



# F2 — Automaten und formale Sprachen

Matthias Jantzen

(nach und mit Folienvorlagen von Berndt Farwer)

Fachbereich Informatik

AB „Theoretische Grundlagen der Informatik“ (TGI)

Universität Hamburg

*jantzen@informatik.uni-hamburg.de*



# Potenzautomat

- Idee: Die Potenzmenge von  $Z$  wird als neue Zustandsmenge verwendet.



# Potenzautomat

- Idee: Die Potenzmenge von  $Z$  wird als neue Zustandsmenge verwendet.
- Werden alle Zustände benötigt?



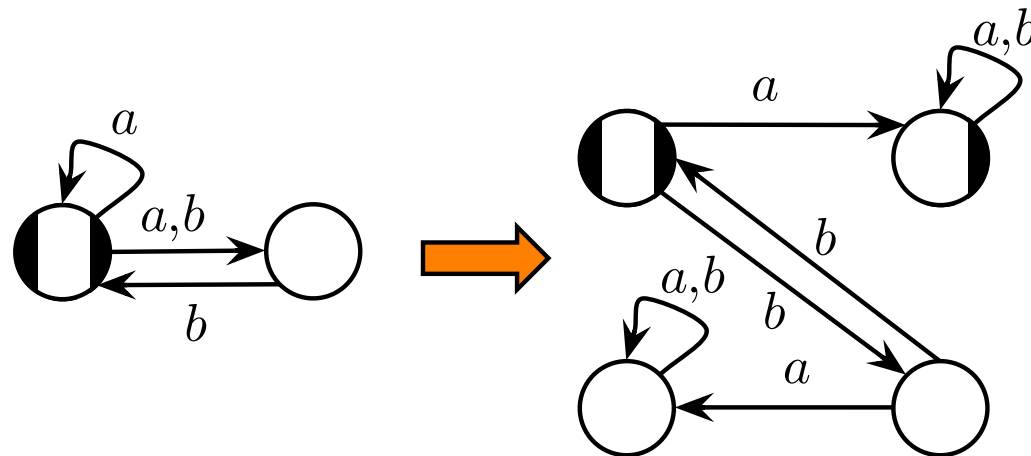
# Potenzautomat

- Idee: Die Potenzmenge von  $Z$  wird als neue Zustandsmenge verwendet.
- Werden alle Zustände benötigt?
- Es gibt NFAs, die tatsächlich alle  $2^{|Z|}$  Zustände erfordern!



# Potenzautomat

- Idee: Die Potenzmenge von  $Z$  wird als neue Zustandsmenge verwendet.
- Werden alle Zustände benötigt?
- Es gibt NFAs, die tatsächlich alle  $2^{|Z|}$  Zustände erfordern!
- **Beispiel:**





# Endliche Sprachen $\in \mathcal{Reg}$

- **Theorem:** Jede endliche Sprache ist regulär.



# Endliche Sprachen $\in \mathcal{Reg}$

- **Theorem:** Jede endliche Sprache ist regulär.
- **Beweis:** Sei  $L = \{w_1, w_2, \dots, w_n\}$  endlich, d.h.  $|L| = n \in \mathbb{N}$ .



# Endliche Sprachen $\in \mathcal{Reg}$

- **Theorem:** Jede endliche Sprache ist regulär.
- **Beweis:** Sei  $L = \{w_1, w_2, \dots, w_n\}$  endlich, d.h.  $|L| = n \in \mathbb{N}$ .
  - Es gibt endliches Alphabet  $\Sigma$  mit  $L \subseteq \Sigma$ .





# Endliche Sprachen $\in \mathcal{Reg}$

- **Theorem:** Jede endliche Sprache ist regulär.
- **Beweis:** Sei  $L = \{w_1, w_2, \dots, w_n\}$  endlich, d.h.  $|L| = n \in \mathbb{N}$ .
  - Es gibt endliches Alphabet  $\Sigma$  mit  $L \subseteq \Sigma^*$ .
  - NFA  $A = (\{z, z'\}, \Sigma, K, \{z\}, \{z'\})$  mit

$$K := \{(z, w_i, z') \mid w_i \in L\}$$

akzeptiert  $L$ .



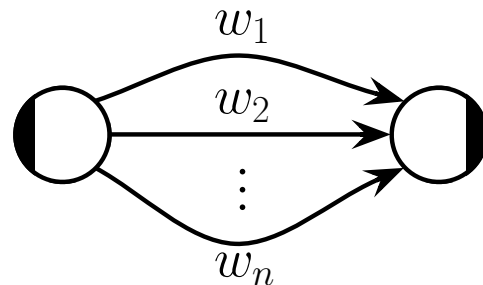
# Endliche Sprachen $\in \mathcal{Reg}$

- **Theorem:** Jede endliche Sprache ist regulär.
- **Beweis:** Sei  $L = \{w_1, w_2, \dots, w_n\}$  endlich, d.h.  $|L| = n \in \mathbb{N}$ .
  - Es gibt endliches Alphabet  $\Sigma$  mit  $L \subseteq \Sigma^*$ .
  - NFA  $A = (\{z, z'\}, \Sigma, K, \{z\}, \{z'\})$  mit

$$K := \{(z, w_i, z') \mid w_i \in L\}$$

akzeptiert  $L$ .

- graphisch:





# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.



# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.



# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.
- Sind  $A$  und  $B$  rationale Ausdrücke, welche die Mengen  $M_A$  und  $M_B$  beschreiben, dann sind **induktiv** folgende rationalen Ausdrücke definiert:



# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

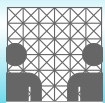
- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.
- Sind  $A$  und  $B$  rationale Ausdrücke, welche die Mengen  $M_A$  und  $M_B$  beschreiben, dann sind **induktiv** folgende rationalen Ausdrücke definiert:
  - $(A + B)$  beschreibt die Menge  $M_A \cup M_B$ ,



# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.
- Sind  $A$  und  $B$  rationale Ausdrücke, welche die Mengen  $M_A$  und  $M_B$  beschreiben, dann sind **induktiv** folgende rationalen Ausdrücke definiert:
  - $(A + B)$  beschreibt die Menge  $M_A \cup M_B$ ,
  - $A \cdot B$  beschreibt die Menge  $M_A \cdot M_B$



# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.
- Sind  $A$  und  $B$  rationale Ausdrücke, welche die Mengen  $M_A$  und  $M_B$  beschreiben, dann sind **induktiv** folgende rationalen Ausdrücke definiert:
  - $(A + B)$  beschreibt die Menge  $M_A \cup M_B$ ,
  - $A \cdot B$  beschreibt die Menge  $M_A \cdot M_B$
  - $(A)^*$  beschreibt die Menge  $M_A^*$





# Rationale Ausdrücke

**Definition:** Die **rationalen Ausdrücke** über einem endlichen Alphabet  $\Sigma$  sind definiert durch:

- $\emptyset$  ist ein rationaler Ausdruck, der die (*leere*) Menge  $M_{\emptyset} := \emptyset$  beschreibt.
- Für jedes  $a \in \Sigma$  ist  $a$  ein rationaler Ausdruck, der die Menge  $M_a := \{a\}$  beschreibt.
- Sind  $A$  und  $B$  rationale Ausdrücke, welche die Mengen  $M_A$  und  $M_B$  beschreiben, dann sind **induktiv** folgende rationalen Ausdrücke definiert:
  - $(A + B)$  beschreibt die Menge  $M_A \cup M_B$ ,
  - $A \cdot B$  beschreibt die Menge  $M_A \cdot M_B$
  - $(A)^*$  beschreibt die Menge  $M_A^*$
  - $(A)^+$  beschreibt die Menge  $M_A^+$



# Sprachfamilie $\mathcal{Rat}$

- Es bezeichne  $\mathcal{Rat}(\Sigma)$  die Familie aller mit rationalen Ausdrücken beschreibbaren Teilmengen von  $\Sigma^*$ .



# Sprachfamilie $\mathcal{Rat}$

- Es bezeichne  $\mathcal{Rat}(\Sigma)$  die Familie aller mit rationalen Ausdrücken beschreibbaren Teilmengen von  $\Sigma^*$ .
- Zusammenfassend:  $\mathcal{Rat} := \bigcup_{\Sigma \text{ ist endl. Alphabet}} \mathcal{Rat}(\Sigma)$ .



# Sprachfamilie $\mathcal{Rat}$

- Es bezeichne  $\mathcal{Rat}(\Sigma)$  die Familie aller mit rationalen Ausdrücken beschreibbaren Teilmengen von  $\Sigma^*$ .
- Zusammenfassend:  $\mathcal{Rat} := \bigcup_{\Sigma \text{ ist endl. Alphabet}} \mathcal{Rat}(\Sigma)$ .
- Regeln zur Klammerersparnis:



# Sprachfamilie $\mathcal{Rat}$

- Es bezeichne  $\mathcal{Rat}(\Sigma)$  die Familie aller mit rationalen Ausdrücken beschreibbaren Teilmengen von  $\Sigma^*$ .
- Zusammenfassend:  $\mathcal{Rat} := \bigcup_{\Sigma \text{ ist endl. Alphabet}} \mathcal{Rat}(\Sigma)$ .
- Regeln zur Klammerersparnis:
  - unäre Operatoren vor binären ( $*$  vor  $\cdot$  und  $+$ )



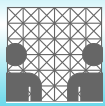
# Sprachfamilie $\mathcal{Rat}$

- Es bezeichne  $\mathcal{Rat}(\Sigma)$  die Familie aller mit rationalen Ausdrücken beschreibbaren Teilmengen von  $\Sigma^*$ .
- Zusammenfassend:  $\mathcal{Rat} := \bigcup_{\Sigma \text{ ist endl. Alphabet}} \mathcal{Rat}(\Sigma)$ .
- Regeln zur Klammerersparnis:
  - unäre Operatoren vor binären ( $*$  vor  $\cdot$  und  $+$ )
  - Punkt vor Strich ( $\cdot$  vor  $+$ )



# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:



# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:
  - Suche mit „regulären Ausdrücken“



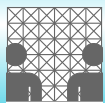


# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:
  - Suche mit „regulären Ausdrücken“
- Gibt man z.B. den Befehl

```
egrep ^Th . . . i . $ /usr/dict/duden
```

ein, so könnten folgende Wörter gefunden und angezeigt werden:

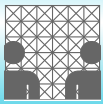


# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:
  - Suche mit „regulären Ausdrücken“
- Gibt man z.B. den Befehl

```
egrep ^Th . . . i . $ /usr/dict/duden
```

ein, so könnten folgende Wörter gefunden und angezeigt werden:
  - Thermik



# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:
  - Suche mit „regulären Ausdrücken“
- Gibt man z.B. den Befehl

```
egrep ^Th . . . i . $ /usr/dict/duden
```

ein, so könnten folgende Wörter gefunden und angezeigt werden:
  - Thermik
  - Theorie

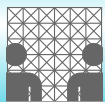


# Anwendung: Unix-CLI

- grep und egrep auf Unix-Betriebssystemen und in Editoren:
  - Suche mit „regulären Ausdrücken“
- Gibt man z.B. den Befehl

```
egrep ^Th . . . i . $ /usr/dict/duden
```

ein, so könnten folgende Wörter gefunden und angezeigt werden:
  - Thermik
  - Theorie
  - Thespis



# egrep-Notation

rationaler Ausdruck	egrep-Notation
Zeilenanfang	$\wedge$
Zeilenende	$\$$
$c$	$c$
(Gesamtalphabet:) $\Sigma$	$.$
$r + \emptyset^*$	$r?$
$r^*$	$r^*$
$r^+$	$r^+$
$r + s$	$r s$
$r \cdot s$	$rs$
$(r)$	$(r)$



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:
  - DFA





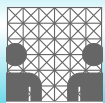
# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:
  - DFA
  - vDFA



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:
  - DFA
  - vDFA
  - NFA



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:
  - DFA
  - vDFA
  - NFA
  - $\lambda$ -freier NFA



# Grundannahmen

- Sei  $L$  eine reguläre Sprache.
- In allen weiteren Konstruktionen dürfen wir nach belieben voraussetzen, dass wir einen der folgenden Automaten vorliegen haben, der  $L$  akzeptiert:
  - DFA
  - vDFA
  - NFA
  - $\lambda$ -freier NFA
  - buchstabierender NFA



# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .



# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )



# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  mit  $L_1 = L(A)$  und  $L_2 = L(B)$  zwei vDFAs.



# Abschlussoperator: Vereinigung

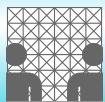
- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  mit  $L_1 = L(A)$  und  $L_2 = L(B)$  zwei vDFAs.
  - Der NFA für  $L_1 \cup L_2$  wird definiert durch:  
 $C_{A \cup B} := (Z_1 \uplus Z_2, \Sigma_1 \cup \Sigma_2, K_3, Z_{3,0}, Z_{3,\text{end}})$





# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  mit  $L_1 = L(A)$  und  $L_2 = L(B)$  zwei vDFAs.
  - Der NFA für  $L_1 \cup L_2$  wird definiert durch:  
 $C_{A \cup B} := (Z_1 \uplus Z_2, \Sigma_1 \cup \Sigma_2, K_3, Z_{3,0}, Z_{3,\text{end}})$ 
    - $Z_{3,\text{start}} := \{z_{1,0}, z_{2,0}\}$



# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  mit  $L_1 = L(A)$  und  $L_2 = L(B)$  zwei vDFAs.
  - Der NFA für  $L_1 \cup L_2$  wird definiert durch:
$$C_{A \cup B} := (Z_1 \uplus Z_2, \Sigma_1 \cup \Sigma_2, K_3, Z_{3,0}, Z_{3,\text{end}})$$
    - $Z_{3,\text{start}} := \{z_{1,0}, z_{2,0}\}$
    - $Z_{3,\text{end}} := Z_{1,\text{end}} \cup Z_{2,\text{end}}$



# Abschlussoperator: Vereinigung

- **Theorem:** Mit  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$  ist  $L_1 \cup L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines NFA für  $L_1 \cup L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  mit  $L_1 = L(A)$  und  $L_2 = L(B)$  zwei vDFAs.
  - Der NFA für  $L_1 \cup L_2$  wird definiert durch:  
 $C_{A \cup B} := (Z_1 \uplus Z_2, \Sigma_1 \cup \Sigma_2, K_3, Z_{3,0}, Z_{3,\text{end}})$ 
    - $Z_{3,\text{start}} := \{z_{1,0}, z_{2,0}\}$
    - $Z_{3,\text{end}} := Z_{1,\text{end}} \cup Z_{2,\text{end}}$
    - $K_3 := \{(p_1, x, \delta_1(p_1, x)) \mid p_1 \in Z_1, x \in \Sigma_1\} \cup \{(p_2, x, \delta_2(p_2, x)) \mid p_2 \in Z_2, x \in \Sigma_2\}$



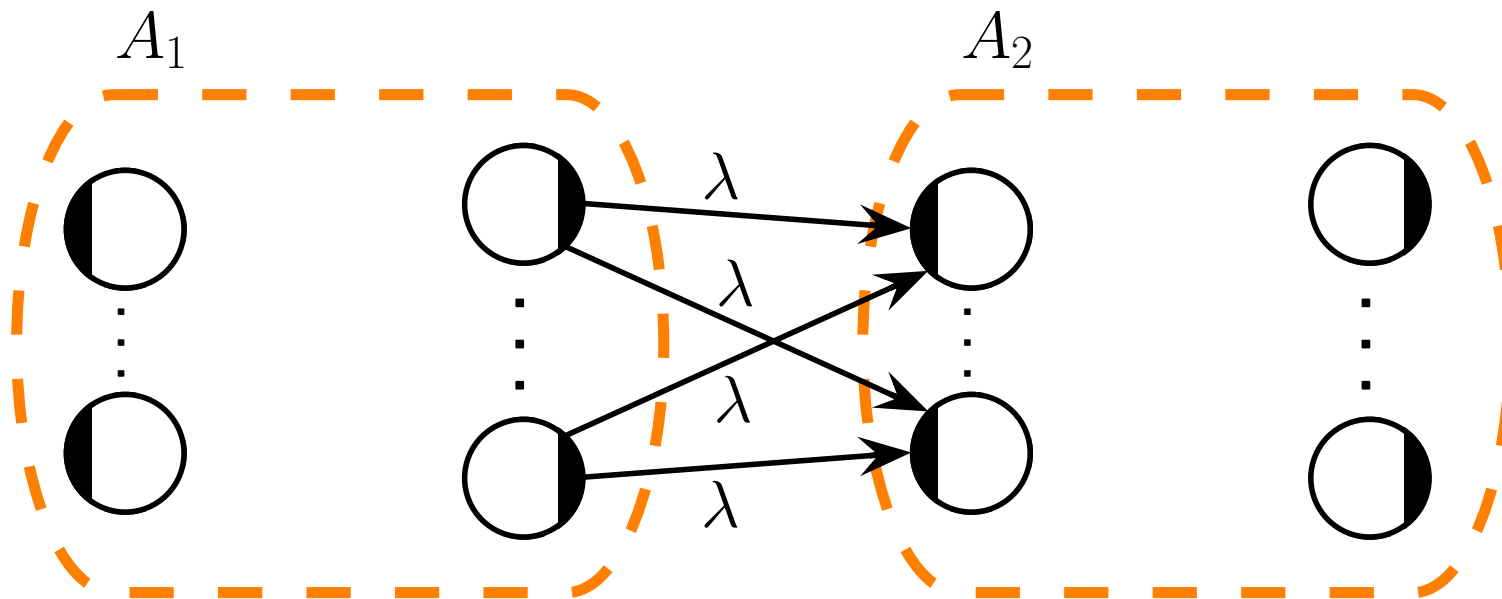
# Produkt (informell)

- Bereits gezeigt: Abschluss gegen Produktbildung.



# Produkt (informell)

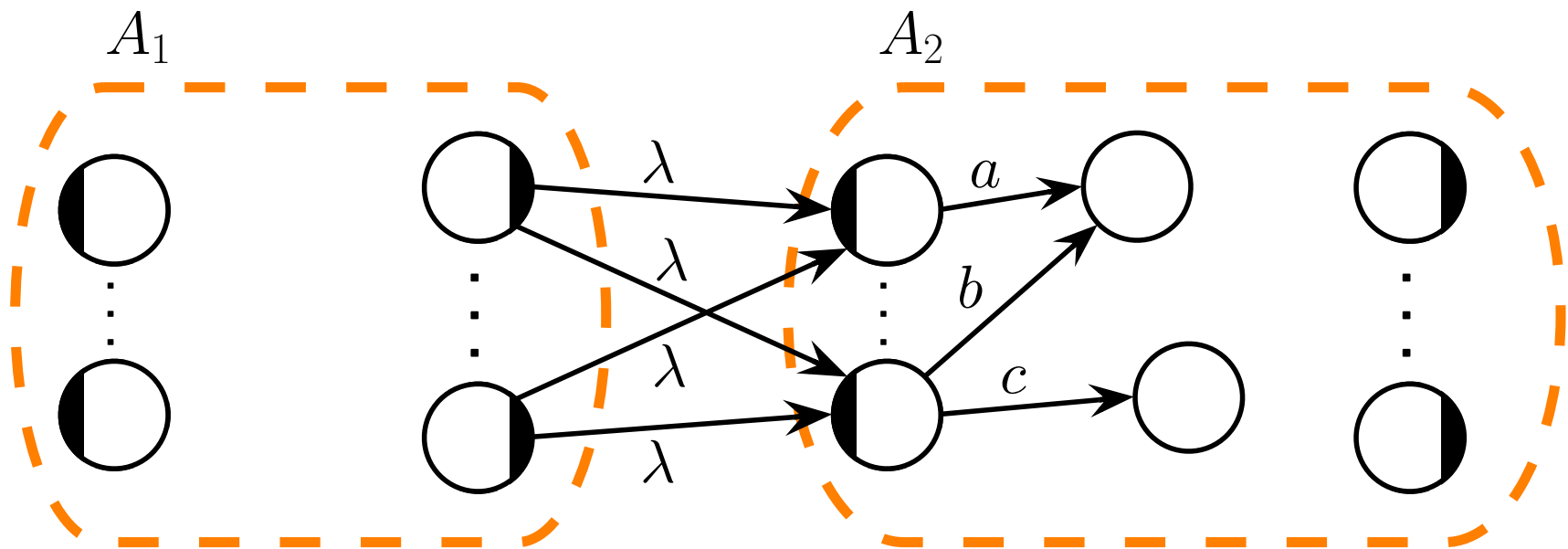
- Bereits gezeigt: Abschluss gegen Produktbildung.
- Alternative Konstruktion führt zu einem buchstabierenden FA:





# Produkt (informell)

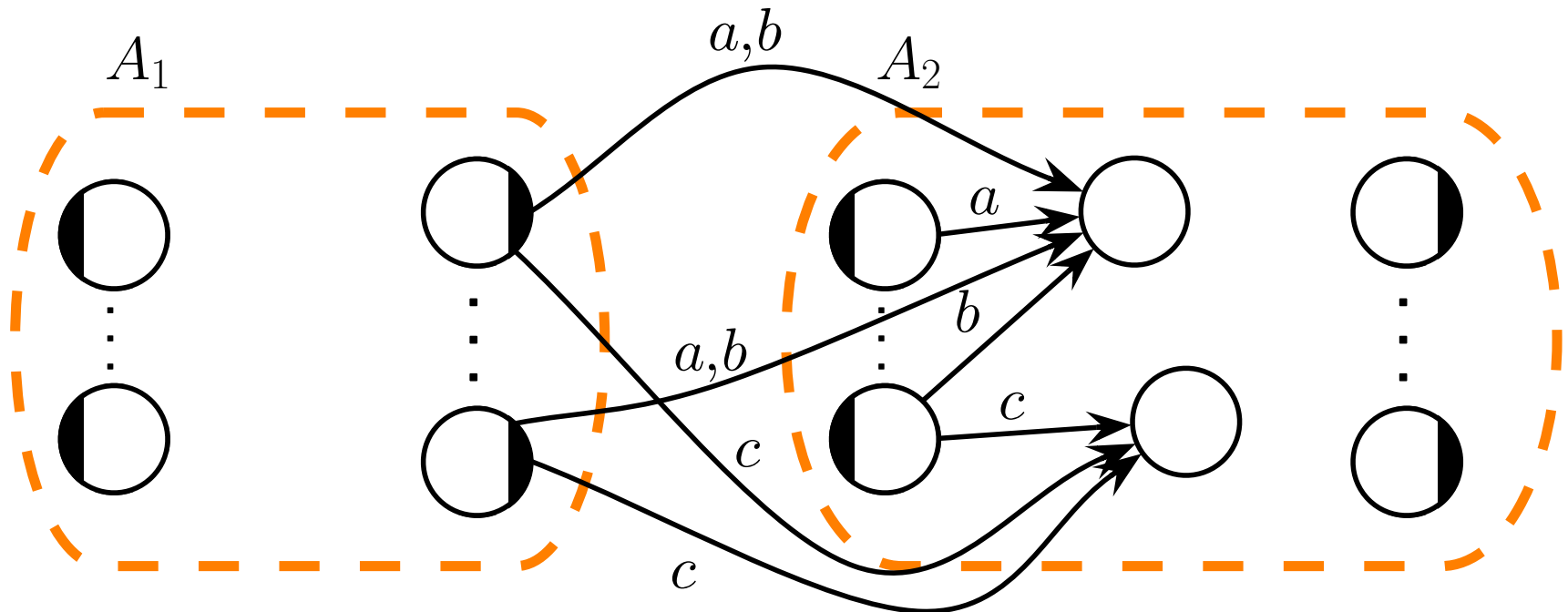
- Bereits gezeigt: Abschluss gegen Produktbildung.
- Alternative Konstruktion führt zu einem buchstabierenden FA:





# Produkt (informell)

- Bereits gezeigt: Abschluss gegen Produktbildung.
- Alternative Konstruktion führt zu einem buchstabierenden FA:





# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .





# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )



# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  vollständige DFA's mit  $Z_1 \cap Z_2 = \emptyset$ ,  $L_1 = L(A)$  und  $L_2 = L(B)$ .



# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  vollständige DFA's mit  $Z_1 \cap Z_2 = \emptyset$ ,  $L_1 = L(A)$  und  $L_2 = L(B)$ .
  - Definiere NFA  $C_{A \cdot B} := (Z_1 \cup Z_2, \Sigma_1 \cup \Sigma_2, K_1 \cup K_2 \cup K_3, z_{1,0}, Z_{2,\text{end}})$  mit



# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  vollständige DFA's mit  $Z_1 \cap Z_2 = \emptyset$ ,  $L_1 = L(A)$  und  $L_2 = L(B)$ .
  - Definiere NFA  $C_{A \cdot B} := (Z_1 \cup Z_2, \Sigma_1 \cup \Sigma_2, K_1 \cup K_2 \cup K_3, z_{1,0}, Z_{2,\text{end}})$  mit
    - $K_1 := \{(p_1, x, \delta_1(p_1, x)) \mid x \in \Sigma_1, p_1 \in Z_1\}$



# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  vollständige DFA's mit  $Z_1 \cap Z_2 = \emptyset$ ,  $L_1 = L(A)$  und  $L_2 = L(B)$ .
  - Definiere NFA  $C_{A \cdot B} := (Z_1 \cup Z_2, \Sigma_1 \cup \Sigma_2, K_1 \cup K_2 \cup K_3, z_{1,0}, Z_{2,\text{end}})$  mit
    - $K_1 := \{(p_1, x, \delta_1(p_1, x)) \mid x \in \Sigma_1, p_1 \in Z_1\}$
    - $K_2 := \{(p_2, x, \delta_2(p_2, x)) \mid x \in \Sigma_2, p_2 \in Z_2\}$



# Abschlussoperator: Produkt

- **Theorem:** Seien  $L_1 \in \mathcal{A}kz(\Sigma_1)$  und  $L_2 \in \mathcal{A}kz(\Sigma_2)$ , dann ist  $L_1 \cdot L_2 \in \mathcal{A}kz(\Sigma_1 \cup \Sigma_2)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L_1 \cdot L_2$ )
  - Seien  $A := (Z_1, \Sigma_1, \delta_1, z_{1,0}, Z_{1,\text{end}})$  und  $B := (Z_2, \Sigma_2, \delta_2, z_{2,0}, Z_{2,\text{end}})$  vollständige DFA's mit  $Z_1 \cap Z_2 = \emptyset$ ,  $L_1 = L(A)$  und  $L_2 = L(B)$ .
  - Definiere NFA  $C_{A \cdot B} := (Z_1 \cup Z_2, \Sigma_1 \cup \Sigma_2, K_1 \cup K_2 \cup K_3, z_{1,0}, Z_{2,\text{end}})$  mit
    - $K_1 := \{(p_1, x, \delta_1(p_1, x)) \mid x \in \Sigma_1, p_1 \in Z_1\}$
    - $K_2 := \{(p_2, x, \delta_2(p_2, x)) \mid x \in \Sigma_2, p_2 \in Z_2\}$
    - $K_3 := \{(p_1, x, \delta_2(z_{2,0}, x)) \mid x \in \Sigma_2, p_1 \in Z_{1,\text{end}}\}$die  $A$  und  $B$  verbindenden, Nicht- $\lambda$ -Kanten.



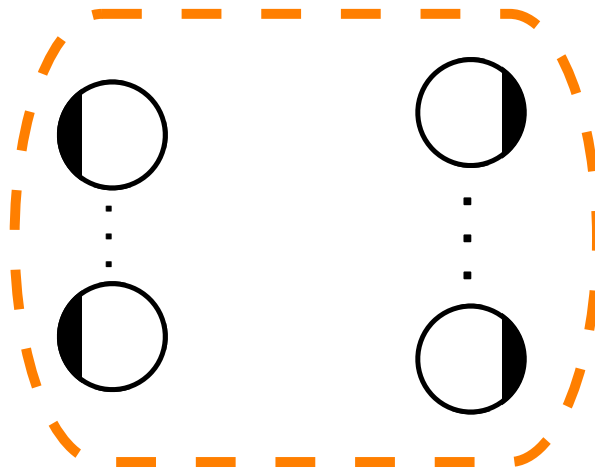
# Stern (informell)

- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.



# Stern (informell)

- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:

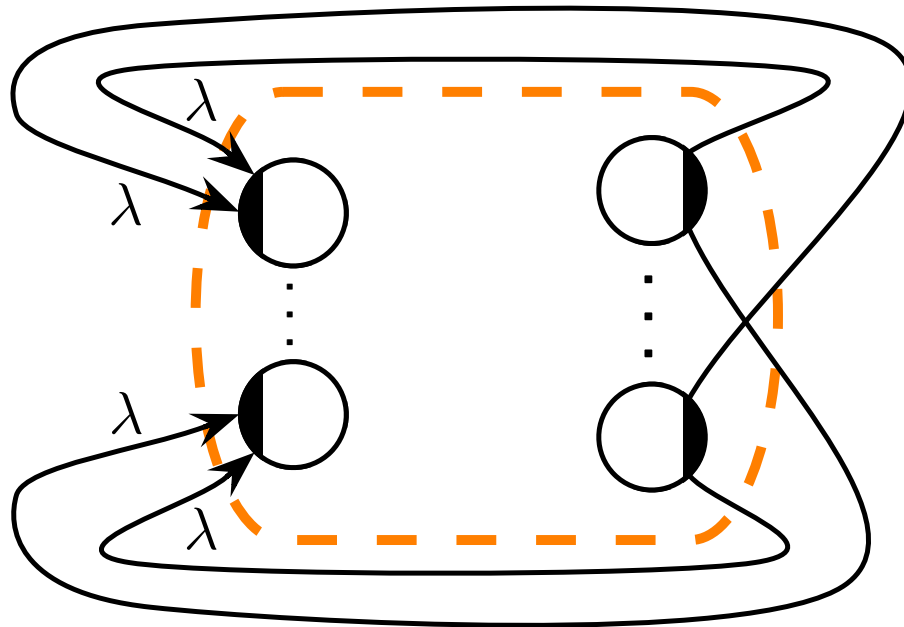






# Stern (informell)

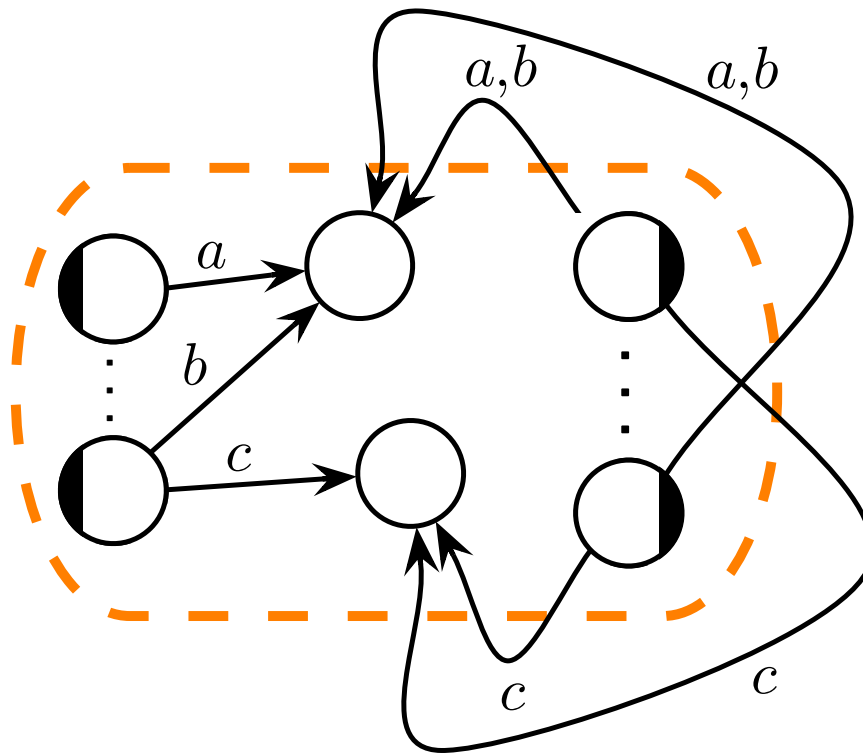
- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:





# Stern (informell)

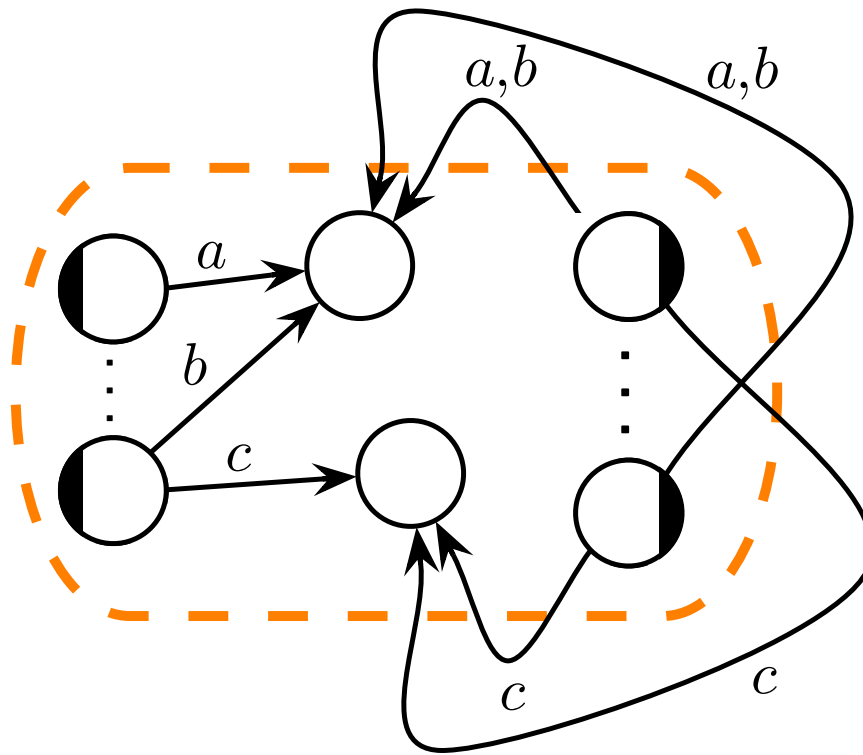
- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:





# Stern (informell)

- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:

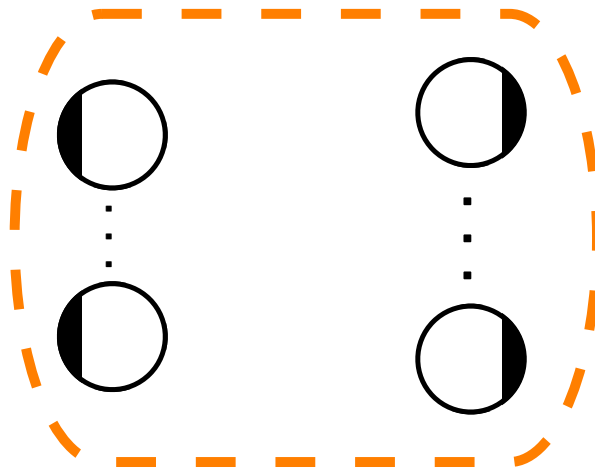


... aber dieser Automat akzeptiert nicht immer  $\lambda \in L^*$ !



# Stern (informell)

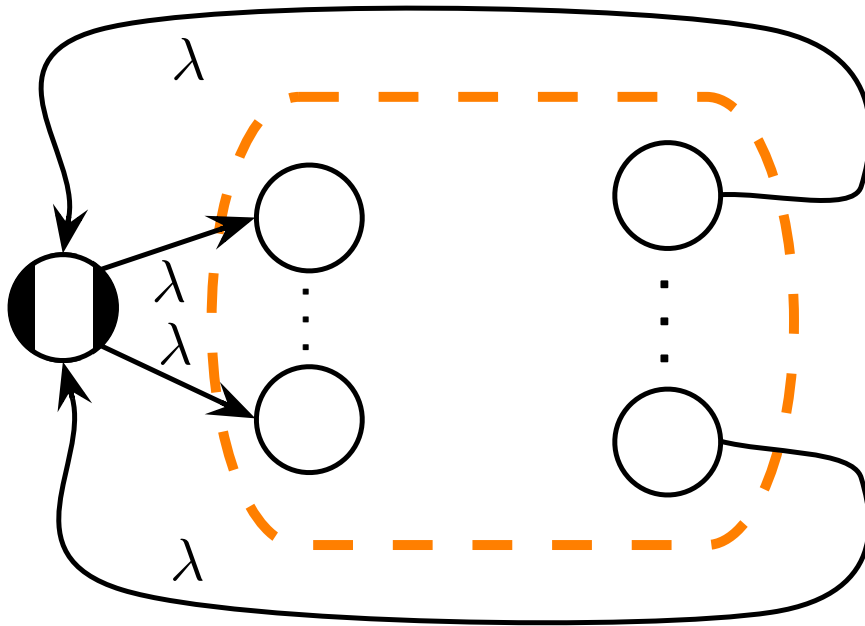
- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:





# Stern (informell)

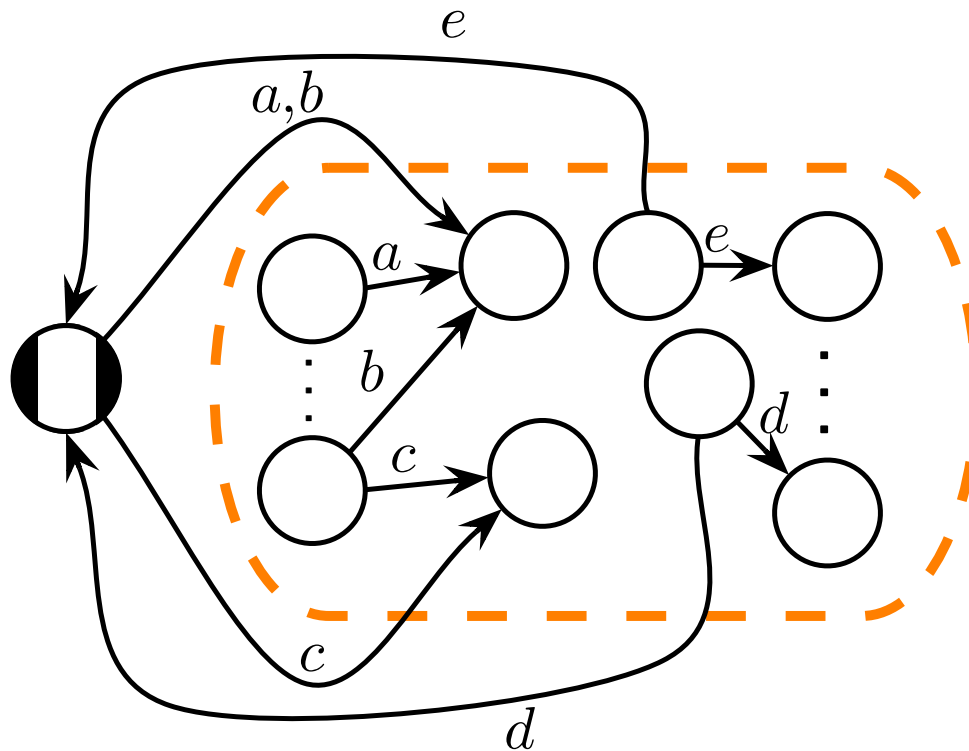
- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:





# Stern (informell)

- Gegeben: Ein endlicher Automat, der  $L$  akzeptiert.
- Konstruktion eines (buchstabierenden) FA, der die Sprache  $L^*$  akzeptiert:





# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .



# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )





# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )
  - Sei  $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$  ein vDFA mit  $L = L(A)$ .



# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )
  - Sei  $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$  ein vDFA mit  $L = L(A)$ .
  - Definiere buchstabierenden NFA  
 $C_{A^*} := (Z \uplus \{p\}, \Sigma, K_1 \cup K_2 \cup K_3, \{p\}, \{p\})$   
mit  $K := K_1 \cup K_2 \cup K_3$ , wobei



# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )
  - Sei  $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$  ein vDFA mit  $L = L(A)$ .
  - Definiere buchstabierenden NFA  $C_{A^*} := (Z \uplus \{p\}, \Sigma, K_1 \cup K_2 \cup K_3, \{p\}, \{p\})$  mit  $K := K_1 \cup K_2 \cup K_3$ , wobei
    - $K_1 := \{(z, x, \delta(p, x)) \mid x \in \Sigma, z \in Z\}$  die zum vDFA  $A$  gehörende Kantenmenge,



# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )
  - Sei  $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$  ein vDFA mit  $L = L(A)$ .
  - Definiere buchstabierenden NFA  $C_{A^*} := (Z \uplus \{p\}, \Sigma, K_1 \cup K_2 \cup K_3, \{p\}, \{p\})$  mit  $K := K_1 \cup K_2 \cup K_3$ , wobei
    - $K_1 := \{(z, x, \delta(p, x)) \mid x \in \Sigma, z \in Z\}$  die zum vDFA  $A$  gehörende Kantenmenge,
    - $K_2 := \{(p, x, \delta(z_0, x)) \mid x \in \Sigma\}$  und



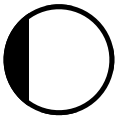
# Abschlussoperator: Stern

- Sei  $L \in \mathcal{A}kz(\Sigma)$ , dann ist auch  $L^* \in \mathcal{A}kz(\Sigma)$ .
- **Beweis:** (Konstruktion eines buchstabierenden NFA für  $L^*$ )
  - Sei  $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$  ein vDFA mit  $L = L(A)$ .
  - Definiere buchstabierenden NFA  $C_{A^*} := (Z \uplus \{p\}, \Sigma, K_1 \cup K_2 \cup K_3, \{p\}, \{p\})$  mit  $K := K_1 \cup K_2 \cup K_3$ , wobei
    - $K_1 := \{(z, x, \delta(p, x)) \mid x \in \Sigma, z \in Z\}$  die zum vDFA  $A$  gehörende Kantenmenge,
    - $K_2 := \{(p, x, \delta(z_0, x)) \mid x \in \Sigma\}$  und
    - $K_3 := \{(z, x, p) \mid x \in \Sigma, \delta(z, x) \in Z_{\text{end}}\}$  die Verbindungen mit dem neuen Startzustand sind.



# Einfache Mengen sind regulär!

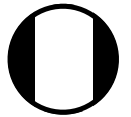
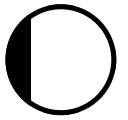
- Die leere Menge ist regulär.





# Einfache Mengen sind regulär!

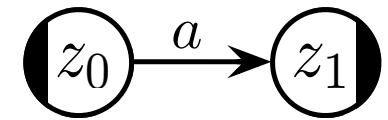
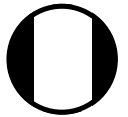
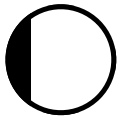
- Die leere Menge ist regulär.
- Die Menge  $\emptyset^* = \{\lambda\}$  ist regulär.





# Einfache Mengen sind regulär!

- Die leere Menge ist regulär.
- Die Menge  $\emptyset^* = \{\lambda\}$  ist regulär.
- Die Menge  $\{a\}$  ist regulär.

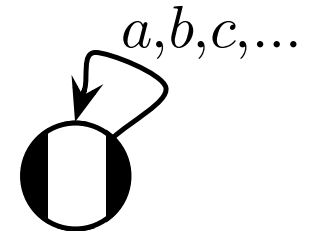
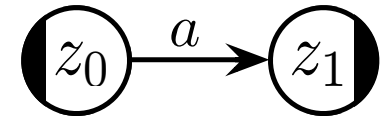
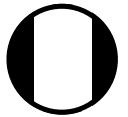
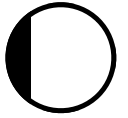






# Einfache Mengen sind regulär!

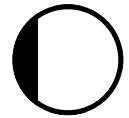
- Die leere Menge ist regulär.
- Die Menge  $\emptyset^* = \{\lambda\}$  ist regulär.
- Die Menge  $\{a\}$  ist regulär.
- $\Sigma^*$  ist regulär.





# Einfache Mengen sind regulär!

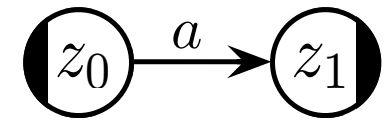
- Die leere Menge ist regulär.



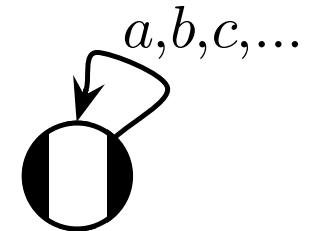
- Die Menge  $\emptyset^* = \{\lambda\}$  ist regulär.



- Die Menge  $\{a\}$  ist regulär.



- $\Sigma^*$  ist regulär.

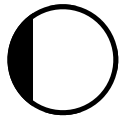


- Für jede Menge  $M$  ist  $M^+$  lediglich eine abkürzende Schreibweise für  $M \cdot M^*$ .



# Einfache Mengen sind regulär!

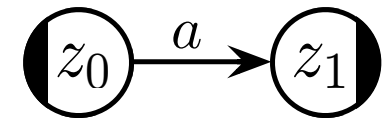
- Die leere Menge ist regulär.



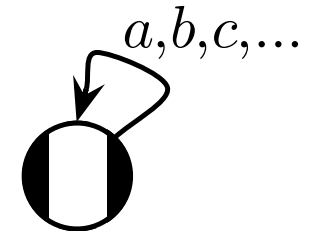
- Die Menge  $\emptyset^* = \{\lambda\}$  ist regulär.



- Die Menge  $\{a\}$  ist regulär.



- $\Sigma^*$  ist regulär.



- Für jede Menge  $M$  ist  $M^+$  lediglich eine abkürzende Schreibweise für  $M \cdot M^*$ .

- Für jede Menge  $M$  ist  $M \cdot \emptyset = \emptyset \cdot M = \emptyset$ .



# Satz von Kleene

- Wegen der Abgeschlossenheit von  $\mathcal{Reg}$  gegen  $\cup$ ,  $\cdot$  und  $*$  ist klar, dass  $\mathcal{Rat} \subseteq \mathcal{Reg}$  gilt.



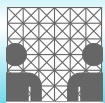
# Satz von Kleene

- Wegen der Abgeschlossenheit von  $\mathcal{Reg}$  gegen  $\cup$ ,  $\cdot$  und  $*$  ist klar, dass  $\mathcal{Rat} \subseteq \mathcal{Reg}$  gilt.
- **Theorem:** Für jedes Alphabet  $\Sigma$  gilt  $\mathcal{Akz}(\Sigma) = \mathcal{Rat}(\Sigma)$  und folglich  $\mathcal{Reg} = \mathcal{Rat}$ .



# Satz von Kleene

- Wegen der Abgeschlossenheit von  $\mathcal{Reg}$  gegen  $\cup$ ,  $\cdot$  und  $*$  ist klar, dass  $\mathcal{Rat} \subseteq \mathcal{Reg}$  gilt.
- **Theorem:** Für jedes Alphabet  $\Sigma$  gilt  $\mathcal{Akz}(\Sigma) = \mathcal{Rat}(\Sigma)$  und folglich  $\mathcal{Reg} = \mathcal{Rat}$ .
- **Beweisidee:** Wir stellen eine Rekursionsformel auf, die von jeder regulären Sprache zeigt, dass sie durch endlich häufiges Anwenden von  $\cup$ ,  $\cdot$  und  $*$  darstellbar ist.



# Satz von Kleene

- Wegen der Abgeschlossenheit von  $\mathcal{R}eg$  gegen  $\cup$ ,  $\cdot$  und  $*$  ist klar, dass  $\mathcal{R}at \subseteq \mathcal{R}eg$  gilt.
- **Theorem:** Für jedes Alphabet  $\Sigma$  gilt  $\mathcal{A}kz(\Sigma) = \mathcal{R}at(\Sigma)$  und folglich  $\mathcal{R}eg = \mathcal{R}at$ .
- **Beweisidee:** Wir stellen eine Rekursionsformel auf, die von jeder regulären Sprache zeigt, dass sie durch endlich häufiges Anwenden von  $\cup$ ,  $\cdot$  und  $*$  darstellbar ist.
  - Konstruktion eines rationalen Ausdruckes für einen beliebigen DFA.



# Satz von Kleene

- Wegen der Abgeschlossenheit von  $\mathcal{R}eg$  gegen  $\cup$ ,  $\cdot$  und  $*$  ist klar, dass  $\mathcal{R}at \subseteq \mathcal{R}eg$  gilt.
- **Theorem:** Für jedes Alphabet  $\Sigma$  gilt  $\mathcal{A}kz(\Sigma) = \mathcal{R}at(\Sigma)$  und folglich  $\mathcal{R}eg = \mathcal{R}at$ .
- **Beweisidee:** Wir stellen eine Rekursionsformel auf, die von jeder regulären Sprache zeigt, dass sie durch endlich häufiges Anwenden von  $\cup$ ,  $\cdot$  und  $*$  darstellbar ist.
  - Konstruktion eines rationalen Ausdruckes für einen beliebigen DFA.
  - Konstruktion funktioniert auch für (buchstabierenden) NFA.





$$Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$$

- Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA



$$Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$$

- Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA
- Sei für  $0 \leq k \leq n$  und  $1 \leq i, j \leq n$ :



# $Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$

- Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA
- Sei für  $0 \leq k \leq n$  und  $1 \leq i, j \leq n$ :

$$R_{i,j}^k := \left\{ w \in \Sigma^* \left| \begin{array}{l} \delta(z_i, w) = z_j \wedge [(w = uv \wedge u \neq \lambda \wedge \\ v \neq \lambda \wedge \delta(z_i, u) = z_r) \rightarrow r \leq k] \end{array} \right. \right\}$$



# $Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$

• Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA

• Sei für  $0 \leq k \leq n$  und  $1 \leq i, j \leq n$ :

$$R_{i,j}^k := \left\{ w \in \Sigma^* \left| \begin{array}{l} \delta(z_i, w) = z_j \wedge [(w = uv \wedge u \neq \lambda \wedge \\ v \neq \lambda \wedge \delta(z_i, u) = z_r) \rightarrow r \leq k] \end{array} \right. \right\}$$

•  $R_{i,j}^k$  enthält alle Wörter, die auf Pfaden von  $z_i$  nach  $z_j$  über die Menge  $\{z_1, \dots, z_k\}$  von **Zwischenzuständen** gelesen werden können.



$$Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$$

• Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA

• Sei für  $0 \leq k \leq n$  und  $1 \leq i, j \leq n$ :

$$R_{i,j}^k := \left\{ w \in \Sigma^* \left| \begin{array}{l} \delta(z_i, w) = z_j \wedge [(w = uv \wedge u \neq \lambda \wedge \\ v \neq \lambda \wedge \delta(z_i, u) = z_r) \rightarrow r \leq k] \end{array} \right. \right\}$$

•  $R_{i,j}^k$  enthält alle Wörter, die auf Pfaden von  $z_i$  nach  $z_j$  über die Menge  $\{z_1, \dots, z_k\}$  von **Zwischenzuständen** gelesen werden können.

• Insbesondere  $R_{i,j}^0 = \{x \in \Sigma \cup \{\lambda\} \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}$



# $Akz(\Sigma) \subseteq \mathcal{Rat}(\Sigma)$

- Sei  $A = (\{z_1, \dots, z_n\}, \Sigma, \delta, z_1, Z_{\text{end}})$  ein DFA

- Sei für  $0 \leq k \leq n$  und  $1 \leq i, j \leq n$ :

$$R_{i,j}^k := \left\{ w \in \Sigma^* \left| \begin{array}{l} \delta(z_i, w) = z_j \wedge [(w = uv \wedge u \neq \lambda \wedge \\ v \neq \lambda \wedge \delta(z_i, u) = z_r) \rightarrow r \leq k] \end{array} \right. \right\}$$

- $R_{i,j}^k$  enthält alle Wörter, die auf Pfaden von  $z_i$  nach  $z_j$  über die Menge  $\{z_1, \dots, z_k\}$  von **Zwischenzuständen** gelesen werden können.
- Insbesondere  $R_{i,j}^0 = \{x \in \Sigma \cup \{\lambda\} \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}$
- Somit:  $L(A) = \bigcup_{z_j \in Z_{\text{end}}} R_{1,j}^n$



# Beweis: $R_{i,j}^k$ -Rekursionsformel

$$R_{i,j}^k = \begin{cases} \{x \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}, & \text{falls } k = 0 \\ R_{i,j}^{k-1} \cup R_{i,k}^{k-1} \cdot (R_{k,k}^{k-1})^* \cdot R_{k,j}^{k-1}, & \text{falls } k \geq 1. \end{cases}$$

● **Induktions-Basis:**  $R_{i,j}^0$  entspricht der Definition



# Beweis: $R_{i,j}^k$ -Rekursionsformel

$$R_{i,j}^k = \begin{cases} \{x \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}, & \text{falls } k = 0 \\ R_{i,j}^{k-1} \cup R_{i,k}^{k-1} \cdot (R_{k,k}^{k-1})^* \cdot R_{k,j}^{k-1}, & \text{falls } k \geq 1. \end{cases}$$

- **Induktions-Basis:**  $R_{i,j}^0$  entspricht der Definition
- **Induktions-Schritt:** Sei  $k = m \geq 0$ .  
 $R_{i,j}^m$  korrekt  $\Rightarrow R_{i,j}^{m+1}$  korrekt

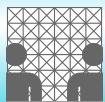




# Beweis: $R_{i,j}^k$ -Rekursionsformel

$$R_{i,j}^k = \begin{cases} \{x \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}, & \text{falls } k = 0 \\ R_{i,j}^{k-1} \cup R_{i,k}^{k-1} \cdot (R_{k,k}^{k-1})^* \cdot R_{k,j}^{k-1}, & \text{falls } k \geq 1. \end{cases}$$

- **Induktions-Basis:**  $R_{i,j}^0$  entspricht der Definition
- **Induktions-Schritt:** Sei  $k = m \geq 0$ .  
 $R_{i,j}^m$  korrekt  $\Rightarrow R_{i,j}^{m+1}$  korrekt
  - Kommt  $z_{m+1}$  nicht vor  $\Rightarrow w \in R_{i,j}^m$ .



# Beweis: $R_{i,j}^k$ -Rekursionsformel

$$R_{i,j}^k = \begin{cases} \{x \mid \delta(z_i, x) = z_j \text{ oder } (x = \lambda) \wedge (i = j)\}, & \text{falls } k = 0 \\ R_{i,j}^{k-1} \cup R_{i,k}^{k-1} \cdot (R_{k,k}^{k-1})^* \cdot R_{k,j}^{k-1}, & \text{falls } k \geq 1. \end{cases}$$

● **Induktions-Basis:**  $R_{i,j}^0$  entspricht der Definition

● **Induktions-Schritt:** Sei  $k = m \geq 0$ .

$R_{i,j}^m$  korrekt  $\Rightarrow R_{i,j}^{m+1}$  korrekt

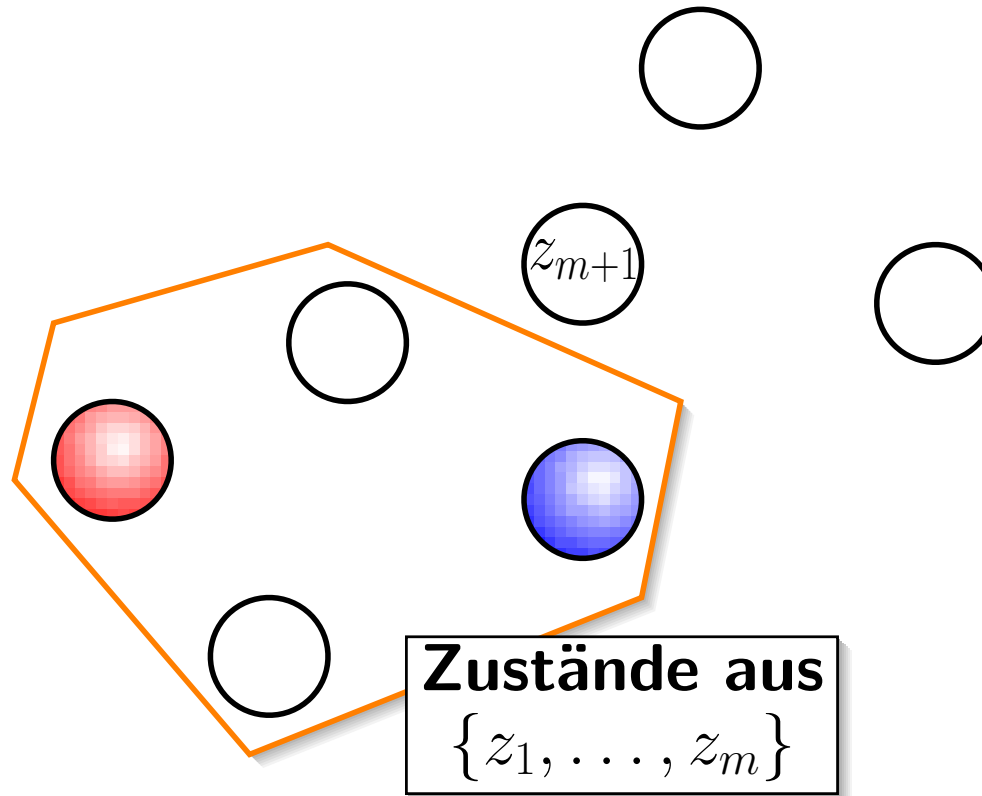
● Kommt  $z_{m+1}$  nicht vor  $\Rightarrow w \in R_{i,j}^m$ .

● Kommt  $z_{m+1}$  auf dem Pfad vor  $\Rightarrow$  Zerlegung:

$$z_i \xrightarrow[u]{*} z_{m+1} \xrightarrow[v_1]{*} z_{m+1} \xrightarrow[v_2]{*} \dots \xrightarrow[v_r]{*} z_{m+1} \xrightarrow[w]{*} z_j$$

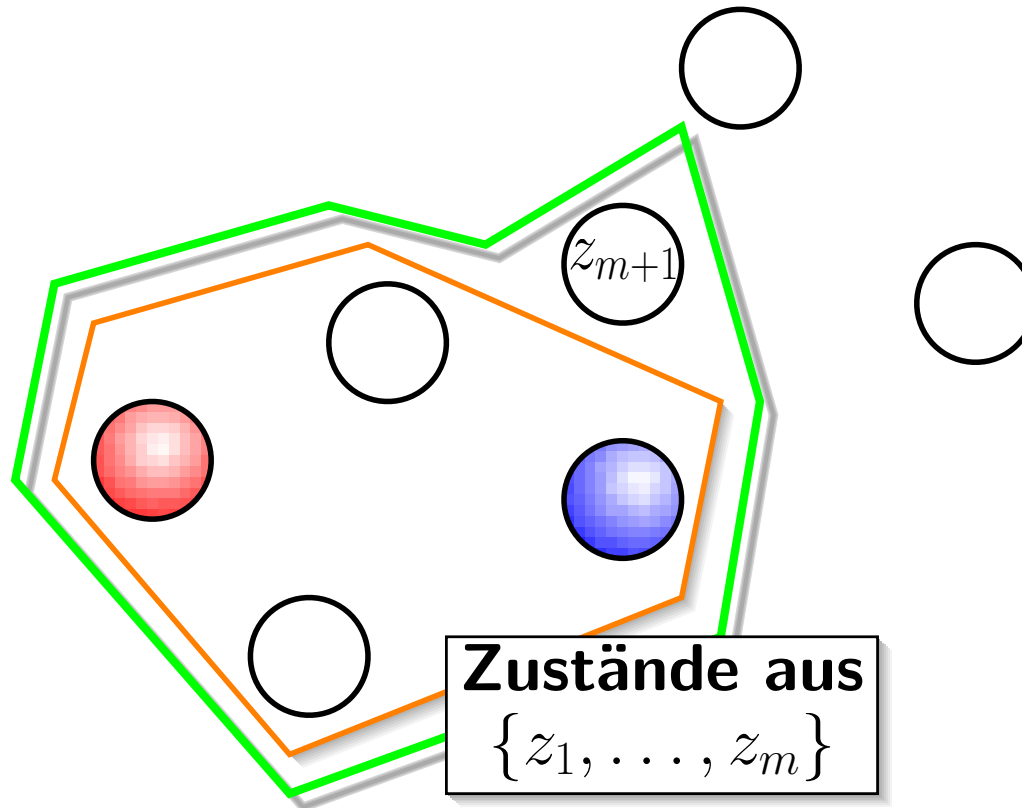


# Pfade in einem FA



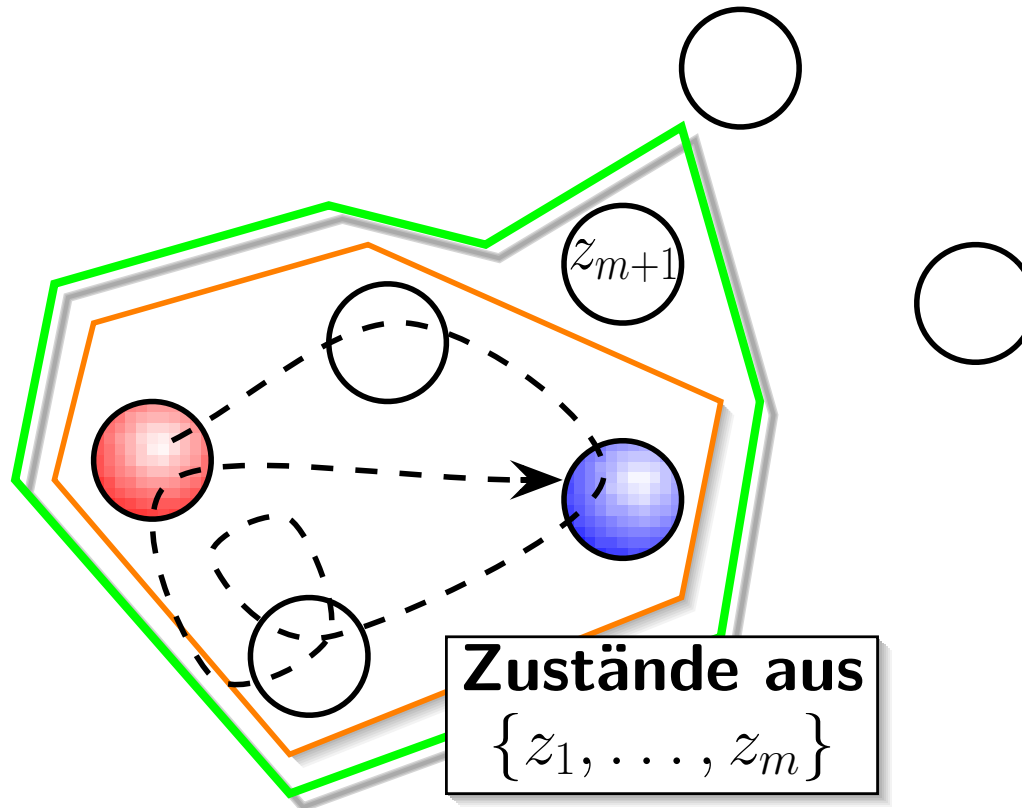


# Pfade in einem FA





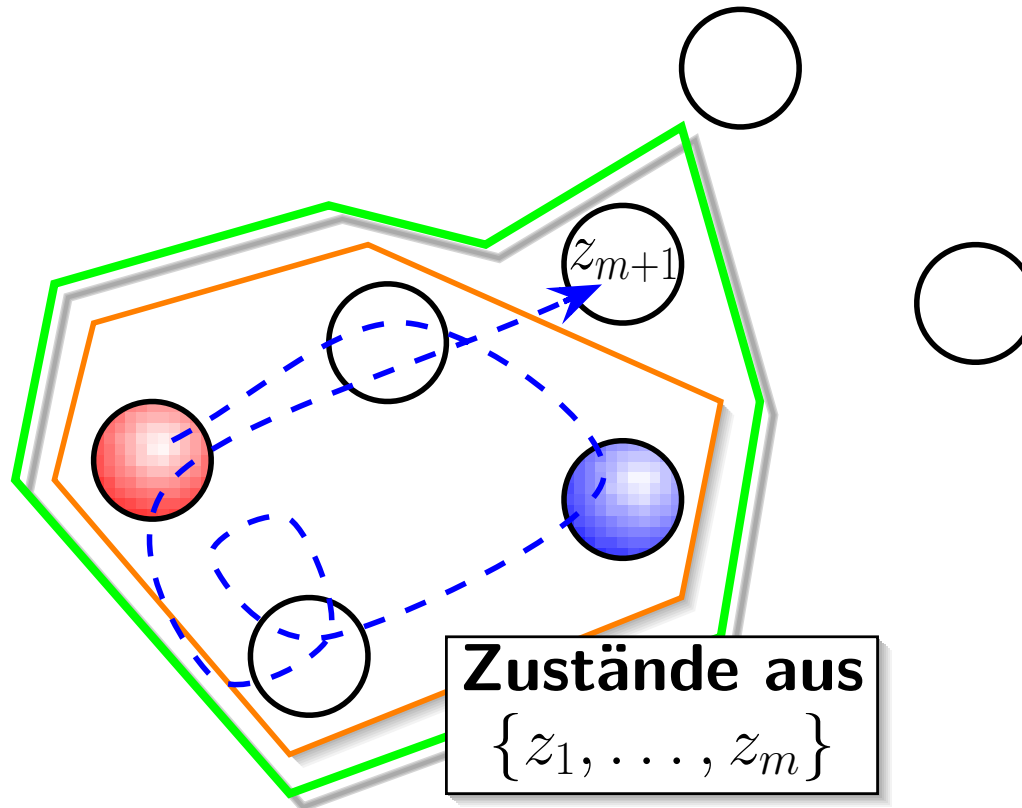
# Pfade in einem FA



$R_{i,j}^m$



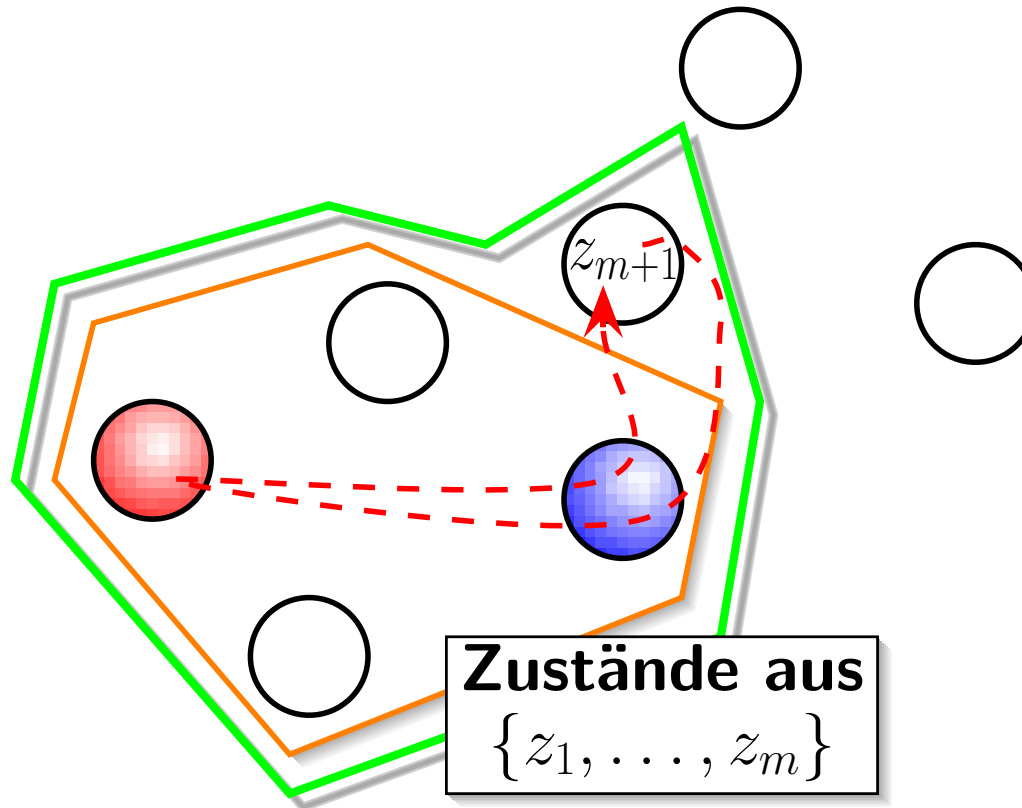
# Pfade in einem FA



$$R_{i,m+1}^m$$



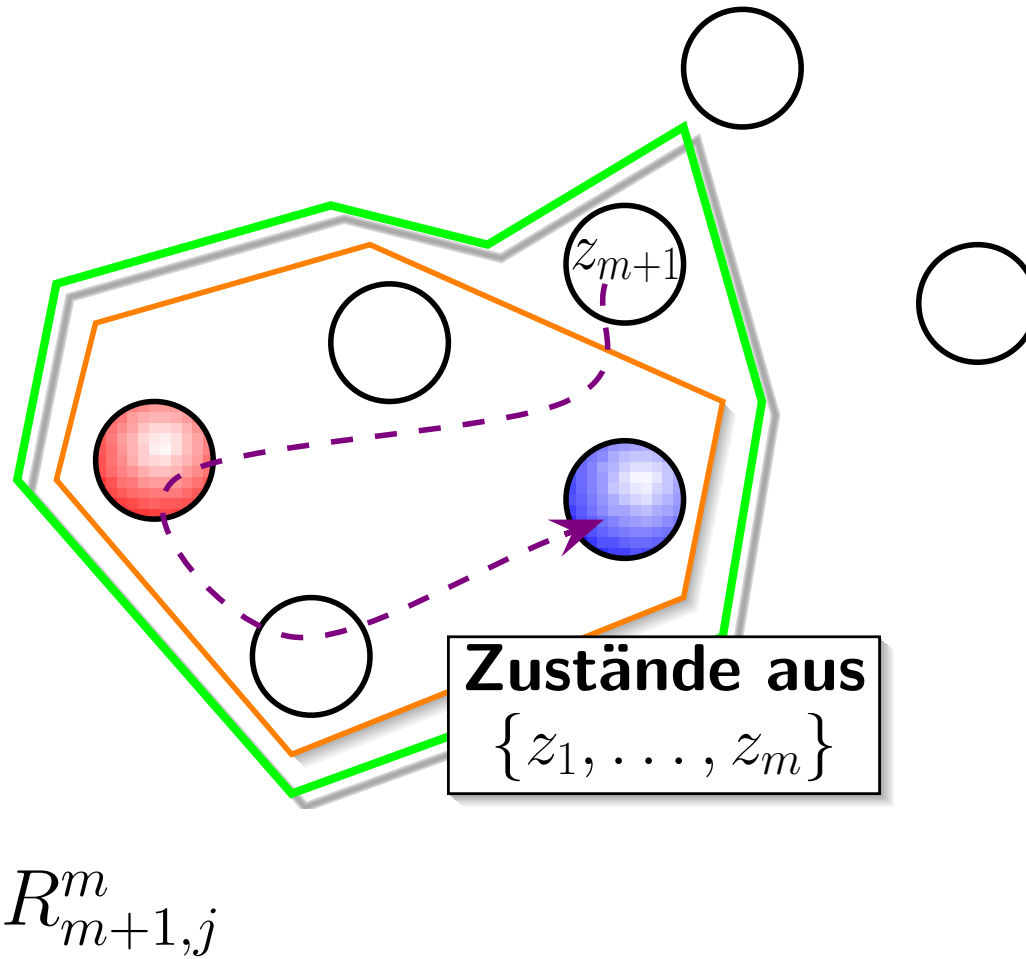
# Pfade in einem FA



$$(R_{m+1, m+1}^m)^*$$



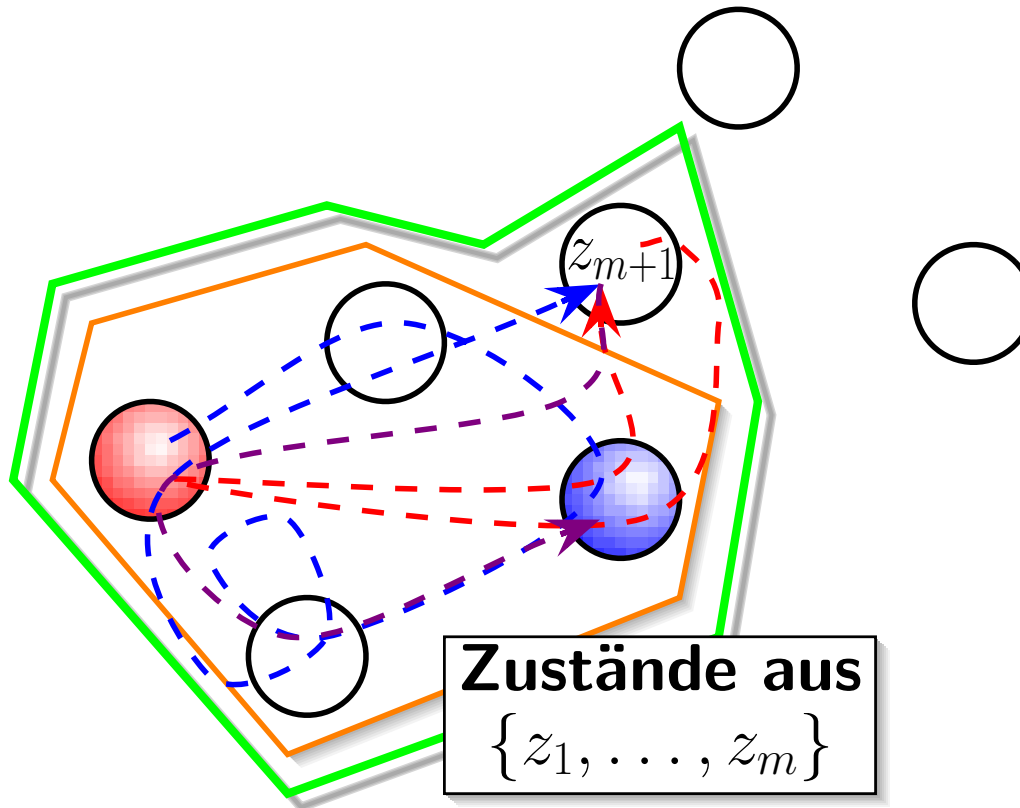
# Pfade in einem FA







# Pfade in einem FA



$$R_{i,j}^m \cup R_{i,m+1}^m \cdot (R_{m+1,m+1}^m)^* \cdot R_{m+1,j}^m$$



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:
  - $u \in R_{i,m+1}^m$ ,



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:
  - $u \in R_{i,m+1}^m$ ,
  - $\forall 1 \leq i \leq r : v_i \in R_{m+1,m+1}^m$  und



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:
  - $u \in R_{i, m+1}^m$ ,
  - $\forall 1 \leq i \leq r : v_i \in R_{m+1, m+1}^m$  und
  - $w \in R_{m+1, j}^m$ .



# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:
  - $u \in R_{i,m+1}^m$ ,
  - $\forall 1 \leq i \leq r : v_i \in R_{m+1,m+1}^m$  und
  - $w \in R_{m+1,j}^m$ .
- Dies zeigt zunächst die Inklusion:
$$R_{i,j}^{m+1} \subseteq R_{i,j}^m \cup R_{i,m+1}^m \cdot (R_{m+1,m+1}^m)^* \cdot R_{m+1,j}^m.$$



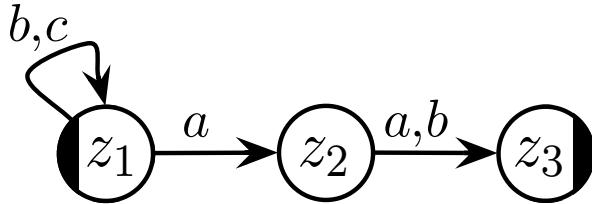
# Beweis (Forts.)

- Die Pfade mit den Wörtern  $u, v_1, v_2, \dots, v_r, w$  durchlaufen nur Zustände der Menge  $\{z_1, \dots, z_m\}$ .
- Offensichtlich gilt:
  - $u \in R_{i,m+1}^m$ ,
  - $\forall 1 \leq i \leq r : v_i \in R_{m+1,m+1}^m$  und
  - $w \in R_{m+1,j}^m$ .
- Dies zeigt zunächst die Inklusion:
$$R_{i,j}^{m+1} \subseteq R_{i,j}^m \cup R_{i,m+1}^m \cdot (R_{m+1,m+1}^m)^* \cdot R_{m+1,j}^m.$$
- Die Umkehrung dieser Inklusion ist leicht ersichtlich.





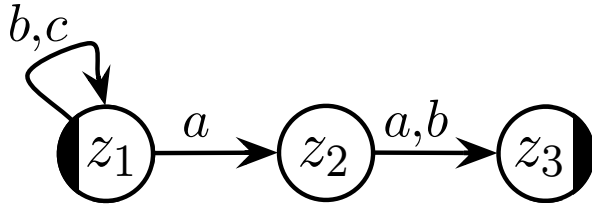
# Beispiel: 1. ohne Zwischenzustand



$$R_{1,1}^0 = \{\lambda, b, c\}$$



# Beispiel: 1. ohne Zwischenzustand

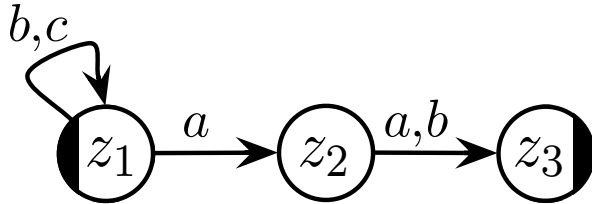


$$R_{1,1}^0 = \{\lambda, b, c\}$$

$$R_{1,2}^0 = \{a\}$$



# Beispiel: 1. ohne Zwischenzustand



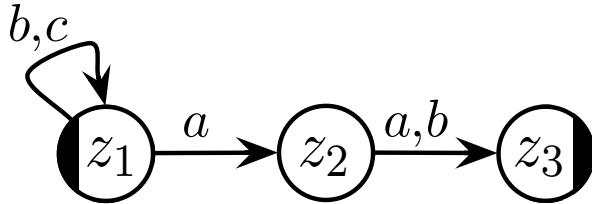
$$R_{1,1}^0 = \{\lambda, b, c\}$$

$$R_{1,2}^0 = \{a\}$$

$$R_{1,3}^0 = R_{2,1}^0 = R_{3,1}^0 = R_{3,2}^0 = \emptyset$$



# Beispiel: 1. ohne Zwischenzustand



$$R_{1,1}^0 = \{\lambda, b, c\}$$

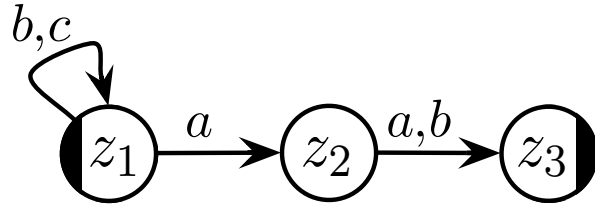
$$R_{1,2}^0 = \{a\}$$

$$R_{1,3}^0 = R_{2,1}^0 = R_{3,1}^0 = R_{3,2}^0 = \emptyset$$

$$R_{2,2}^0 = \{\lambda\}$$



# Beispiel: 1. ohne Zwischenzustand



$$R_{1,1}^0 = \{\lambda, b, c\}$$

$$R_{1,2}^0 = \{a\}$$

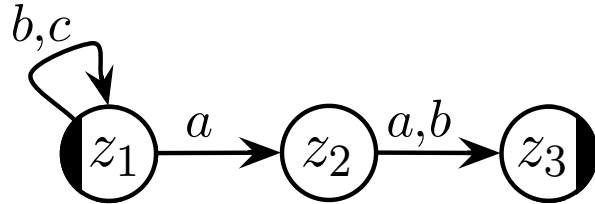
$$R_{1,3}^0 = R_{2,1}^0 = R_{3,1}^0 = R_{3,2}^0 = \emptyset$$

$$R_{2,2}^0 = \{\lambda\}$$

$$R_{2,3}^0 = \{a, b\}$$



# Beispiel: 1. ohne Zwischenzustand



$$R_{1,1}^0 = \{\lambda, b, c\}$$

$$R_{1,2}^0 = \{a\}$$

$$R_{1,3}^0 = R_{2,1}^0 = R_{3,1}^0 = R_{3,2}^0 = \emptyset$$

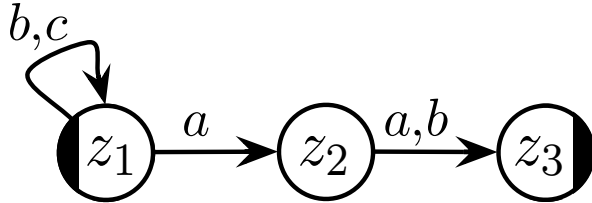
$$R_{2,2}^0 = \{\lambda\}$$

$$R_{2,3}^0 = \{a, b\}$$

$$R_{3,3}^0 = \{\lambda\}$$



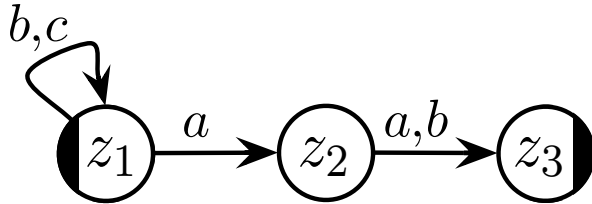
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$R_{1,1}^1 = R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0$$



## Beispiel: 2. Zwischenzustände $\{z_1\}$

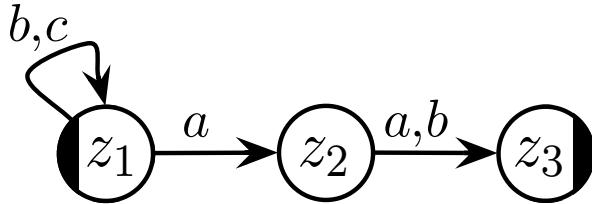


$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \end{aligned}$$





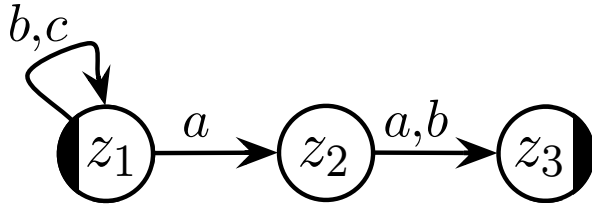
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \end{aligned}$$



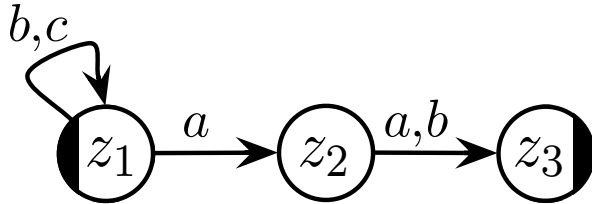
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \\ &= R_{1,1}^0 \cdot (R_{1,1}^0)^* \end{aligned}$$



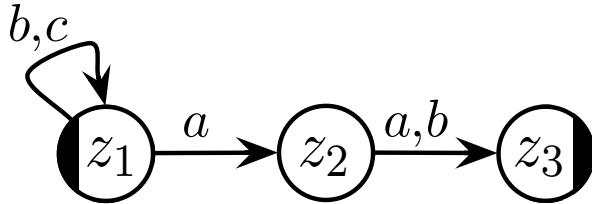
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \\ &= R_{1,1}^0 \cdot (R_{1,1}^0)^* \\ &= (R_{1,1}^0)^+ \end{aligned}$$



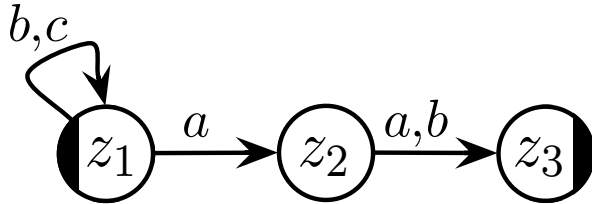
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \\ &= R_{1,1}^0 \cdot (R_{1,1}^0)^* \\ &= (R_{1,1}^0)^+ \\ &= \{\lambda, b, c\}^+ \end{aligned}$$



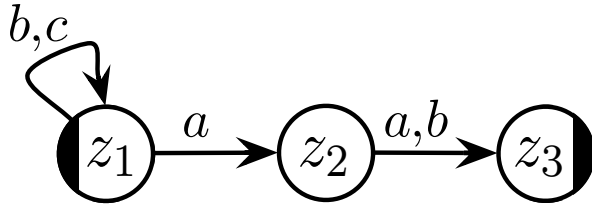
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \\ &= R_{1,1}^0 \cdot (R_{1,1}^0)^* \\ &= (R_{1,1}^0)^+ \\ &= \{\lambda, b, c\}^+ \\ &= \{b, c\}^* \end{aligned}$$



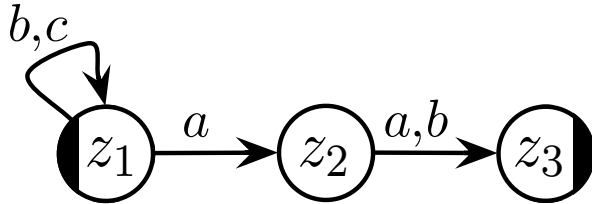
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,1}^1 &= R_{1,1}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,1}^0 \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^* \cdot R_{1,1}^0) \\ &= R_{1,1}^0 \cdot (\{\lambda\} \cup (R_{1,1}^0)^+) \\ &= R_{1,1}^0 \cdot (R_{1,1}^0)^* \\ &= (R_{1,1}^0)^+ \\ &= \{\lambda, b, c\}^+ \\ &= \{b, c\}^* \\ &= (b + c)^* \end{aligned}$$



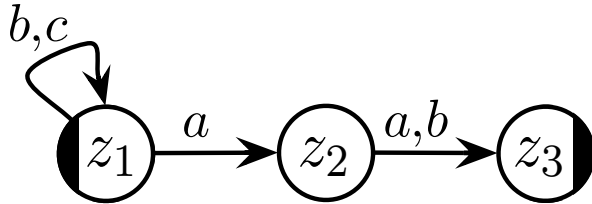
## Beispiel: 2. Zwischenzustände $\{z_1\}$



$$\begin{aligned} R_{1,2}^1 &= R_{1,2}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_0^{1,2} \\ &= \{a\} \cup \{\lambda, b, c\} \cdot \{\lambda, b, c\}^* \cdot \{a\} \\ &= \{a\} \cup \{\lambda, b, c\}^+ \cdot \{a\} \\ &= \{b, c\}^* \cdot \{a\} \\ &= (b + c)^* a \end{aligned}$$



## Beispiel: 2. Zwischenzustände $\{z_1\}$



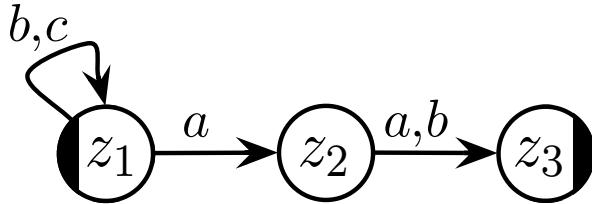
$$\begin{aligned} R_{1,2}^1 &= R_{1,2}^0 \cup R_{1,1}^0 \cdot (R_{1,1}^0)^* \cdot R_0^{1,2} \\ &= \{a\} \cup \{\lambda, b, c\} \cdot \{\lambda, b, c\}^* \cdot \{a\} \\ &= \{a\} \cup \{\lambda, b, c\}^+ \cdot \{a\} \\ &= \{b, c\}^* \cdot \{a\} \\ &= (b + c)^* a \end{aligned}$$

$$\begin{aligned} R_{2,3}^1 &= R_{2,3}^0 \cup R_{2,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,3}^0 \\ &= (a + b) \end{aligned}$$





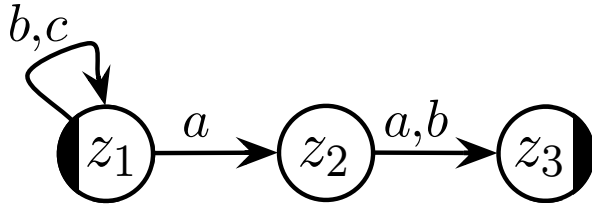
# Beispiel: 3. alle Zwischenzustände



$$R_{1,3}^3 = R_{1,3}^2 \cup R_{1,3}^2 \cdot (R_{3,3}^2)^* \cdot R_{3,3}^2$$



# Beispiel: 3. alle Zwischenzustände

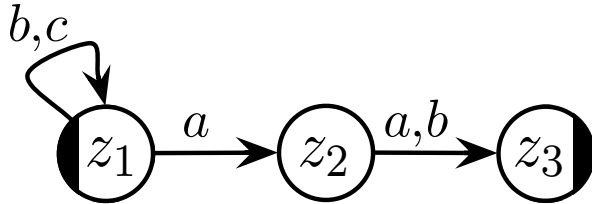


$$R_{1,3}^3 = R_{1,3}^2 \cup R_{1,3}^2 \cdot (R_{3,3}^2)^* \cdot R_{3,3}^2$$

$$\begin{aligned} R_{3,3}^2 &= R_{3,3}^1 \cup R_{3,2}^1 \cdot (R_{2,2}^1)^* \cdot R_{2,3}^1 \\ &= R_{3,3}^1 \cup \emptyset \cdot \dots \end{aligned}$$



# Beispiel: 3. alle Zwischenzustände



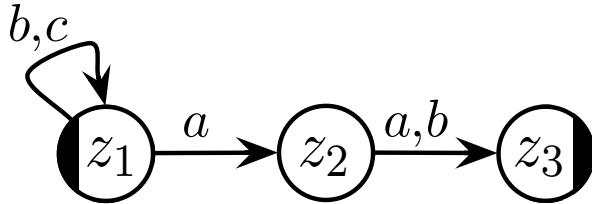
$$R_{1,3}^3 = R_{1,3}^2 \cup R_{1,3}^2 \cdot (R_{3,3}^2)^* \cdot R_{3,3}^2$$

$$\begin{aligned} R_{3,3}^2 &= R_{3,3}^1 \cup R_{3,2}^1 \cdot (R_{2,2}^1)^* \cdot R_{2,3}^1 \\ &= R_{3,3}^1 \cup \emptyset \cdot \dots \end{aligned}$$

$$\begin{aligned} R_{3,3}^1 &= R_{3,3}^0 \cup R_{3,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,3}^0 \\ &= \{\lambda\} \cup \emptyset \cdot \dots \\ &= \{\lambda\} \end{aligned}$$



# Beispiel: 3. alle Zwischenzustände



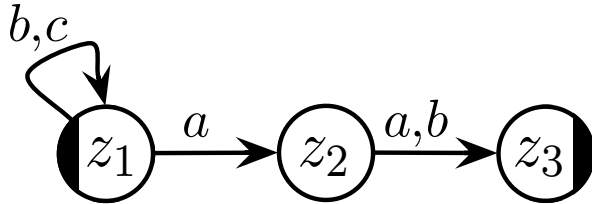
$$\begin{aligned} R_{1,3}^3 &= R_{1,3}^2 \cup R_{1,3}^2 \cdot (R_{3,3}^2)^* \cdot R_{3,3}^2 \\ &= R_{1,3}^2 \cup R_{1,3}^2 \cdot \{\lambda\}^* \cdot \{\lambda\} \\ &= R_{1,3}^2 \end{aligned}$$

$$\begin{aligned} R_{3,3}^2 &= R_{3,3}^1 \cup R_{3,2}^1 \cdot (R_{2,2}^1)^* \cdot R_{2,3}^1 \\ &= R_{3,3}^1 \cup \emptyset \cdot \dots \end{aligned}$$

$$\begin{aligned} R_{3,3}^1 &= R_{3,3}^0 \cup R_{3,1}^0 \cdot (R_{1,1}^0)^* \cdot R_{1,3}^0 \\ &= \{\lambda\} \cup \emptyset \cdot \dots \\ &= \{\lambda\} \end{aligned}$$



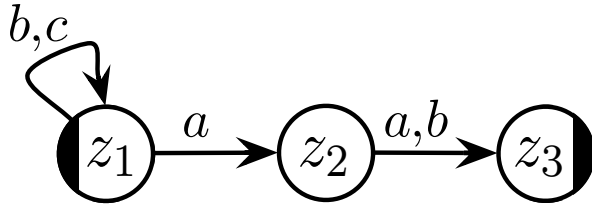
# Beispiel: 3. alle Zwischenzustände



Also  $L(A) = R_{1,3}^3 = R_{1,3}^2$  mit



# Beispiel: 3. alle Zwischenzustände

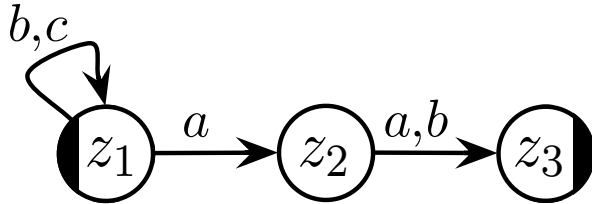


Also  $L(A) = R_{1,3}^3 = R_{1,3}^2$  mit

$$\begin{aligned} R_{1,3}^2 &= R_{1,3}^1 \cup R_{1,2}^1 \cdot (R_{2,2}^1)^* \cdot R_{2,3}^1 \\ &= \emptyset + (b + c)^*(a)(a + b) \\ &= (b + c)^*(a)(a + b) \\ &= L(A) \end{aligned}$$



# Beispiel: 3. alle Zwischenzustände



Also  $L(A) = R_{1,3}^3 = R_{1,3}^2$  mit

$$\begin{aligned} R_{1,3}^2 &= R_{1,3}^1 \cup R_{1,2}^1 \cdot (R_{2,2}^1)^* \cdot R_{2,3}^1 \\ &= \emptyset + (b + c)^*(a)(a + b) \\ &= (b + c)^*(a)(a + b) \\ &= L(A) \end{aligned}$$

...hier hätte man den rationalen Ausdruck auch direkt vom Automaten „ablesen“ können!