

Übung 3: Reguläre Ausdrücke und Minimalautomaten

Theoretische Informatik Sommersemester 2014

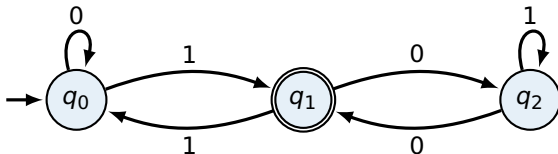
Markus Kaiser

3. Mai 2014

Definition (Deterministischer endlicher Automat)

Ein DFA ist ein Tupel $M = (Q, \Sigma, \delta, q_0, F)$ aus einer/einem

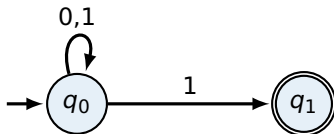
- endlichen Menge von Zuständen Q
- endlichen Eingabealphabet Σ
- totalen Übergangsfunktion $\delta : Q \times \Sigma \rightarrow Q$
- Startzustand $q_0 \in Q$
- Menge von Endzuständen $F \subseteq Q$



Definition (Nicht-Deterministischer endlicher Automat)

Ein NFA ist ein Tupel $N = (Q, \Sigma, \delta, S, F)$ mit

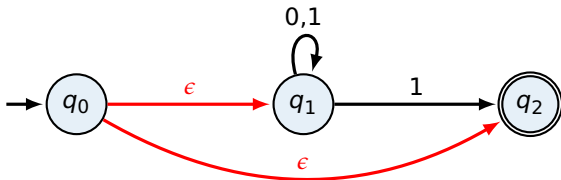
- Q, Σ, F wie ein DFA
- Menge von Startzuständen $S \subseteq F$
- Übergangsfunktion $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$



Definition (NFA mit ϵ -Übergängen)

Ein ϵ -NFA ist ein Tupel $N = (Q, \Sigma, \delta, S, F)$ mit

- Q, Σ, F wie ein DFA
- Menge von **Startzuständen** $S \subseteq F$
- **Übergangsfunktion** $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$



Definition (Regulärer Ausdruck)

Reguläre Ausdrücke sind induktiv definiert

- \emptyset ist ein regulärer Ausdruck
- ϵ ist ein regulärer Ausdruck
- Für alle $a \in \Sigma$ ist a ein regulärer Ausdruck
- Sind α und β reguläre Ausdrücke, dann auch

Konkatenation $\alpha\beta$

Veroderung $\alpha \mid \beta$

Wiederholung α^*

Analoge Sprachdefinition, z.B. $L(\alpha\beta) = L(\alpha)L(\beta)$

Beispiel

- $\alpha = (0|1)^*00$
- Worte bestehen aus einer beliebigen Folge von Einsen und Nullen gefolgt von zwei Nullen.
- $L(\alpha) \supseteq \{x \mid x \text{ Binärzahl, } x \bmod 4 = 0\}$

Idee (Kleene)

Für einen Ausdruck γ wird rekursiv mit struktureller Induktion ein ϵ -NFA konstruiert.

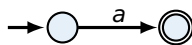
$$\gamma = \emptyset$$



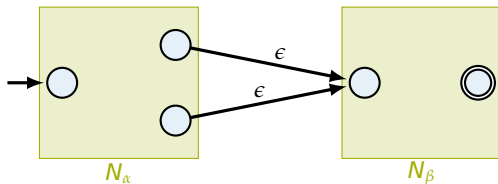
$$\gamma = \epsilon$$



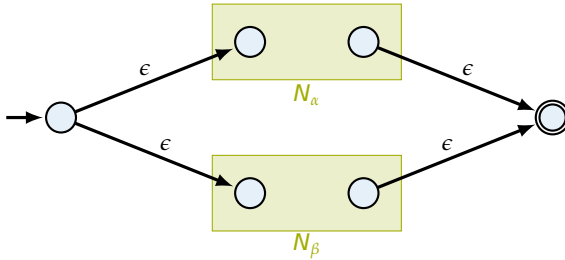
$$\gamma = a \in \Sigma$$



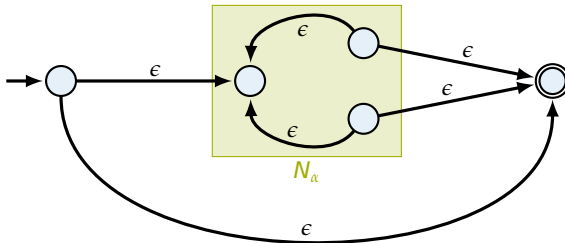
$$\gamma = \alpha\beta$$



$$\gamma = \alpha \mid \beta$$



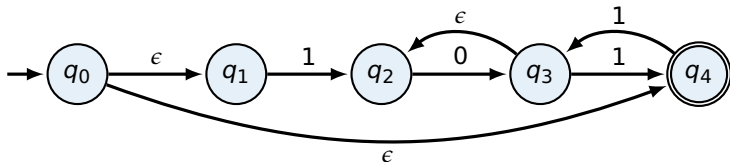
$$\gamma = \alpha^*$$



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

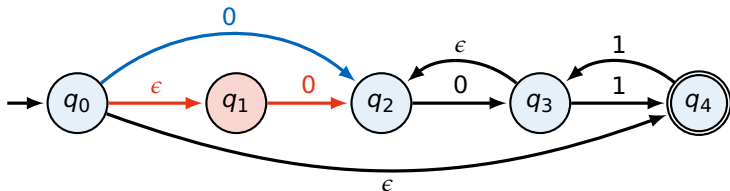
- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle **ϵ -Kanten** und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

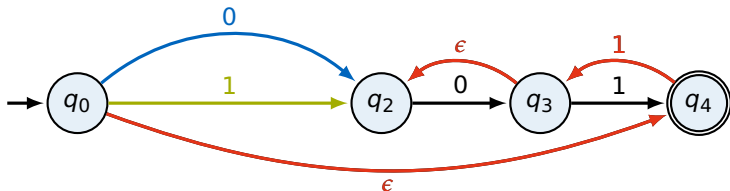
- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle **ϵ -Kanten** und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

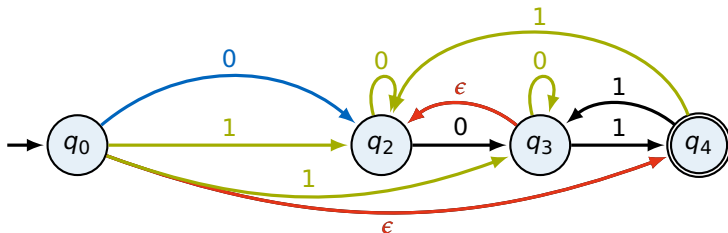
- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle ϵ -Kanten und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

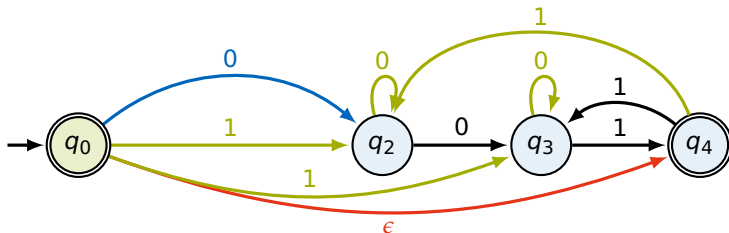
- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle **ϵ -Kanten** und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

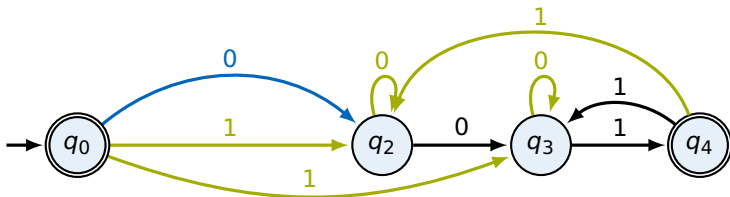
- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle **ϵ -Kanten** und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Idee

Entferne ϵ -Kanten durch das Bilden von ϵ -Hüllen.

- 1 Entferne **unnötige Knoten**.
- 2 Für jeden **Pfad** der Form $\epsilon \dots \epsilon a \epsilon \dots \epsilon$ verbinde Anfangs- und Endknoten mit einer **a -Kante**.
- 3 Entferne alle **ϵ -Kanten** und unerreichbare Knoten.
- 4 Wurde das leere Wort akzeptiert mache den **Anfangszustand** zum Endzustand.



Potenzmengenkonstruktion

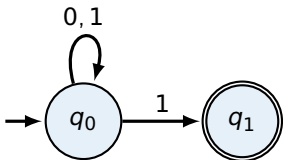
Konstruiere einen Automaten, der **alle möglichen Pfade** gleichzeitig berücksichtigt. Gegeben ein NFA $(Q, \Sigma, \delta, S, F)$, konstruiere einen DFA mit Zuständen aus $\mathcal{P}(Q)$.

- Starte in $\{S\}$
- Die Übergangsfunktion speichert **alle möglichen Schritte**

$$\bar{\delta}: \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$$

$$(M, a) \mapsto \bigcup_{q \in M} \delta(q, a)$$

- M ist Endzustand wenn $F \cap M \neq \emptyset$



Potenzmengenkonstruktion

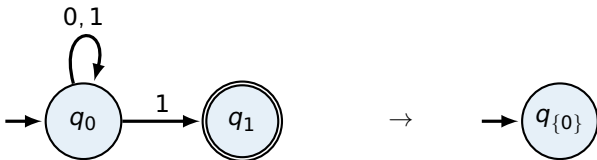
Konstruiere einen Automaten, der **alle möglichen Pfade** gleichzeitig berücksichtigt. Gegeben ein NFA $(Q, \Sigma, \delta, S, F)$, konstruiere einen DFA mit Zuständen aus $\mathcal{P}(Q)$.

- Starte in $\{S\}$
- Die Übergangsfunktion speichert **alle möglichen Schritte**

$$\bar{\delta}: \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$$

$$(M, a) \mapsto \bigcup_{q \in M} \delta(q, a)$$

- M ist Endzustand wenn $F \cap M \neq \emptyset$



Potenzmengenkonstruktion

Konstruiere einen Automaten, der **alle möglichen Pfade** gleichzeitig berücksichtigt. Gegeben ein NFA $(Q, \Sigma, \delta, S, F)$, konstruiere einen DFA mit Zuständen aus $\mathcal{P}(Q)$.

- Starte in $\{S\}$
- Die Übergangsfunktion speichert **alle möglichen Schritte**

$$\bar{\delta}: \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$$

$$(M, a) \mapsto \bigcup_{q \in M} \delta(q, a)$$

- M ist Endzustand wenn $F \cap M \neq \emptyset$



Potenzmengenkonstruktion

Konstruiere einen Automaten, der **alle möglichen Pfade** gleichzeitig berücksichtigt. Gegeben ein NFA $(Q, \Sigma, \delta, S, F)$, konstruiere einen DFA mit Zuständen aus $\mathcal{P}(Q)$.

- Starte in $\{S\}$
- Die Übergangsfunktion speichert **alle möglichen Schritte**

$$\bar{\delta} : \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$$

$$(M, a) \mapsto \bigcup_{q \in M} \delta(q, a)$$

- M ist Endzustand wenn $F \cap M \neq \emptyset$



Definition (Äquivalente Worte)

Jede Sprache $L \subseteq \Sigma^*$ induziert eine **Äquivalenzrelation** $\equiv_L \subseteq \Sigma^* \times \Sigma^*$

$$u \equiv_L v \iff (\forall w \in \Sigma^*. uw \in L \iff vw \in L)$$

Definition (Äquivalente Zustände)

Zwei Zustände im DFA A sind **äquivalent** wenn sie die selbe Sprache akzeptieren.

$$p \equiv_A q \iff (\forall w \in \Sigma^*. \hat{\delta}(p, w) \in F \iff \hat{\delta}(q, w) \in F)$$

Definition (Äquivalente Worte)

Jede Sprache $L \subseteq \Sigma^*$ induziert eine **Äquivalenzrelation** $\equiv_L \subseteq \Sigma^* \times \Sigma^*$

$$u \equiv_L v \iff (\forall w \in \Sigma^*. uw \in L \iff vw \in L)$$

Definition (Äquivalente Zustände)

Zwei Zustände im DFA A sind **äquivalent** wenn sie die selbe Sprache akzeptieren.

$$p \equiv_A q \iff (\forall w \in \Sigma^*. \hat{\delta}(p, w) \in F \iff \hat{\delta}(q, w) \in F)$$

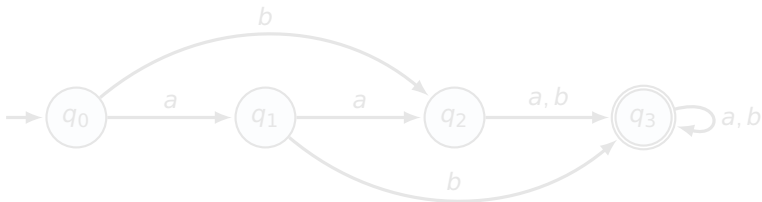
Definition (Unterscheidbarkeit)

Zwei Zustände sind **unterscheidbar**, wenn sie unterschiedliche Sprachen akzeptieren.

$$p \not\equiv_A q \iff (\exists w \in \Sigma^*. \delta(p, w) \in F \wedge \delta(q, w) \notin F)$$

Satz

Sind $\delta(p, a)$ und $\delta(q, a)$ unterscheidbar, dann auch p und q .



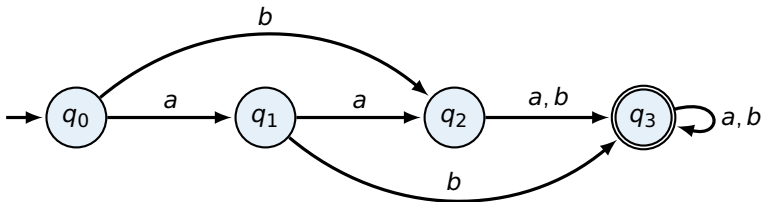
Definition (Unterscheidbarkeit)

Zwei Zustände sind **unterscheidbar**, wenn sie unterschiedliche Sprachen akzeptieren.

$$p \not\equiv_A q \iff (\exists w \in \Sigma^*. \delta(p, w) \in F \wedge \delta(q, w) \notin F)$$

Satz

Sind $\delta(p, a)$ und $\delta(q, a)$ unterscheidbar, dann auch p und q .



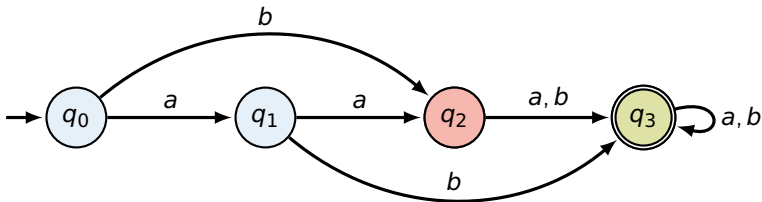
Definition (Unterscheidbarkeit)

Zwei Zustände sind **unterscheidbar**, wenn sie unterschiedliche Sprachen akzeptieren.

$$p \not\equiv_A q \iff (\exists w \in \Sigma^*. \delta(p, w) \in F \wedge \delta(q, w) \notin F)$$

Satz

Sind $\delta(p, a)$ und $\delta(q, a)$ unterscheidbar, dann auch p und q .



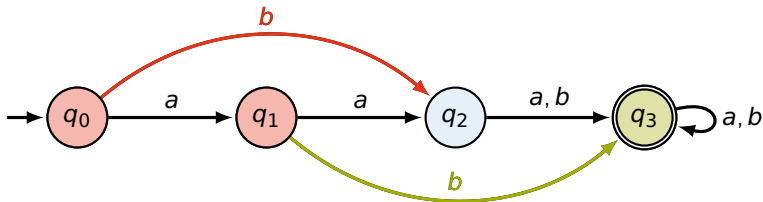
Definition (Unterscheidbarkeit)

Zwei Zustände sind **unterscheidbar**, wenn sie unterschiedliche Sprachen akzeptieren.

$$p \not\equiv_A q \iff (\exists w \in \Sigma^*. \delta(p, w) \in F \wedge \delta(q, w) \notin F)$$

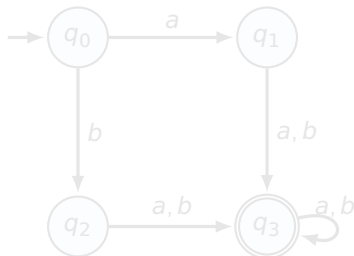
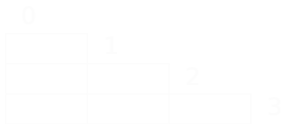
Satz

Sind $\delta(p, a)$ und $\delta(q, a)$ unterscheidbar, dann auch p und q .



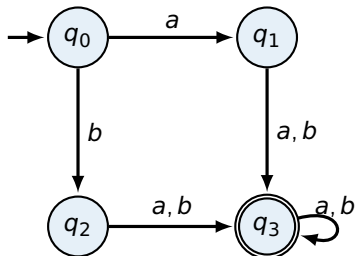
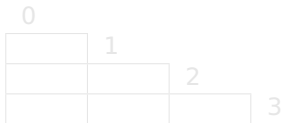
Quotientenautomat

- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände



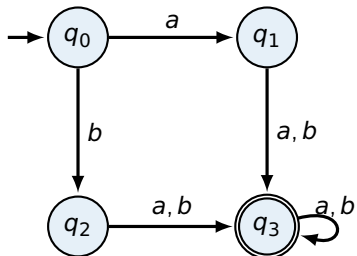
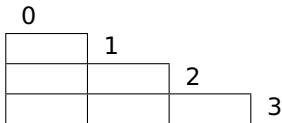
Quotientenautomat

- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände



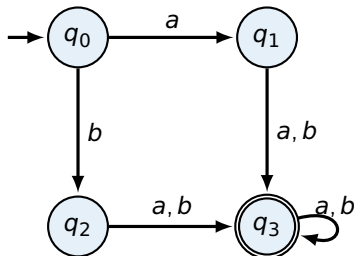
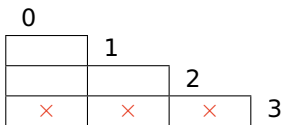
Quotientenautomat

- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände



Quotientenautomat

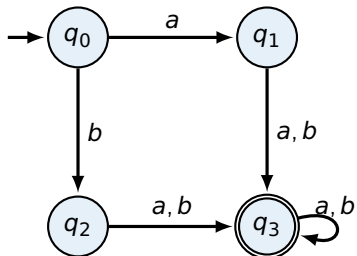
- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände



Quotientenautomat

- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände

0			
1/a	1		
1/a		2	
×	×	×	3



Quotientenautomat

- 1 Entferne alle von q_0 **nicht erreichbaren** Zustände
- 2 Berechne die **unterscheidbaren** Zustände
- 3 **Kollabiere** die äquivalenten Zustände

0			
1/a	1		
1/a		2	
×	×	×	3

