table-Filling Algorithmus

17

- Algorithmus zum effizienten Finden aller äquivalenten Zustände

# Table-Filling Algorithmus

17

- Algorithmus zum effizienten Finden aller äquivalenten Zustände
- Datenstruktur: Tabelle  $T$ , mit  $|Q|$  Zeilen und Spalten,  $Q = \{1, 2, 3, \dots, n\}$ .



- Algorithmus zum effizienten Finden aller äquivalenten Zustände
- Datenstruktur: Tabelle  $T$ , mit  $|Q|$  Zeilen und Spalten,  $Q = \{1, 2, 3, \dots, n\}$ .
- **Invariante:** Enthält  $T[p, q]$  die Markierung **1** dann sind  $p$  und  $q$  trennbar ( $p \neq q$ )

- Algorithmus zum effizienten Finden aller äquivalenten Zustände
- Datenstruktur: Tabelle  $T$ , mit  $|Q|$  Zeilen und Spalten,  $Q = \{1, 2, 3, \dots, n\}$ .
- **Invariante:** Enthält  $T[p, q]$  die Markierung **1** dann sind  $p$  und  $q$  trennbar ( $p \neq q$ )
- Tabelle  $T$  wird nach und nach mit 1en gefüllt bis eine Abbruchbedingung eintritt

- Algorithmus zum effizienten Finden aller äquivalenten Zustände
- Datenstruktur: Tabelle  $T$ , mit  $|Q|$  Zeilen und Spalten,  $Q = \{1, 2, 3, \dots, n\}$ .
- **Invariante:** Enthält  $T[p, q]$  die Markierung **1** dann sind  $p$  und  $q$  trennbar ( $p \neq q$ )
- Tabelle  $T$  wird nach und nach mit 1en gefüllt bis eine Abbruchbedingung eintritt
- am Ende notieren alle unmarkierten Einträge  $T[p, q]$  äquivalente Zustandspaare  $p, q$

# Table-Filling Algorithmus

18

4. Vorlesung

# Table-Filling Algorithmus

18

- Wir füllen nur die Hälfte der Tabelle  $T$  aus ( $T[p, q]$  mit  $p < q$ )

- Wir füllen nur die Hälfte der Tabelle  $T$  aus ( $T[p, q]$  mit  $p < q$ )

---

## Algorithm 1: TableFilling Algorithmus

---

```
1 Initialisiere  $T \equiv 0$ ;  
2 for all  $(p, q) \in Q \times Q$  do  
3   | if  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$  then  
4   |    $T[p, q] = 1$ ;  
5 end  
6 repeat  
7   | for all  $(p, q) \in Q \times Q$  mit  $T[p, q] \neq 1$  do  
8   |   | if  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  then  $T[p, q] = 1$ ;  
9   |   end  
10 until keine neue Markierung gesetzt;
```

---

- Wir füllen nur die Hälfte der Tabelle  $T$  aus ( $T[p, q]$  mit  $p < q$ )

---

## Algorithm 1: TableFilling Algorithmus

---

```
1 Initialisiere  $T \equiv 0$ ; Initialisierung
2 for all  $(p, q) \in Q \times Q$  do
3   | if  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$  then
4   |    $T[p, q] = 1$ ;
5 end
6 repeat
7   | for all  $(p, q) \in Q \times Q$  mit  $T[p, q] \neq 1$  do
8   |   | if  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  then  $T[p, q] = 1$ ;
9   |   end
9 until keine neue Markierung gesetzt;
```

---

# Table-Filling Algorithmus

18

- Wir füllen nur die Hälfte der Tabelle  $T$  aus ( $T[p, q]$  mit  $p < q$ )

---

## Algorithm 1: TableFilling Algorithmus

---

```
1 Initialisiere  $T \equiv 0$ ; Initialisierung
2 for all  $(p, q) \in Q \times Q$  do
3   | if  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$  then
4   |    $T[p, q] = 1$ ;
5 end
6 repeat
7   | for all  $(p, q) \in Q \times Q$  mit  $T[p, q] \neq 1$  do
8   |   | if  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  then  $T[p, q] = 1$ ;
9   |   | Bedingung zum Setzen neuer Marken
10  |   end
11 until keine neue Markierung gesetzt;
```

---



# Table-Filling Algorithmus

18

- Wir füllen nur die Hälfte der Tabelle  $T$  aus ( $T[p, q]$  mit  $p < q$ )

---

## Algorithm 1: TableFilling Algorithmus

---

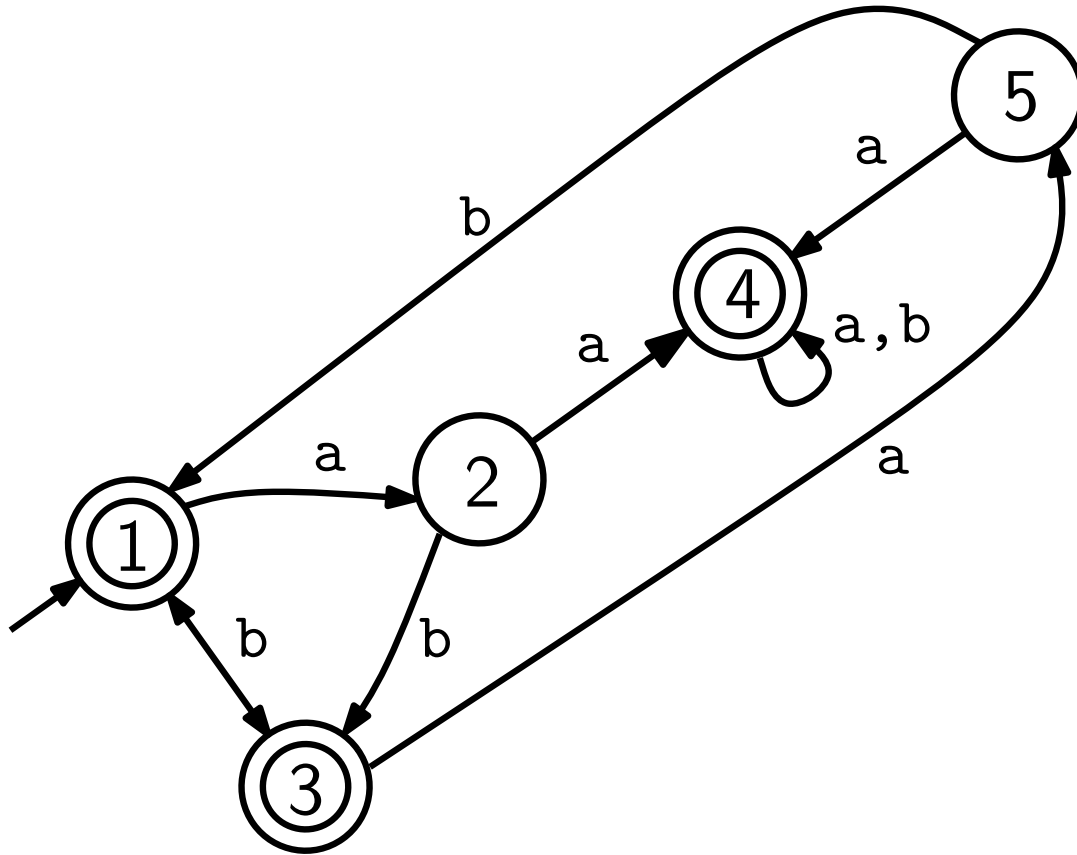
```
1 Initialisiere  $T \equiv 0$ ; Initialisierung
2 for all  $(p, q) \in Q \times Q$  do
3   | if  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$  then
4   |    $T[p, q] = 1$ ;
5 end
6 repeat
7   | for all  $(p, q) \in Q \times Q$  mit  $T[p, q] \neq 1$  do
8   |   | if  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  then  $T[p, q] = 1$ ;
9   |   end Bedingung zum Setzen neuer Marken
10 until keine neue Markierung gesetzt;
```

---

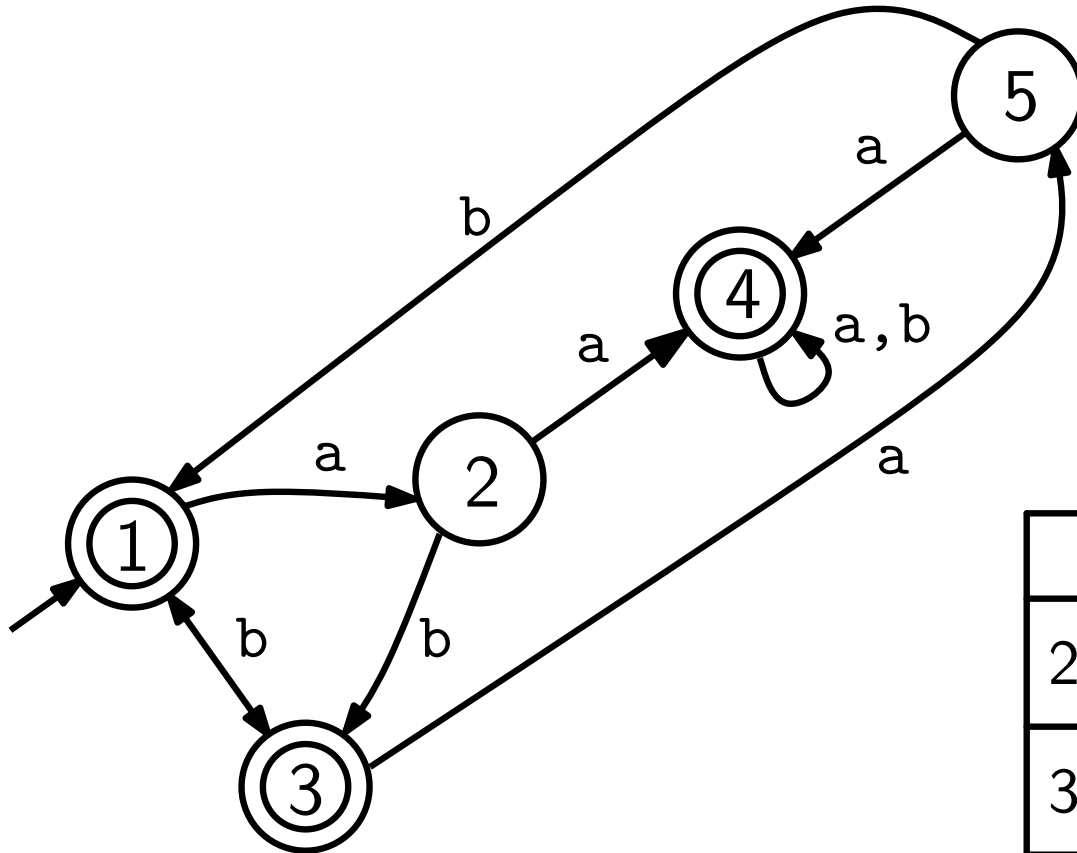
Abbruchbedingung

# Table-Filling Algorithmus im Beispiel<sup>19</sup>

# Table-Filling Algorithmus im Beispiel<sup>19</sup>

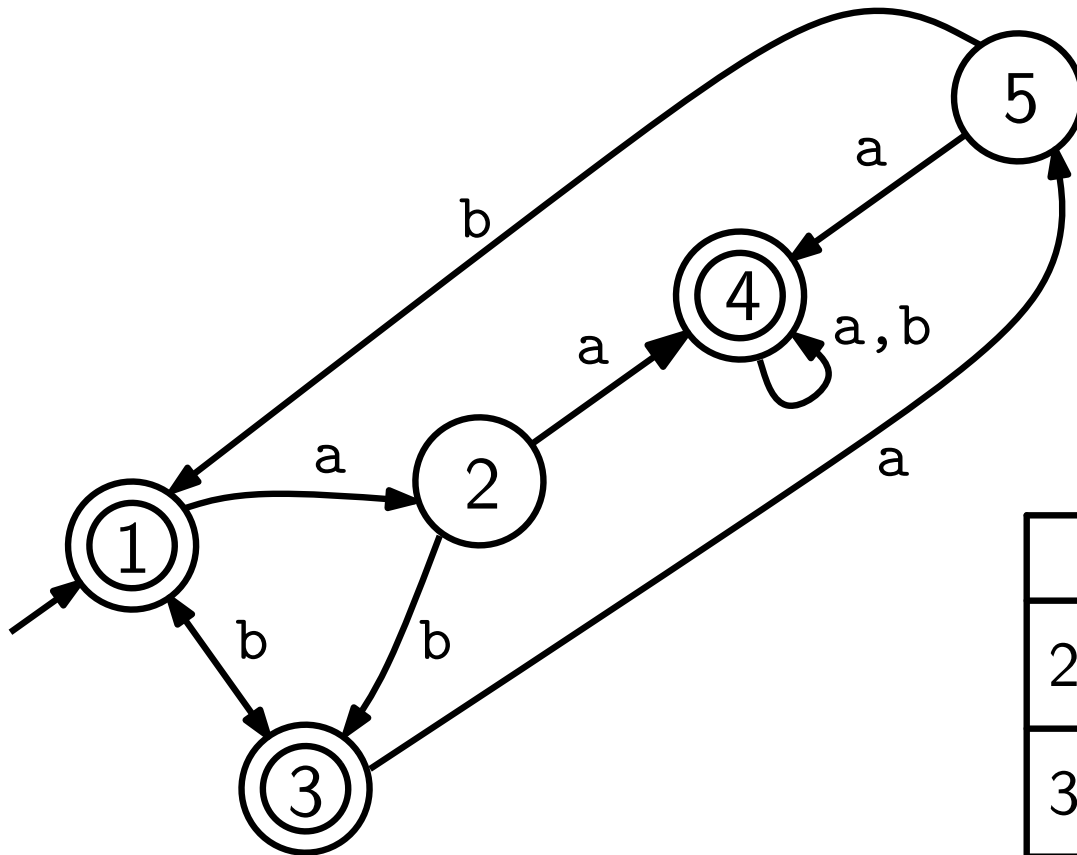


# Table-Filling Algorithmus im Beispiel<sup>19</sup>



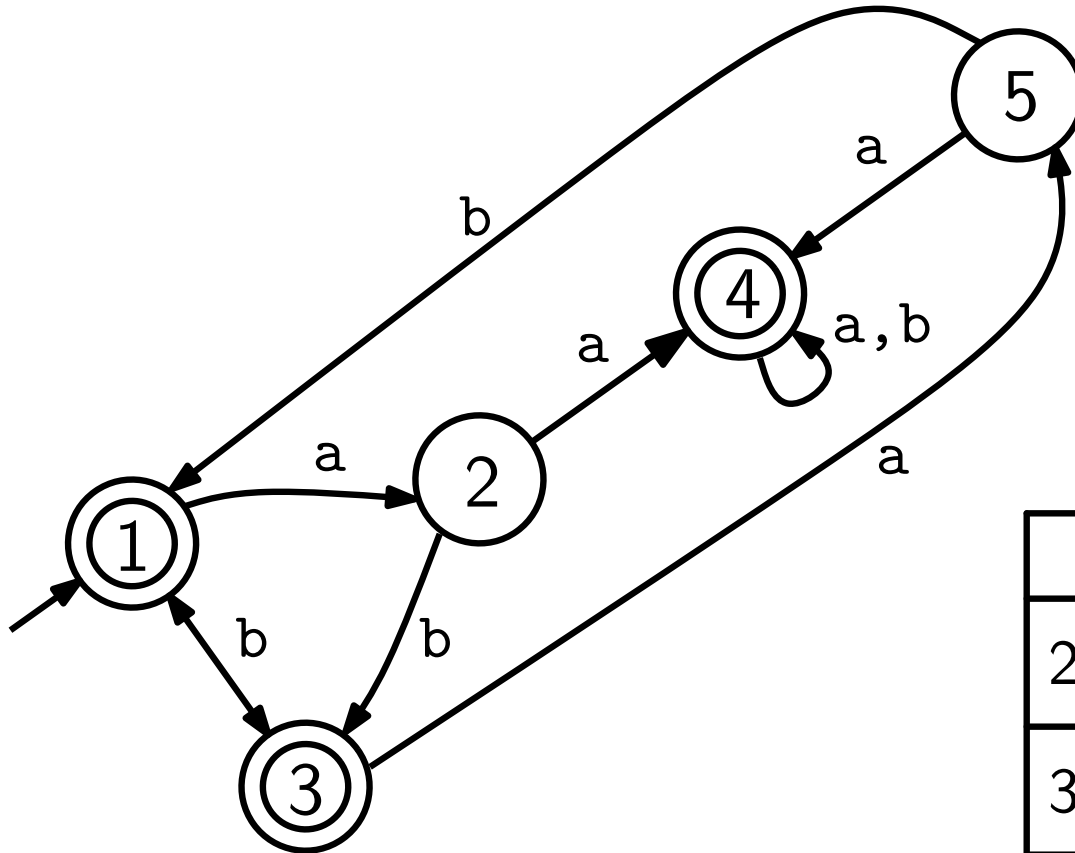
	1	2	3	4
2				
3				
4				
5				

# Table-Filling Algorithmus im Beispiel<sup>19</sup>



	1	2	3	4
2	1			
3		1		
4		1		
5	1		1	1

# Table-Filling Algorithmus im Beispiel<sup>19</sup>

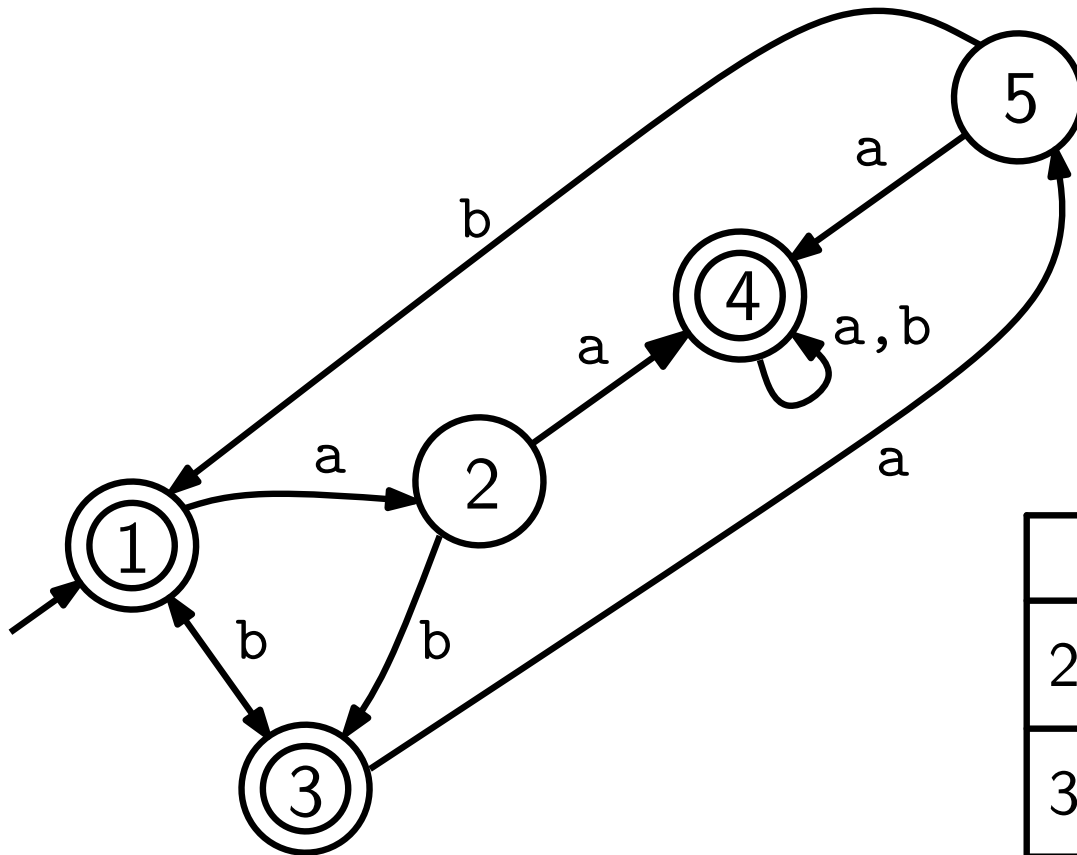


$$(1, 3) \xrightarrow{a} (2, 5)$$

$$(1, 3) \xrightarrow{b} (1, 3)$$

	1	2	3	4
2	1			
3		1		
4		1		
5	1		1	1

# Table-Filling Algorithmus im Beispiel<sup>19</sup>

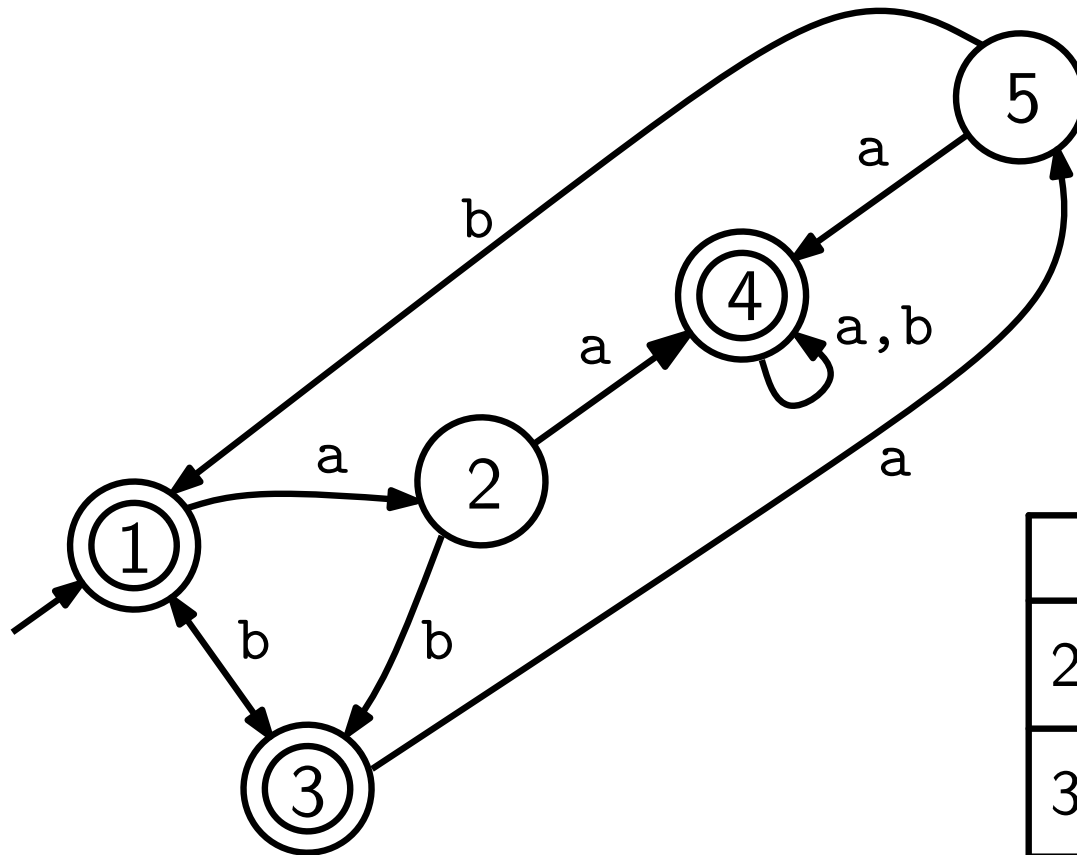


$$(1, 4) \xrightarrow{a} (2, 4)$$

$$(1, 4) \xrightarrow{b} (3, 4)$$

	1	2	3	4
2	1			
3		1		
4	1	1		
5	1		1	1

# Table-Filling Algorithmus im Beispiel<sup>19</sup>



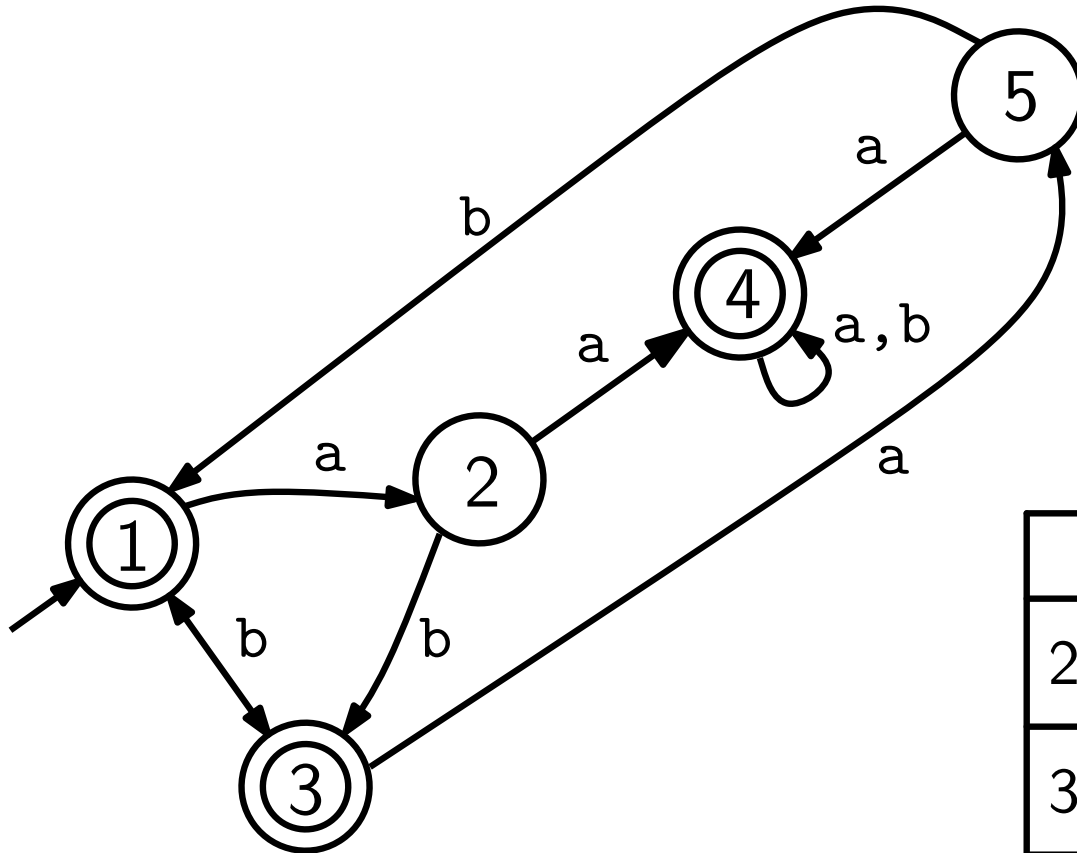
$$(3, 4) \xrightarrow{a} (5, 4)$$

$$(3, 4) \xrightarrow{b} (1, 4)$$

	1	2	3	4
2	1			
3		1		
4	1	1	1	
5	1		1	1



# Table-Filling Algorithmus im Beispiel<sup>19</sup>

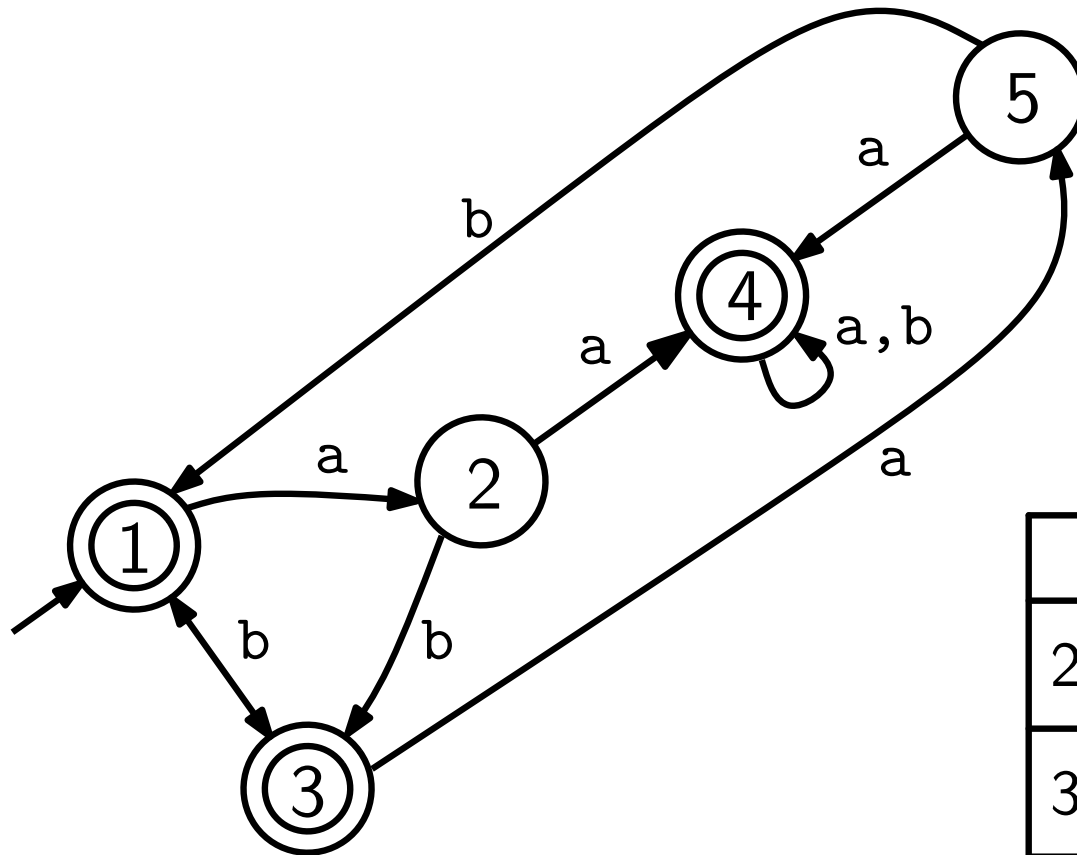


$$(2, 5) \xrightarrow{a} (4, 4)$$

$$(2, 5) \xrightarrow{b} (1, 3)$$

	1	2	3	4
2	1			
3		1		
4	1	1	1	
5	1		1	1

# Table-Filling Algorithmus im Beispiel<sup>19</sup>



Keine weiteren Veränderungen

→  $1 \approx 3$  und  $2 \approx 5$

	1	2	3	4
2	1			
3		1		
4	1	1	1	
5	1		1	1

# Korrektheit TF-Algorithmus

20

4. Vorlesung

# Korrektheit TF-Algorithmus

- Zu zeigen:
1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)
  2. alle trennbaren Paare wurden markiert

# Korrektheit TF-Algorithmus

Zu zeigen:

1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)
2. alle trennbaren Paare wurden markiert

- 1.** • 2 Möglichkeiten wie markiert wurde

# Korrektheit TF-Algorithmus

Zu zeigen: 1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)  
2. alle trennbaren Paare wurden markiert

1.

- 2 Möglichkeiten wie markiert wurde
- Bei der Initialisierung:  
**if**  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$   
**then**  $T[p, q] = 1$

# Korrektheit TF-Algorithmus

Zu zeigen:

1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)
2. alle trennbaren Paare wurden markiert

**1.**

- 2 Möglichkeiten wie markiert wurde
- Bei der Initialisierung:  
**if**  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$   
**then**  $T[p, q] = 1$   
 $\rightarrow p, q$  trennbar mit  $\varepsilon$

# Korrektheit TF-Algorithmus

Zu zeigen: 1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)  
2. alle trennbaren Paare wurden markiert

1.

- 2 Möglichkeiten wie markiert wurde
- Bei der Initialisierung:  
**if**  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$   
**then**  $T[p, q] = 1$   
 $\rightarrow p, q$  trennbar mit  $\varepsilon$
- In der **repeat** Schleife:  
**if**  $\exists a \in \Sigma : T[\delta(p, a), \delta(q, a)] == 1$  **then**  $T[p, q] = 1$



# Korrektheit TF-Algorithmus

Zu zeigen: 1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)  
2. alle trennbaren Paare wurden markiert

1.

- 2 Möglichkeiten wie markiert wurde
- Bei der Initialisierung:  
**if**  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$   
**then**  $T[p, q] = 1$   
 $\rightarrow p, q$  trennbar mit  $\varepsilon$
- In der **repeat** Schleife:  
**if**  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  **then**  $T[p, q] = 1$   
 $\rightarrow \delta(p, a), \delta(q, a)$  trennbar mit  $w \in \Sigma^*$  (Invariante)

# Korrektheit TF-Algorithmus

Zu zeigen: 1. nur trennbare Paare wurden markiert  
(Invariante bleibt erhalten)  
2. alle trennbaren Paare wurden markiert

1.

- 2 Möglichkeiten wie markiert wurde
- Bei der Initialisierung:  
**if**  $(p \in F \text{ und } q \notin F)$  oder  $(p \notin F \text{ und } q \in F)$   
**then**  $T[p, q] = 1$   
→  $p, q$  trennbar mit  $\varepsilon$
- In der **repeat** Schleife:  
**if**  $\exists a \in \Sigma: T[\delta(p, a), \delta(q, a)] == 1$  **then**  $T[p, q] = 1$   
→  $\delta(p, a), \delta(q, a)$  trennbar mit  $w \in \Sigma^*$  (Invariante)  
→  $p, q$  trennbar mit  $aw$

2. Alle trennbaren Paare wurden markiert

21

**2.** Alle trennbaren Paare wurden markiert

**Schlechtes Paar:** trennbar aber nicht markiert

**2.** Alle trennbaren Paare wurden markiert

**Schlechtes Paar:** trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$

**2.** Alle trennbaren Paare wurden markiert

**Schlechtes Paar:** trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden

## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$

## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$

$w =$ 

--	--	--	--	--	--	--	--	--	--

OBdA

$$\delta^*(p, w) \in F$$

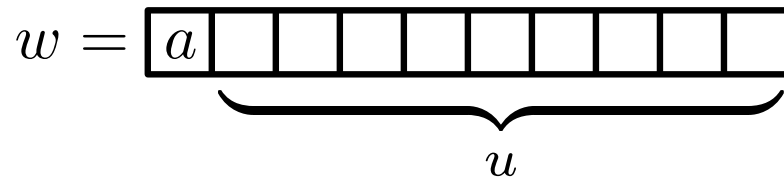
$$\delta^*(q, w) \notin F$$



## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$



OBdA

$$\delta^*(p', u) = \delta^*(p, w) \in F$$

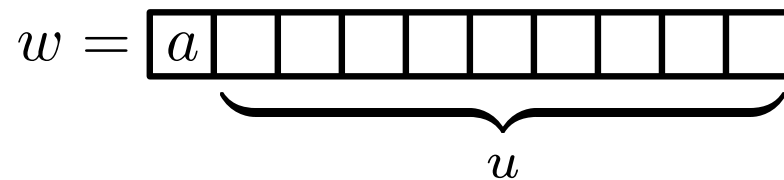
$$\delta^*(q', u) = \delta^*(q, w) \notin F$$

## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$

OBdA



$$\delta^*(p', u) = \delta^*(p, w) \in F$$

$$\delta^*(q', u) = \delta^*(q, w) \notin F$$

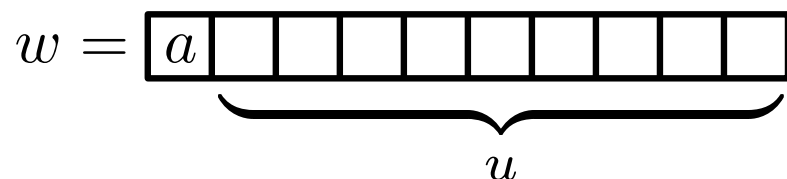
- $T[p', q'] \neq 1$ , denn sonst wäre während der Ausführung  $T[p, q] = 1$  gesetzt

## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$

OBdA



$$\delta^*(p', u) = \delta^*(p, w) \in F$$

$$\delta^*(q', u) = \delta^*(q, w) \notin F$$

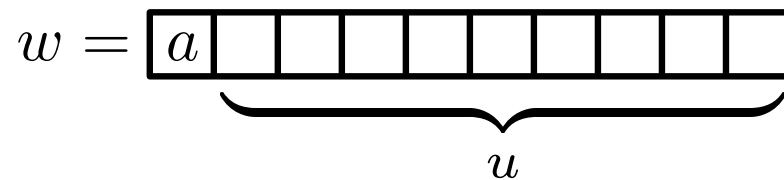
- $T[p', q'] \neq 1$ , denn sonst wäre während der Ausführung  $T[p, q] = 1$  gesetzt
- **aber:**  $(p', q')$  trennbar mit  $u$ , und  $|u| < |w|$

## 2. Alle trennbaren Paare wurden markiert

### Schlechtes Paar: trennbar aber nicht markiert

- Annahme: sei  $(p, q)$  schlechtes Paar mit kürzestem Trennwort  $w$  und kein schlechtes Paar besitzt ein Trennwort kürzer als  $w$
- $w \neq \varepsilon$ , denn sonst wäre  $(p, q)$  während der Initialisierung markiert worden
- $w = au$  mit  $a \in \Sigma$
- $p' = \delta(p, a)$  und  $q' = \delta(q, a)$

OBdA



$$\delta^*(p', u) = \delta^*(p, w) \in F$$

$$\delta^*(q', u) = \delta^*(q, w) \notin F$$

- $T[p', q'] \neq 1$ , denn sonst wäre während der Ausführung  $T[p, q] = 1$  gesetzt
- **aber:**  $(p', q')$  trennbar mit  $u$ , und  $|u| < |w|$   
 → Widerspruch zur Annahme, d.h. es gibt kein schlechtes Paar!