

# Algorithmen und Datenstrukturen

## Theoretische Grundlagen der Informatik

Prof. Dr. rer. nat. Jörg Striegnitz

Fachhochschule Aachen  
striegnitz@fh-aachen.de

29. April 2024

# Algorithmen und Datenstrukturen

## Nichtdeterministische endliche Automaten

Prof. Dr. rer. nat. Jörg Striegnitz

Fachhochschule Aachen  
striegnitz@fh-aachen.de

29. April 2024

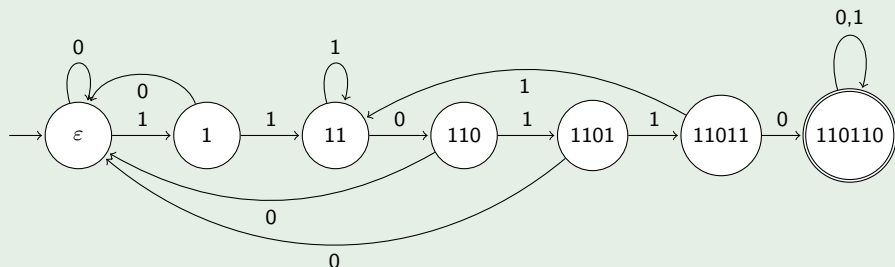
# Deterministischer endlicher Automat (Beispiel)

## Beispiel 1.40 (DEA - Deterministischer endlicher Automat)

Betrachten wir noch einmal den DEA, der die Sprache

$$L(M) = \{w \in \Sigma_{Bool}^* \mid w \text{ enthält } 110110\}$$

erkennt:

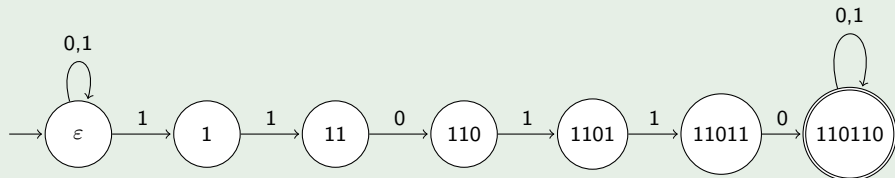


Insbesondere die Konstruktion der »**Rücksprünge**« war recht komplex (wir suchten das größte Suffix der gelesenen Eingabe, welches gleichzeitig Präfix des Suchmusters ist).

# Nichtdeterministische Automaten (Beispiel)

## Beispiel 1.41 (NEA - Nichtdeterministischer endlicher Automat)

Hier nun ein nichtdeterministischer Automat, der dieselbe Sprache erkennt:



### Beobachtungen:

- Im Vergleich zum DEA fehlen die »komplizierten« Rückschritte - die Konstruktion scheint daher leichter
- Von einem Zustand sind mehrere Zustände über einen Buchstaben erreichbar (z.B. Startzustand  $\varepsilon$ : mit 1 erreicht man die Zustände  $\varepsilon$  und 1)
- Es gibt nicht immer für jeden Buchstaben eine Transitionsmöglichkeit (z.B. ist unklar, wo man vom Zustand 11 mit der 1 hinkommt).

# Nichtdeterministische endliche Automaten

Abgesehen von **Transitionsfunktion** gleicht das statische Modell des NEA dem des DEA:

## Definition 1.44 (NEA)

Ein nichtdeterministischer endlicher Automat (NEA)  $M$  ist ein Quintupel  $(Q, \Sigma, \delta, q_0, F)$  aus

- einer endlichen Menge von **Zuständen**  $Q$ ,
- einem **Eingabealphabet**  $\Sigma$ ,
- einer **Transitionsfunktion**  $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$ ,
- einem **Startzustand**  $q_0 \in Q$  und
- einer Menge von **akzeptierenden Zuständen**  $F \subseteq Q$ .

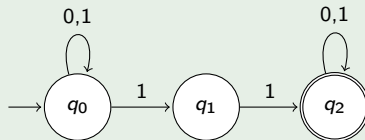
**Notation:** Falls  $q \in \delta(p, a)$  schreiben wir  $p \xrightarrow{a}_M q$  ( für  $p, q \in Q$  und  $a \in \Sigma$ ), oder falls  $M$  aus dem Kontext heraus bekannt:  $p \xrightarrow{a} q$ .

## Beachte:

- Die Transitionsfunktion liefert hier eine Menge von Zuständen als Ergebnis zurück und ist partiell!
- In der Literatur findet man anstelle der Transitionsfunktion manchmal auch eine dreistellige Relation  $\Delta \subseteq Q \times \Sigma \times Q$ .

## Beispiel 1.42

Betrachten wir folgenden Transitionsgraph zu einem NEA  $M$ :



**Formale Beschreibung:**  $M = (\{q_0, q_1, q_2\}, \Sigma_{Bool}, \delta, q_0, \{q_2\})$  mit

$$\delta(q_0, 0) = \{q_0\}$$

$$\delta(q_0, 1) = \{q_0, q_1\}$$

$$\delta(q_1, 0) = \emptyset$$

$$\delta(q_1, 1) = \{q_2\}$$

$$\delta(q_2, 0) = \{q_2\}$$

$$\delta(q_2, 1) = \{q_2\}$$

- Konfigurationen beschreibt man wie bei DEAs als Tupel aus  $Q \times \Sigma^*$
- Auch die Definition der Schrittrelation können wir direkt übernehmen:

## Definition 1.45 (Schrittrelation)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein NEA. Ein **Konfigurationsübergang** (ein Schritt) von  $M$  ist eine Relation  $\vdash_M: (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ , die definiert ist durch:

$$(p, aw) \vdash_M (q, w) \Leftrightarrow p \xrightarrow{a}_M q \Leftrightarrow q \in \delta(p, a)$$

- Im Vergleich zum DEA ist die Schrittrelation u.U. **nicht mehr eindeutig**, da  $\delta(p, a)$  eine Menge von Zuständen liefert.

Berechnungen müssen wir anpassen, da der Automat »abstürzen« kann:

## Definition 1.46 (Berechnung eines NEA)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein NEA. Eine Berechnung  $C$  von  $M$  ist eine endliche Folge von Konfigurationen

$$C = C_0, \dots, C_n \text{ mit } C_i \vdash C_{i+1} \quad (1 \leq i \leq n-1)$$

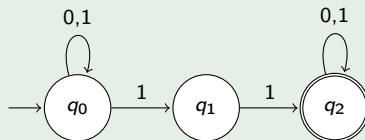
$C$  ist die Berechnung von  $M$  für eine Eingabe  $w$ , falls  $C_0 = (q_0, w)$  eine Startkonfiguration ist und entweder

- $C_n = (q, \varepsilon)$  eine Endkonfiguration ist, oder
- $C_n = (q, aw)$  und  $\delta(q, a) = \emptyset$  gilt (der Automat „stürzt ab“).

Ist  $C_n$  eine akzeptierende Endkonfiguration, so heißt  $C$  **akzeptierenden Berechnung** von  $M$  auf  $w$ . Ist  $C_n$  eine verwerfende Konfiguration, so heißt  $C$  eine **verwerfende Berechnung** von  $M$  auf  $w$  (wir sagen das  $M$  das Wort  $w$  verwirft).

## Beispiel 1.43

Betrachten wir folgenden Transitionsgraph zu einem NEA  $M$ :



**! Wichtig:** Beim NEA kann es für ein Wort mehrere Berechnungen geben!

• Hier z.B. drei mögliche Berechnungen für das Wort 10110:

1.  $(q_0, 10110) \vdash (q_0, 0110) \vdash (q_0, 110) \vdash (q_0, 10) \vdash (q_1, 0)$   
[Verwerfende Berechnung - Absturz]
2.  $(q_0, 10110) \vdash (q_0, 0110) \vdash (q_0, 110) \vdash (q_0, 10) \vdash (q_0, 0) \vdash (q_0, \varepsilon)$   
[Verwerfende Berechnung]
3.  $(q_0, 10110) \vdash (q_0, 0110) \vdash (q_0, 110) \vdash (q_1, 10) \vdash (q_2, 0) \vdash (q_2, \varepsilon)$   
[Akzeptierende Berechnung]

Definiert man genau wie beim DEA:

## Definition 1.47 (Reflexiver und transitiver Abschluss von $\vdash$ )

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein NEA. Wir definieren  $\vdash^* : (Q \times \Sigma^*) \times (Q \times \Sigma^*)$  als reflexiven und transitiven Abschluss der Schrittrelation  $\vdash$ :

- $(q, w) \vdash^* (q, w)$  für alle  $q \in Q$  und  $w \in \Sigma^*$  [*reflexiver Teil*]
- $(p, uv) \vdash^* (q, v)$ , falls  $\exists a_1, \dots, a_n \in \Sigma$  mit  $u = a_1 \dots a_n$  und  $\exists q_1, \dots, q_{n-1} \in Q$ , so dass  $(p, a_1 a_2 \dots a_n v) \vdash (q_1, a_2 \dots a_n v) \vdash \dots \vdash (q_{n-1}, a_n v) \vdash (q, v)$  (mit  $u, v \in \Sigma^*$ ,  $p, q \in Q$ ) [*transitiver Teil*].

Die Fortsetzung von  $\delta$  auf Wörter  $\hat{\delta}$ , muss dem neuen Typ von  $\delta$  angepasst werden:

## Definition 1.48 (Fortsetzung der Transitionsfunktion auf Wörter)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein NEA. Die Fortsetzung  $\hat{\delta} : Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$  der Transitionsfunktion  $\delta$  auf Wörter definieren wir induktiv als reflexiven und transitiven Abschluss von  $\delta$ :

- $\hat{\delta}(q, \varepsilon) = \{q\}$
- Sei  $w = va$  und  $\hat{\delta}(q, v) = \{q_1, \dots, q_n\}$  dann gilt

$$\hat{\delta}(q, va) = \bigcup_{i=1}^n \delta(q_i, a)$$

für  $v, w \in \Sigma^*$ ,  $a \in \Sigma$  und  $q \in Q$  und  $q_i \in Q$  ( $1 \leq i \leq n$ ).

**Notation:** Falls  $q \in \hat{\delta}(p, w)$ , so schreiben wir auch  $p \xrightarrow{w}_M q$ , oder (falls  $M$  aus Kontext bekannt):  $p \xrightarrow{w} q$ .

## Definition 1.49 (Vom NEA erkannte Sprache / äquivalente NEAs)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein NEA. Die von  $M$  akzeptierte Sprache  $L(M)$  ergibt sich aus

$$\begin{aligned} L(M) &= \left\{ w \in \Sigma^* \mid (q_0, w) \stackrel{*}{\vdash} (q_f, \varepsilon) \text{ mit } q_f \in F \right\} \\ &= \left\{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset \right\} \\ &= \left\{ w \in \Sigma^* \mid q_0 \stackrel{w}{\rightarrow} q_f \text{ mit } q_f \in F \right\} \end{aligned}$$

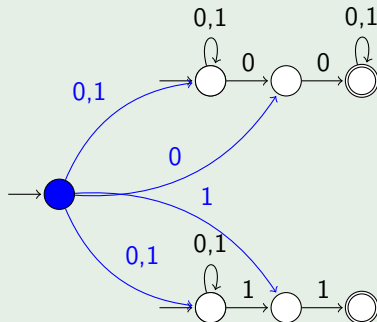
Zwei NEAs  $M_1$  und  $M_2$  heißen **äquivalent**, falls sie dieselbe Sprache erkennen; also:  $L(M_1) = L(M_2)$

- Betrachte  $L_3 = L_1 \cup L_2$ ; d.h.
  - wir können  $L_3$  zerlegen in die (vielleicht leichteren) Sprachen  $L_1$  und  $L_2$
  - ein Wort ist genau dann in  $L_3$  wenn es in  $L_1$  oder in  $L_2$  liegt.

## Beispiel 1.44 (Verknüpfung von Automaten)

Betrachte  $L_1 = \{w \in \Sigma_{Bool}^* \mid w \text{ enthält } 00\}$  und  $L_2 = \{w \in \Sigma_{Bool}^* \mid w \text{ endet auf } 11\}$ :

- wir können leicht einen NEA für  $L_3 = L_1 \cup L_2$  konstruieren:
- neuer Startzustand verhält sich wie Startzustand der Einzelautomaten
- Trick: Ausnutzung der eingebauten Suchfunktion des NEA

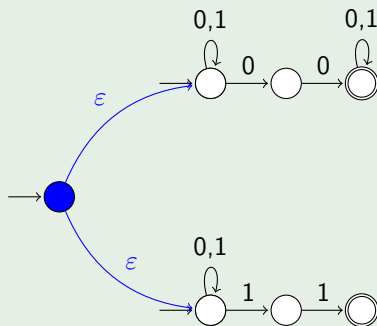


**Problem:** innere Struktur von  $M_1$  und  $M_2$  muss bekannt sein!

- **Lösung:** führe »stille« Transitionen ein, bei denen der Lesekopf nicht bewegt wird
- diese heißen  $\epsilon$ -Transitionen:

## Beispiel 1.45 (Verknüpfung von Automaten)

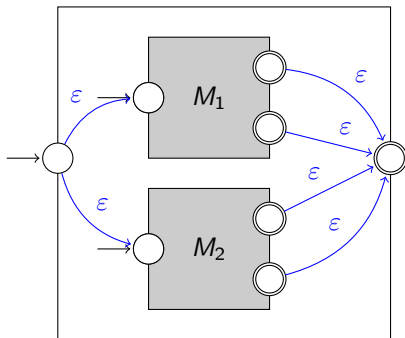
- Wir führen neuen **Startzustand** ein und
- verbinden diesen per  $\epsilon$ -Transitionen mit den Startzuständen von  $M_1$  und  $M_2$
- der resultierende NEA *sucht* jetzt in  $M_1$  und  $M_2$  nach akzeptierender Berechnung



Jetzt können wir Automaten wie »**Makrobausteine**« verwenden!

# Automaten als Makrobausteine

Jetzt können wir zwei beliebige Automaten  $M_1$  und  $M_2$  zu einem  $\varepsilon$ -NEA zusammenschalten, der  $L(M_1) \cup L(M_2)$  erkennt:

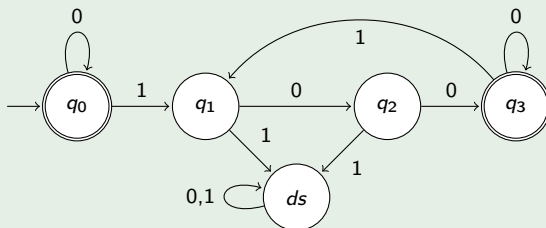


**Beachte:**  $M_1$  und  $M_2$  dürfen jeweils DEAs, NEAs, oder  $\varepsilon$ -NEAs sein!

## Beispiel 1.46

Folgender DEA erkennt die Sprache

$$L = \{w \in \Sigma_{Bool}^* \mid w = 0^* \vee w = 0^* (100^+)^+\}$$

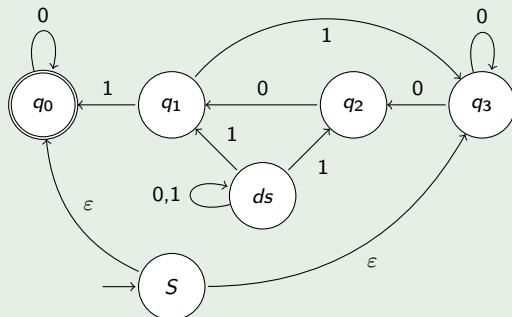


Gesucht ist ein endlicher Automat, der  $L^R$  erkennt.

## Beispiel 1.48

Wir erhalten einen  $\varepsilon$ -NEA, der  $L^R$  erkennt durch

- umkehren aller Transitionen
- alter Startzustand  $q_0$  wird einziger akzeptierender Zustand
- Neuer Startzustand  $S$ , der via  $\varepsilon$ -Transitionen mit alten akzeptierenden Zuständen ( $q_0$  und  $q_3$ ) verbunden wird:



Die formale Definition entspricht der des NEA, allerdings müssen wir die  $\varepsilon$ -Transitionen ergänzen:

## Definition 1.50 ( $\varepsilon$ -NEA)

Ein **nichtdeterministischer endlicher Automat mit  $\varepsilon$ -Transitionen** ( $\varepsilon$ -NEA)  $M$  ist ein Quintupel  $(Q, \Sigma, \delta, q_0, F)$  aus

- einer endlichen Menge von **Zuständen**  $Q$ ,
- einem **Eingabealphabet**  $\Sigma^a$ ,
- einer **Transitionsfunktion**  $\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow \mathcal{P}(Q)$ ,
- einem **Startzustand**  $q_0 \in Q$  und
- einer Menge von **akzeptierenden Zuständen**  $F \subseteq Q$ .

**Notation:** Falls  $q \in \delta(p, a)$  schreiben wir  $p \xrightarrow{a}_M q$  ( für  $p, q \in Q$  und  $a \in \Sigma \cup \{\varepsilon\}$  ).

---

<sup>a</sup>Zur Erinnerung: wir hatten vereinbart, dass  $\varepsilon$  - als Symbol - in keinem Alphabet auftreten darf.

- Konfigurationen definieren wir analog zu DEA und NEA als Paar aus  $Q \times \Sigma^*$ .
- Schrittrelation  $\vdash$  müssen wir anpassen, denn Übergänge der Art

$$(p, w) \vdash_M (q, w)$$

sind jetzt möglich - **Beachte:** Lesekopf wird nicht bewegt!

- Mehrere  $\varepsilon$ -Transitionen können in einem Schritt durchlaufen werden!  
Wir müssen für jeden Zustand die über  $\varepsilon$ -Transitionen erreichbaren Zustände kennen

## Definition 1.51 ( $\varepsilon$ -Hülle)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein  $\varepsilon$ -NEA. Die  $\varepsilon$ -Hülle  $\varepsilon_C : Q \rightarrow \mathcal{P}(Q)$  ist definiert durch:

$$\varepsilon_C(q) = \{q\} \cup \bigcup_{q' \in \delta(q, \varepsilon)} \varepsilon_C(q')$$

**Beachte:** definitionsgemäß ist jeder Knoten  $q$  in seiner eigenen  $\varepsilon$ -Hülle  $\varepsilon_C(q)$  enthalten.

## Definition 1.52 (Schrittrelation für $\varepsilon$ -NEA)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein  $\varepsilon$ -NEA. Ein **Konfigurationsübergang** (ein Schritt) von  $M$  ist eine Relation  $\vdash_M: (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ , die definiert ist durch:

$$(q_1, aw) \vdash_M (q_2, w) \Leftrightarrow q_2 \in \delta(q_1, a)$$

und

$$(q_1, w) \vdash_M (q_2, w) \Leftrightarrow q_2 \in \varepsilon_C(q_1)$$

wobei  $a \in \Sigma$ ,  $w \in \Sigma^*$  und  $\{q_1, q_2\} \subseteq Q$ .  $\vdash_M$  nennen wir **Schrittrelation**.

Ein Schritt bedeutet also

- das »Konsumieren« eines Zeichens oder
- das Bewegen entlang beliebig vieler  $\varepsilon$ -Transitionen

Definiert man genau wie beim NEA:

## Definition 1.53 (Reflexiver und transitiver Abschluss von $\vdash$ )

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein  $\varepsilon$ -NEA. Wir definieren  $\vdash^*: (Q \times \Sigma^*) \times (Q \times \Sigma^*)$  als reflexiven und transitiven Abschluss der Schrittrelation  $\vdash$ :

- $(q, w) \vdash^* (q, w)$  für alle  $q \in Q$  und  $w \in \Sigma^*$  [*reflexiver Teil*]
- $(p, uv) \vdash^* (q, v)$ , falls  $\exists a_1, \dots, a_n \in \Sigma$  mit  $u = a_1 \dots a_n$  und  $\exists q_1, \dots, q_{n-1} \in Q$ , so dass  $(p, a_1 a_2 \dots a_n v) \vdash (q_1, a_2 \dots a_n v) \vdash \dots \vdash (q_{n-1}, a_n v) \vdash (q, v)$  (mit  $u, v \in \Sigma^*$ ,  $p, q \in Q$ ) [*transitiver Teil*].

Analog zum NEA:

## Definition 1.54 (Fortsetzung der Transitionsfunktion auf Wörter)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein  $\varepsilon$ .NEA. Die Fortsetzung  $\hat{\delta} : Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$  der Transitionsfunktion  $\delta$  auf Wörter definieren wir induktiv als reflexiven und transitiven Abschluss von  $\delta$ :

- $\hat{\delta}(q, \varepsilon) = \{q\}$
- Sei  $w = va$  und  $\hat{\delta}(q, v) = \{q_1, \dots, q_n\}$  dann gilt

$$\hat{\delta}(q, va) = \bigcup_{i=1}^n \delta(q_i, a)$$

für  $v, w \in \Sigma^*$ ,  $a \in \Sigma$  und  $q \in Q$  und  $q_i \in Q$  ( $1 \leq i \leq n$ ).

**Notation:** Falls  $q \in \hat{\delta}(p, w)$ , so schreiben wir auch  $p \xrightarrow{w}_M q$ , oder (falls  $M$  aus Kontext bekannt):  $p \xrightarrow{w} q$ .

Analog zum NEA:

**Definition 1.55** (Vom  $\varepsilon$ -NEA erkannte Sprache / äquivalente NEAs)

Sei  $M = (Q, \Sigma, \delta, q_0, F)$  ein  $\varepsilon$ -NEA. Die von  $M$  akzeptierte Sprache  $L(M)$  ergibt sich aus

$$\begin{aligned} L(M) &= \left\{ w \in \Sigma^* \mid (q_0, w) \stackrel{*}{\vdash} (q_f, \varepsilon) \text{ mit } q_f \in F \right\} \\ &= \left\{ w \in \Sigma^* \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset \right\} \\ &= \left\{ w \in \Sigma^* \mid q_0 \xrightarrow{w} q_f \text{ mit } q_f \in F \right\} \end{aligned}$$

Zwei  $\varepsilon$ -NEAs  $M_1$  und  $M_2$  heißen **äquivalent**, falls sie dieselbe Sprache erkennen; also:  $L(M_1) = L(M_2)$

## • Pro:

- Automaten sind Algorithmen (Programme) zur Lösung von Entscheidungsproblemen
- NEA- und  $\varepsilon$ -NEA-Modell erleichtern die Konstruktion von Algorithmen also die »Programmierung«
- insbesondere  $\varepsilon$ -NEAs sind ein gutes Hilfsmittel zur funktionalen Dekomposition Zerlegung eines Problems in kleinere, leichter zu lösende Teilprobleme

## • Contra:

- Zustände implizieren keine Äquivalenzklassen mehr  
Mehr Kreativität bei Beweisen gefordert?
- DEA: Zahl der effektiven Rechenschritte verhält sich 1:1 zur Größe der Eingabe (der Wortlänge).
- **Aber:** Wollten wir die nichtdeterministischen Modelle mit einem Computerprogramm (deterministisch) simulieren, müssten wir deren eingebaute Suchfunktion nachbilden. Die Zahl der effektiven Rechenschritte steigt dabei im ungünstigsten Fall exponentiell mit der Größe der Eingabe!  
Wir erkaufen »Eleganz in der Formulierung« mit Laufzeit!