Automatentheorie und Formale Sprachen

Teil 5 – Epsilon-EA, verallgemeinerte EA und Automatenminimierung

Prof. Dr. Hans-Werner Sehring

NORDAKADEMIE
HOCHSCHULE DER WIRTSCHAFT

Abschnitt 1

Endliche Automaten mit ε-Übergängen

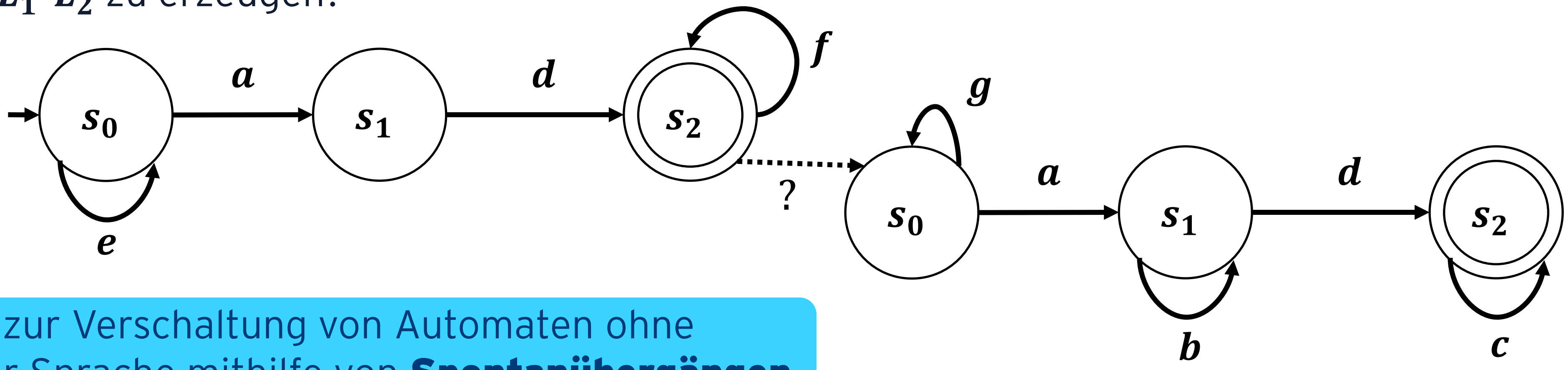
Motivation

Mit DEA können wir die Klasse der regulären Automaten beschreiben.

Mit NEA können wir z.B. den Beweis der Abgeschlossenheit regulärer Sprachen unter Mengenvereinigung ($\boldsymbol{L_1}$ und $\boldsymbol{L_2}$ regulär $\Rightarrow$ $\boldsymbol{L_1} \cup \boldsymbol{L_2}$ regulär) leichter führen.

Wie ist es nun mit der Abgeschlossenheit unter Konkatenation? Sprachen $\boldsymbol{L_1}$ und $\boldsymbol{L_2}$ regulär $\Rightarrow$ $\boldsymbol{L_1} \circ \boldsymbol{L_2}$ regulär.

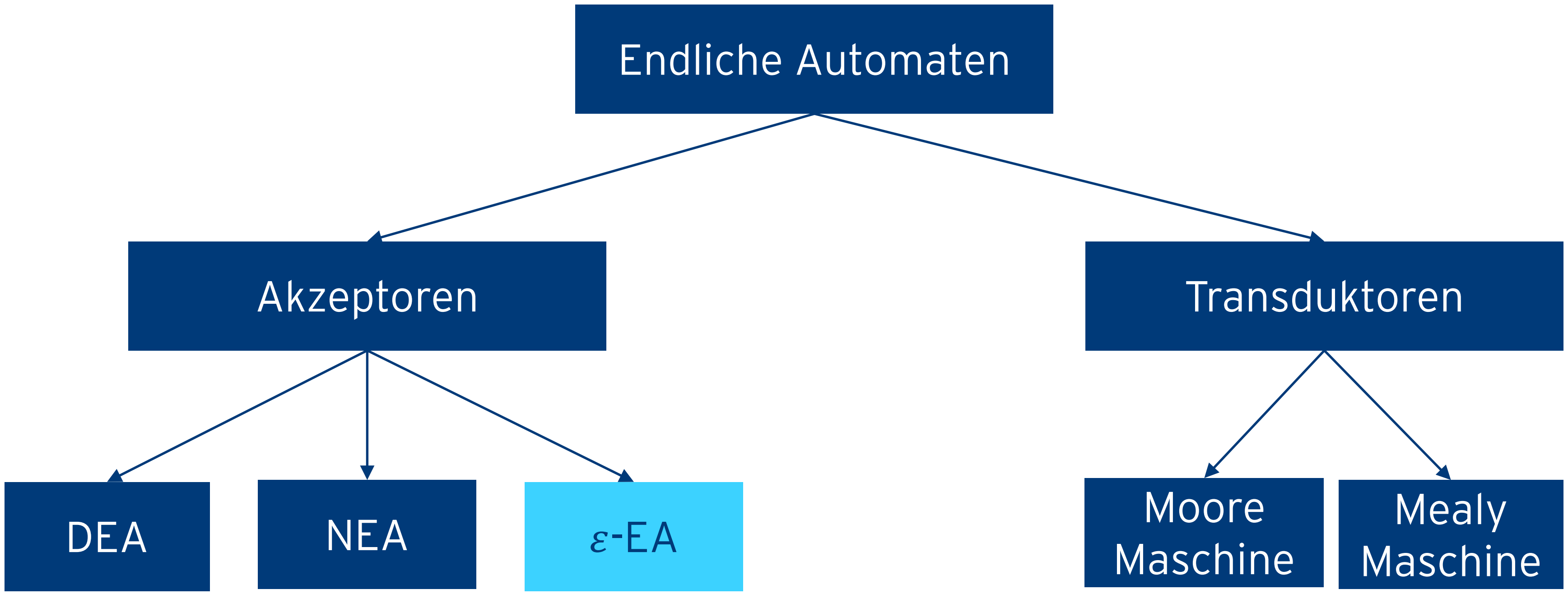
Gegeben seien zwei Automaten $\boldsymbol{A_1}$ und $\boldsymbol{A_2}$, die die Sprachen $\boldsymbol{L_1}$ und $\boldsymbol{L_2}$ als akzeptierendes Konzept beschreiben. Auf welche Weise lassen sich diese Automaten miteinander verschalten, um daraus bspw. die Sprache $\boldsymbol{L_3} = \boldsymbol{L_1} \circ \boldsymbol{L_2}$ zu erzeugen?



Konzepte zur Verschaltung von Automaten ohne Modifikation der Sprache mithilfe von **Spontanübergängen**

EA mit ε-Übergängen

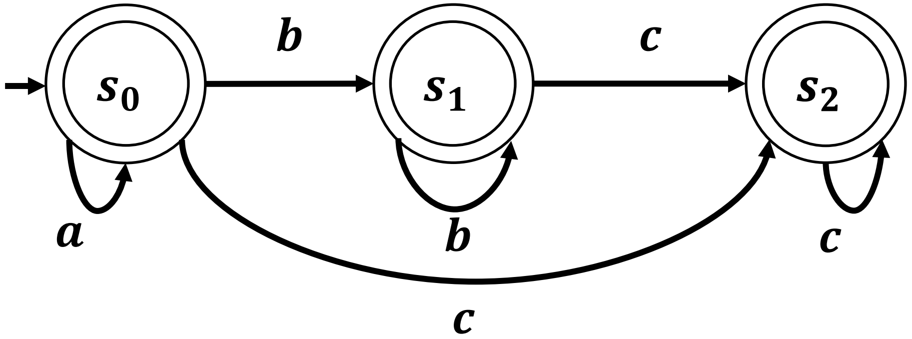
Folgende Automatentypen werden im Rahmen der Vorlesung behandelt:



Endliche Automaten
mit ε-Übergängen

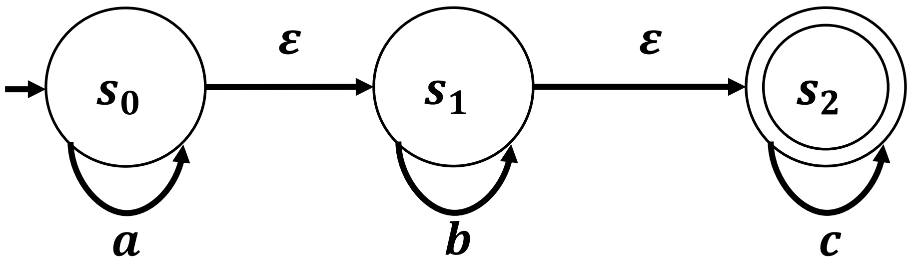
Spontanübergänge

Gesucht sei der Automat, der die Sprache $L_{abc} = \{w \in \Sigma^* \mid a^i b^j c^k; i, j, k \geq 0\}$ akzeptiert.



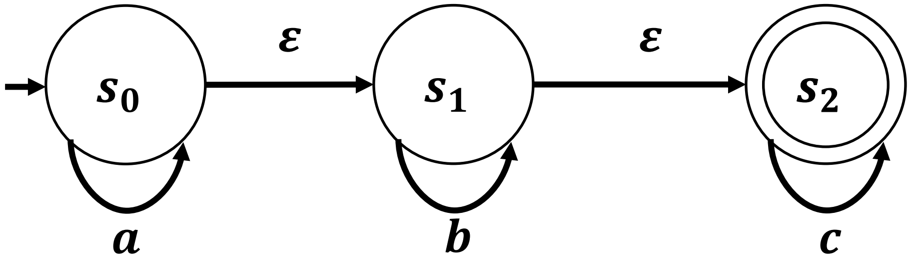
Mit der Erweiterung auf das leere Wort lässt sich der Automat kompakter schreiben als

Bisher definiert:
$\forall s \in S\colon\ \delta^*(s, \varepsilon) = s$



Wird kein Zeichen gelesen, ist ein Spontanübergang möglich.

EA mit ε-Übergängen (1)



Ein **ε-EA** ist definiert durch

$$A_\varepsilon = (\Sigma \cup \{\varepsilon\}, S, \delta_\varepsilon, S_0, F)$$

mit

$$\delta_\varepsilon \subseteq S \times (\Sigma \cup \{\varepsilon\}) \times S$$

$$(s, aw) \vdash (s', w) \text{ gdw. } (s, a, s') \in \delta_\varepsilon, a \in \Sigma \cup \{\varepsilon\}, w \in \Sigma^*$$

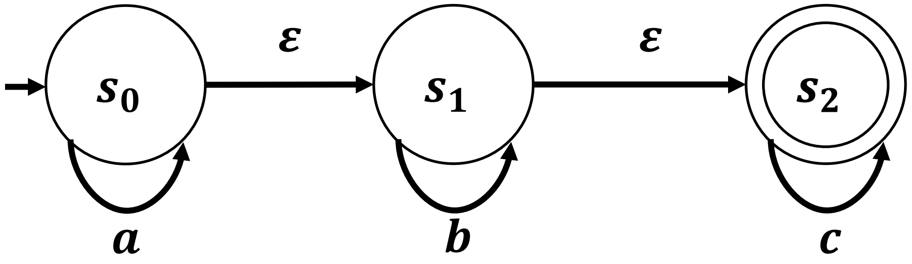
Das heißt, das Alphabet wird um das leere Wort ε erweitert.

Die Zustandsüberführungsfunktion lässt sich über

$$\delta'_\varepsilon: P(S) \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(S)$$

in allgemeiner Form beschreiben.

EA mit ε-Übergängen (2)



Analog zum NEA gilt für die Sprache $\boldsymbol{L}$:

$$L(A) = \{w \in \Sigma^* \mid (s_0, w) \vdash^* (s, \varepsilon), s_0 \in S_0, s \in F\}$$

Mit

$$\varepsilon FA_\Sigma = \bigcup_{L \subseteq \Sigma^*,\ \exists A \text{ ein } \varepsilon-EA: L(A)=L} L$$

wird die Klasse der Sprachen notiert, die von $\boldsymbol{\varepsilon}$-EA akzeptiert werden.

EA mit $\boldsymbol{\varepsilon}$-Übergängen werden z.B. für die modulare Verschaltung von Automaten verwendet.

Äquivalenz EA mit ε-Übergängen und NEA (1)

Es gilt: $\boldsymbol{NFA_\Sigma = \varepsilon FA_\Sigma}$, d.h., NEA und $\boldsymbol{\varepsilon}$-EA sind **gleichmächtig**.

1. Jeder NEA ist auch ein spezieller $\boldsymbol{\varepsilon}$-EA, $\boldsymbol{NFA_\Sigma \subseteq \varepsilon FA_\Sigma}$, der allerdings keine Epsilon-Übergänge enthält. Die von NEA akzeptierten Sprachen werden auch von Epsilon-EA akzeptiert.

2. Ebenso lässt sich jeder $\boldsymbol{\varepsilon}$-EA in einen NEA überführen, d.h., es gilt $\boldsymbol{NFA_\Sigma \supseteq \varepsilon FA_\Sigma}$.

Die Inklusion $\boldsymbol{NFA_\Sigma \subseteq \varepsilon FA_\Sigma}$ nehmen wir als offensichtlich hin.

Die Inklusion $\boldsymbol{NFA_\Sigma \supseteq \varepsilon FA_\Sigma}$ zeigen wir durch eine mehrschrittige Transformation.

Sei dazu $\boldsymbol{A = (\Sigma \cup \{\varepsilon\}, S, \delta, S_0, \mathrm{F})}$ ein $\boldsymbol{\varepsilon}$-EA.

Äquivalenz EA mit ε-Übergängen und NEA (2)

Transformationsschritt 1

Einfügen genau eines Anfangszustands $\boldsymbol{s_0}$ und eines Endzustands $\boldsymbol{f}$, d.h., $\boldsymbol{F}$ wird durch $\{\boldsymbol{f}\}$ ersetzt. Damit ergibt sich folgende Darstellung für den neuen EA:

$$\boldsymbol{A}' = (\boldsymbol{\Sigma} \cup \{\boldsymbol{\varepsilon}\}, \boldsymbol{S} \cup \{\boldsymbol{s_0}, \boldsymbol{f}\}, \boldsymbol{\delta}', \{\boldsymbol{s_0}\}, \{\boldsymbol{f}\}), \boldsymbol{s_0}, \boldsymbol{f} \notin \boldsymbol{S}$$

mit

$$\forall \boldsymbol{s} \in \boldsymbol{S_0}\colon\ \boldsymbol{\delta}'(\boldsymbol{s_0}, \boldsymbol{\varepsilon}) = \boldsymbol{s}$$

und

$$\forall \boldsymbol{s} \in \boldsymbol{F}\colon\ \boldsymbol{\delta}'(\boldsymbol{s}, \boldsymbol{\varepsilon}) = \boldsymbol{f}$$

Das heiβt,

- vom neuen Startzustand $\boldsymbol{s_0}$ werden zusätzliche Übergänge zu den Startzuständen des Epsilon-EA und
- von allen akzeptierenden Zuständen des Epsilon-EA werden Übergänge zum neuen finalen Zustand $\boldsymbol{f}$

hinzugefügt.

Äquivalenz EA mit ε-Übergängen und NEA (3)

Beispiel: Einfügen neuer Endzustände und eines neuen Startzustands

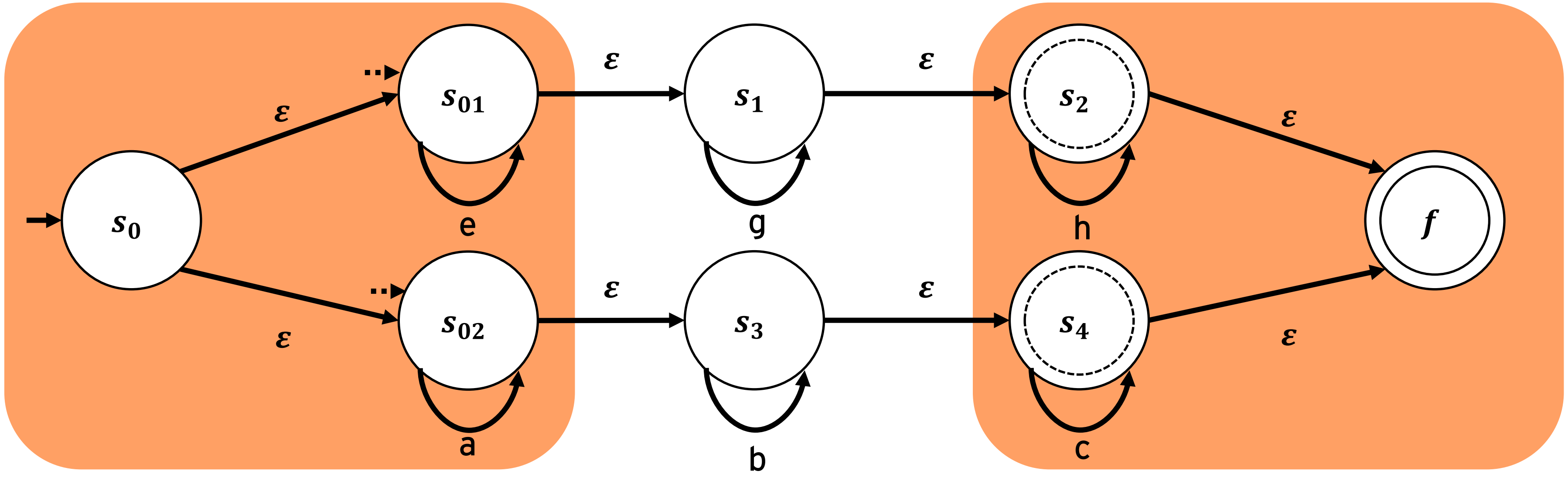
$$A' = (\Sigma \cup \{\varepsilon\}, S \cup \{s_0\}, \delta', \{s_0\}, \{f\})$$

mit

$$\delta'(s_0, \varepsilon) = s \in S_0$$

und

$$\forall s \in F\colon\ \delta'(s, \varepsilon) = f$$



Äquivalenz EA mit ε-Übergängen und NEA (4)

Transformationsschritt 2

Elimination von $\boldsymbol{\varepsilon}$-Zyklen:

Alle aufeinanderfolgenden $\boldsymbol{\varepsilon}$-Übergänge, die am Ende der Sequenz auf sich selbst $(\boldsymbol{s}, \boldsymbol{\varepsilon})_{\boldsymbol{i}}$ führen, werden in einem einzelnen $\boldsymbol{\varepsilon}$-Übergang zusammengefasst.

$$(\boldsymbol{s_{start}}, \boldsymbol{\varepsilon})_{\boldsymbol{j}} \vdash (s_{i_1}\boldsymbol{s^{i=1}}, \boldsymbol{\varepsilon})_{\boldsymbol{j+1}} \vdash \cdots \vdash \left(\boldsymbol{s_{start}}^{\boldsymbol{i=k}}, \boldsymbol{\varepsilon}\right)_{\boldsymbol{j+k}} \Rightarrow (\boldsymbol{s_\varepsilon}, \boldsymbol{\varepsilon});\;\; \boldsymbol{s}, \boldsymbol{s_{start}} \in \boldsymbol{S}, \boldsymbol{s_\varepsilon} \notin \boldsymbol{S}$$

wobei alle Übergänge, die auf alle $\boldsymbol{s^i}$ gerichtet sind, nun auf $\boldsymbol{s_\varepsilon}$ gerichtet werden und alle Übergänge, die von $\boldsymbol{s^i}$ ausgingen, nun von $\boldsymbol{s_\varepsilon}$ ausgehen.

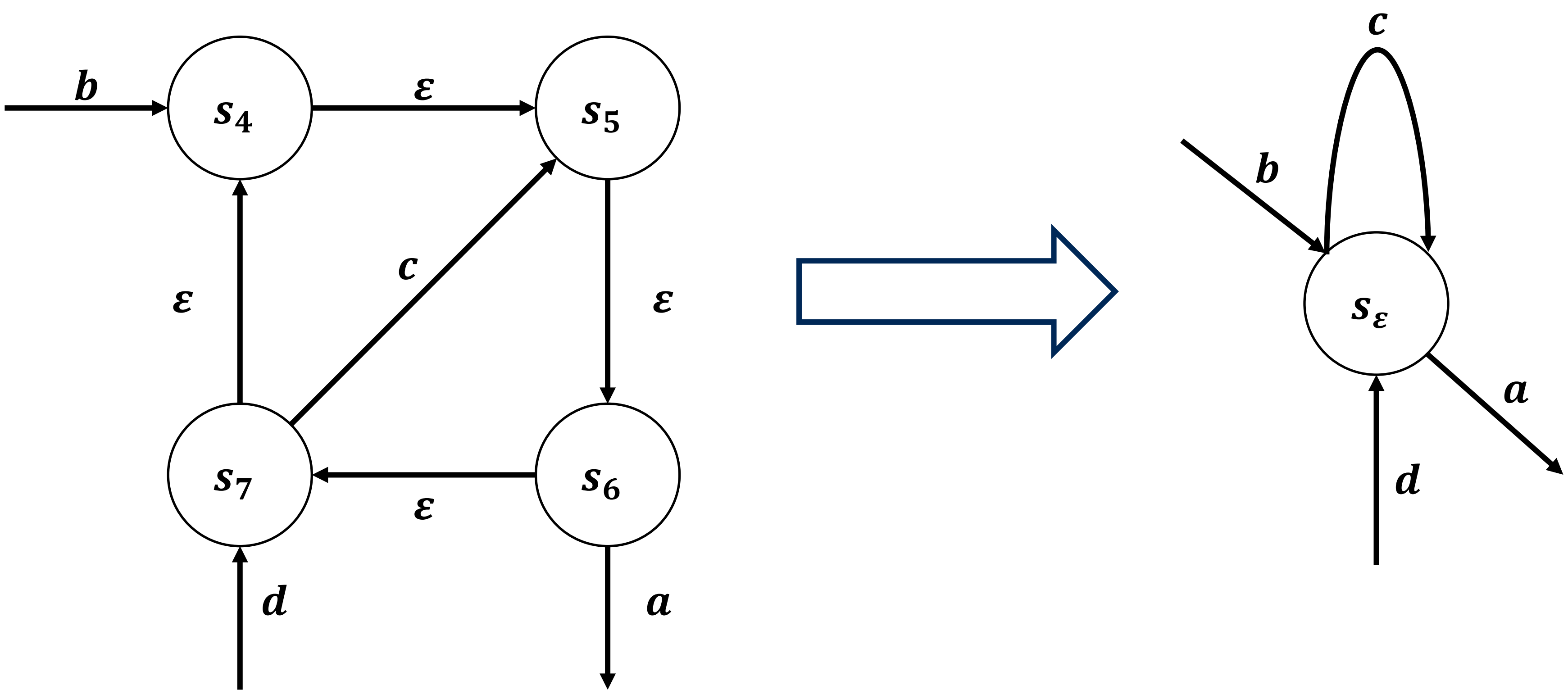
Ferner gilt:

$$\boldsymbol{s^i} \notin \boldsymbol{S'}; \boldsymbol{S'} = \{\boldsymbol{s_\varepsilon}\} \cup \boldsymbol{S} \cup \{\boldsymbol{s_0}\} \cup \{\boldsymbol{f}\}$$

d.h., die Menge übrig gebliebener Zustände wird mit den neuen Zuständen vereinigt

Äquivalenz EA mit ε-Übergängen und NEA (5)

Beispiel:

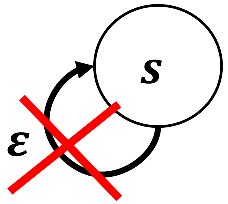


Äquivalenz EA mit ε-Übergängen und NEA (6)

Transformationsschritt 3

Ferner werden $\boldsymbol{\varepsilon}$-Übergänge entfernt, die auf sich selbst führen

$$\delta(\{s\}, \varepsilon) = \{s\} \Longrightarrow (s, \varepsilon, s) \notin \delta'$$

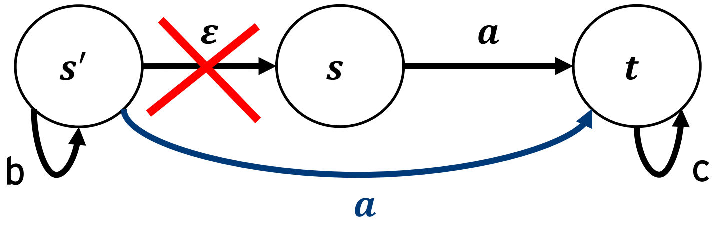


Äquivalenz EA mit ε-Übergängen und NEA (7)

Transformationsschritt 4

Der nächste Schritt besteht im Austausch der noch enthaltenen ε-Übergänge durch echte Übergänge.

$$\boldsymbol{\delta^*(s', \varepsilon) = s \wedge \delta(s, a) = t \implies (s', a, t) \in \delta'}$$



Äquivalenz EA mit ε-Übergängen und NEA (8)

Transformationsschritt 5

Der abschließende Schritt besteht in der Bestimmung der Endzustandsmenge.

$$\boldsymbol{F} = \{\boldsymbol{t} \in \boldsymbol{S}' | (\boldsymbol{t}, \boldsymbol{\varepsilon}) \vdash^* (\boldsymbol{f}, \boldsymbol{\varepsilon})\}$$

Das heißt, alle $\boldsymbol{\varepsilon}$ -Übergänge von $\boldsymbol{t}$ nach $\boldsymbol{f}$ werden beseitigt und das jeweilige $\boldsymbol{t}$ wird zum Endzustand.

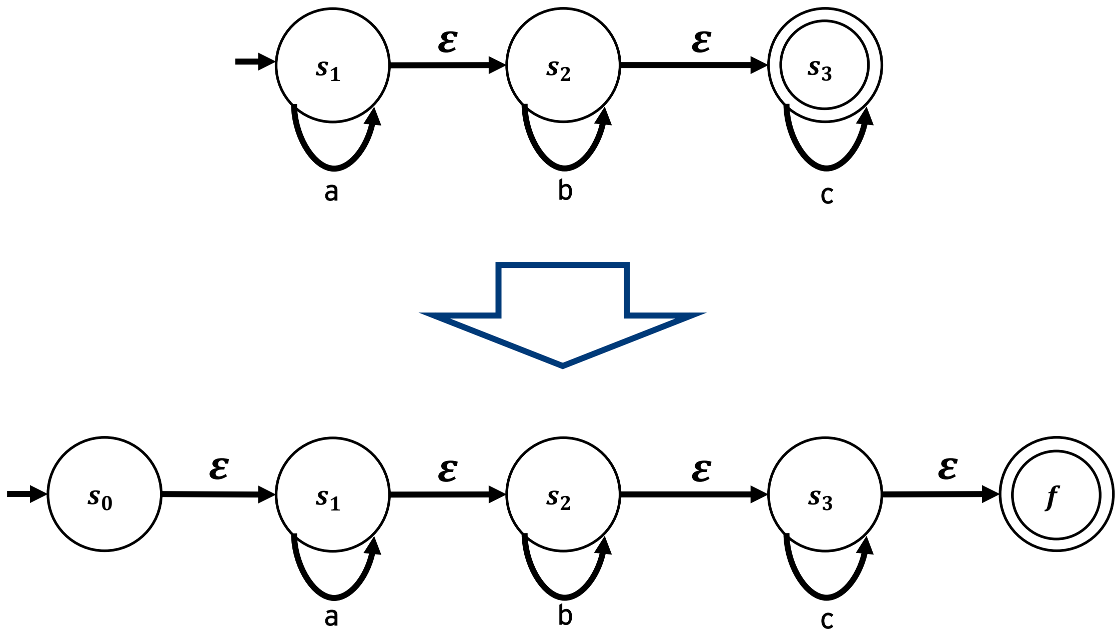
Äquivalenz EA mit ε-Übergängen und NEA (9)

Beispiel:

$$A(L_{abc}), L_{abc} = \{w \in \Sigma^* | a^i b^j c^k; i, j, k \geq 0\}$$

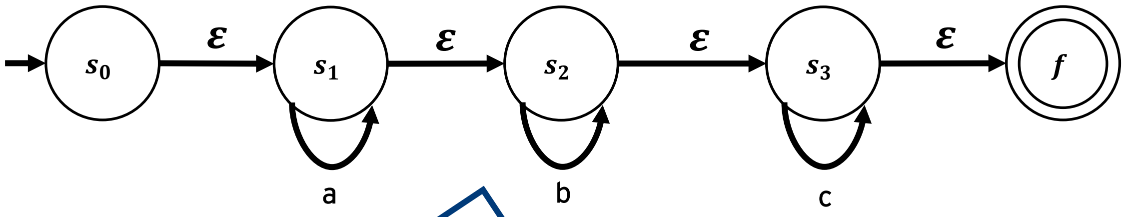
Einfügen neuer End- und Anfangszustände:

$$\delta' := \delta \cup (s_0, \varepsilon, s_1) \cup (s_3, \varepsilon, f)$$



Äquivalenz EA mit ε-Übergängen und NEA (10)

Eliminierung der ε-Übergänge und abschließende Bestimmung der Endzustandsmenge:



$$\left.\begin{aligned}\delta(s_0,\varepsilon)&=s_1\\ \delta(s_1,a)&=s_1\end{aligned}\right\}\rightarrow \delta'(s_0,a)=s_1\rightarrow \delta\cup\{(s_0,a,s_1)\}$$

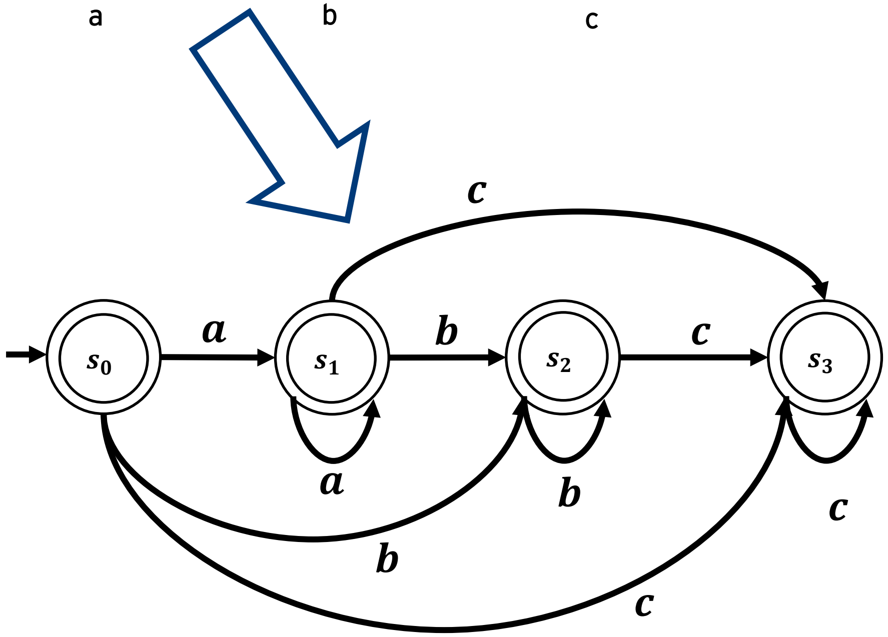
$$\left.\begin{aligned}\delta(s_1,\varepsilon)&=s_2\\ \delta(s_2,b)&=s_2\end{aligned}\right\}\rightarrow \delta'(s_1,b)=s_2\rightarrow \delta\cup\{(s_1,b,s_2)\}$$

$$\left.\begin{aligned}\delta(s_2,\varepsilon)&=s_3\\ \delta(s_3,c)&=s_3\end{aligned}\right\}\rightarrow \delta'(s_2,c)=s_3\rightarrow \delta\cup\{(s_2,c,s_3)\}$$

$$\left.\begin{aligned}\delta^*(s_0,\varepsilon)&=s_2\\ \delta(s_2,b)&=s_2\end{aligned}\right\}\rightarrow \delta'(s_0,b)=s_2\rightarrow \delta\cup\{(s_0,b,s_2)\}$$

$$\left.\begin{aligned}\delta^*(s_0,\varepsilon)&=s_3\\ \delta(s_3,c)&=s_3\end{aligned}\right\}\rightarrow \delta'(s_0,c)=s_3\rightarrow \delta\cup\{(s_0,c,s_3)\}$$

$$\left.\begin{aligned}\delta^*(s_1,\varepsilon)&=s_3\\ \delta(s_3,c)&=s_3\end{aligned}\right\}\rightarrow \delta'(s_1,c)=s_3\rightarrow \delta\cup\{(s_1,c,s_3)\}$$



$$\delta\backslash\{(s_0,\varepsilon,s_1),(s_1,\varepsilon,s_2),(s_2,\varepsilon,s_3)\}$$

Abschnitt 2

Verallgemeinerte Endliche Automaten

Verallgemeinerte Automaten (1)

Definition **verallgemeinerte endliche Automaten**

Verallgemeinerte endliche Automaten eignen sich zur kompakten Darstellung, da sie für Transitionen ganze Wörter als Eingabe akzeptieren.

$$A_G = (\Sigma, S, \delta_*, s_0, F)$$

mit

$$\delta_* \subseteq S \times \Sigma^* \times S$$
$$s_0 \in S, F \subseteq S$$
$$|\delta_*| \neq \infty$$

Dabei ergibt sich für die transitive Hülle der Konfiguration $\vdash^*$

$$(s, vw) \vdash^* (s', w) \text{ gdw. } \delta_*(s, v) = s', v \in \Sigma^+, w \in \Sigma^*$$

Analog dazu

$$L(A_G) = \{w \in \Sigma^* | (s_0, w) \vdash^* (s, \varepsilon), s \in F\}$$

Verallgemeinerte Automaten (2)

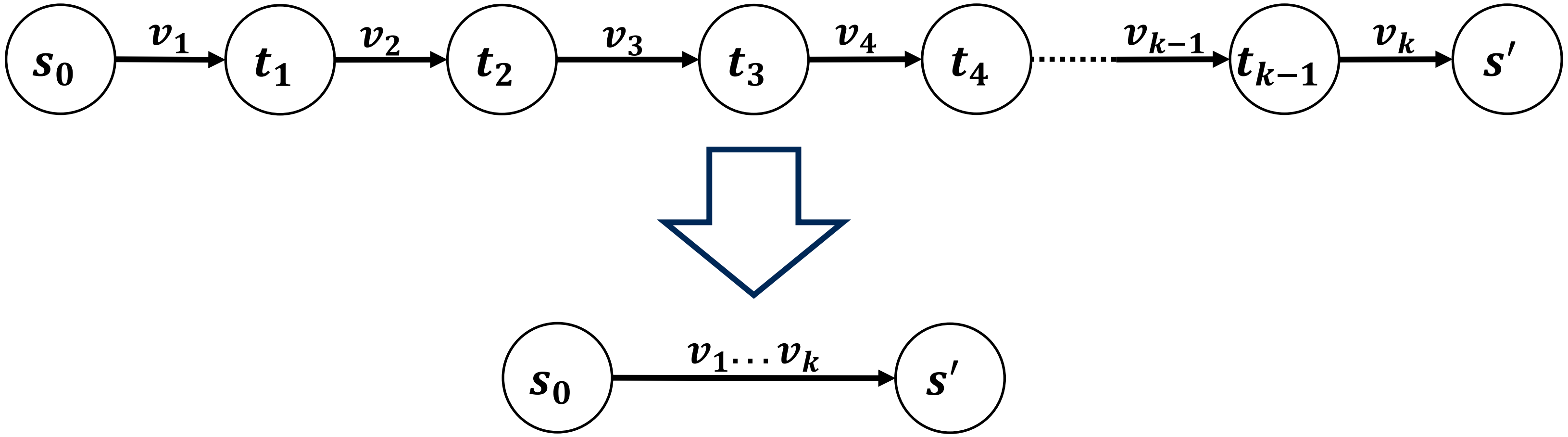
Die Klasse aller Sprachen über einen verallgemeinerten endlichen Automaten wird

$$GFA_{\Sigma} = \{L(A)\}$$

genannt.

Satz: Zu jedem verallgemeinerten endlichen Automaten existiert ein äquivalenter endlicher Automat A_G.

Umformungsprinzip:
Zustandsübergänge können zusammengefasst werden.



Abschnitt 3

Automaten-minimierung

Satz von Myhill und Nerode (1)

Gegeben seien $\boldsymbol{\Sigma}$ und $\boldsymbol{L} \subseteq \boldsymbol{\Sigma}^*$. Die Relation

$$\boldsymbol{R_L} \subseteq \boldsymbol{L} \times \boldsymbol{L}$$

sei wie folgt definiert: für $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{L}$

$$\boldsymbol{xR_Ly} \text{ gdw. } \forall \boldsymbol{z} \in \boldsymbol{\Sigma}^*\colon\ \boldsymbol{xz}, \boldsymbol{yz} \in \boldsymbol{L}$$

Diese Relation ist

- reflexiv $\quad \forall \boldsymbol{x} \in \boldsymbol{L}: \boldsymbol{xR_Lx}$
- transitiv $\quad \forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w} \in \boldsymbol{L}: \boldsymbol{xR_Ly} \wedge \boldsymbol{yR_Lw} \Rightarrow \boldsymbol{xR_Lw}$
- symmetrisch $\quad \forall \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{L}: \boldsymbol{xR_Ly} \Leftrightarrow \boldsymbol{yR_Lx}$

⇒ Die Relation $\boldsymbol{R_L}$ ist eine **Äquivalenzrelation**

Satz von Myhill und Nerode (2)

Gegeben sei der DEA $\boldsymbol{A} = (\boldsymbol{\Sigma}, \boldsymbol{S}, \boldsymbol{\delta}, \boldsymbol{s_0}, \boldsymbol{F})$. Die Relation $\boldsymbol{R_A}$ ist dann wie folgt definiert:

$$\boldsymbol{xR_Ay} \text{ gdw. } \boldsymbol{\delta^*(s_0, x) = \delta^*(s_0, y)}$$

Das heiβt, der Automat erreicht beim Abarbeiten der Wörter $\boldsymbol{x}$ und $\boldsymbol{y}$ den gleichen Zustand. Diese Relation ist ebenfalls eine Äquivalenzrelation. Diese Relation hat zudem die definierende Eigenschaft von $\boldsymbol{R_L}$, denn es gilt für $\boldsymbol{x, y \in L, xR_Ay}$:

$$\boldsymbol{\delta^*(s_0, xz) = \delta^*(\delta^*(s_0, x), z) = \delta^*(\delta^*(s_0, y), z) = \delta^*(s_0, yz)}$$

Bei den Relationen $\boldsymbol{R_A}$ und $\boldsymbol{R_L}$ handelt es sich um **Rechtskongruenzen**, d.h., die Äquivalenzrelationen sind verträglich mit der Konkatenation von Wörtern von rechts.

Satz von Myhill und Nerode (3)

Für eine Äquivalenzrelation $\boldsymbol{R} \subseteq \boldsymbol{\Sigma}^* \times \boldsymbol{\Sigma}^*$ nennen wir für ein $\boldsymbol{a} \in \boldsymbol{\Sigma}^*$ die Mengen

$$[\boldsymbol{a}]_{\boldsymbol{R}} := \{\boldsymbol{b} \in \boldsymbol{A} \mid \boldsymbol{a}\, \boldsymbol{R}\, \boldsymbol{b}\}$$

Äquivalenzklassen von $\boldsymbol{R}$. Für die Teilmenge $[\boldsymbol{a}]_{\boldsymbol{R}}$ heißt $\boldsymbol{a}$ **Repräsentant**.

Die Menge aller Äquivalenzklassen ist die **Partitionierung** von $\boldsymbol{A}$ und wird mit $\boldsymbol{A}/_{\boldsymbol{R}}$ bezeichnet:

$$\boldsymbol{A}/_{\boldsymbol{R}} := \{[\boldsymbol{a}]_{\boldsymbol{R}} \mid \boldsymbol{a} \in \boldsymbol{\Sigma}^*\}$$

$|\boldsymbol{A}/_{\boldsymbol{R}}|$, die Anzahl der Äquivalenzklassen von $\boldsymbol{\Sigma}^*$, heißt der **Index** von $\boldsymbol{R}$.

Satz von Myhill und Nerode (4)

Satz von Myhill und Nerode

Sei $\boldsymbol{\Sigma}$ ein Alphabet und $\boldsymbol{L} \subseteq \boldsymbol{\Sigma}^*$ eine Sprache über diesem Alphabet, so sind folgende Aussagen äquivalent:

- $\boldsymbol{L} \in \boldsymbol{REG}_{\boldsymbol{\Sigma}}$
- $\boldsymbol{L}$ ist eine Vereinigung von Äquivalenzklassen einer Rechtskongruenz auf $\boldsymbol{\Sigma}^*$ mit endlichem Index.
- Der Index von $\boldsymbol{R}_{\boldsymbol{L}}$ ist endlich.

(Ohne Beweis.)

Mit dem Satz von Myhill-Nerode können wir also bestimmen, ob eine Sprache eine reguläre Sprache ist.

Der Satz ist außerdem Ausgangspunkt für die Automatenminimierung über den Automaten

$$\boldsymbol{A}_{\boldsymbol{R}_{\boldsymbol{L}}}$$

mithilfe des Markierungsalgorithmus.

Automatenminimierung (1)

Gegeben: Ein *vollständiger* DEA $\boldsymbol{A}$

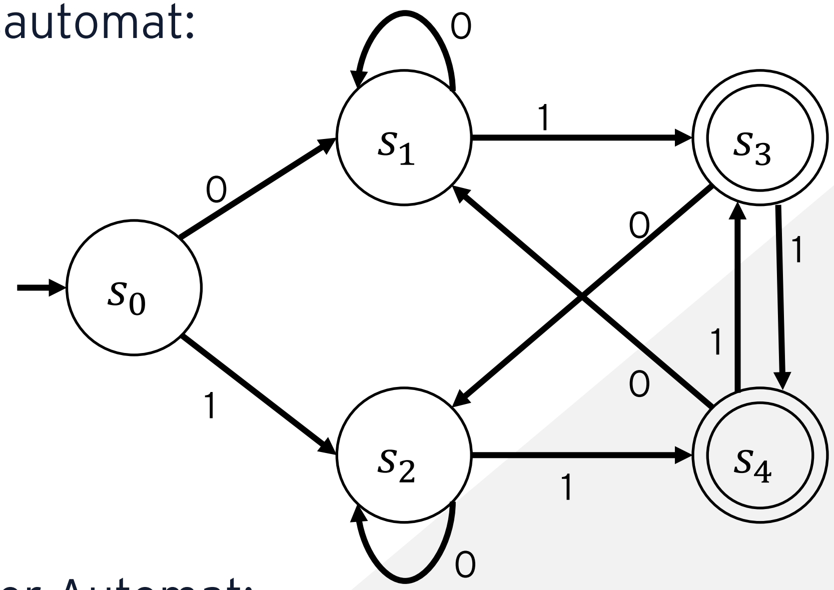
Markierungsalgorithmus:

1. Bilde eine Tabelle für alle Zustandspaare
$$\{\mathbf{s},\mathbf{t}\}, \mathbf{s},\mathbf{t} \in \mathbf{S}, \mathbf{s} \neq \mathbf{t}$$
2. Markiere alle Paare $\{\mathbf{s},\mathbf{t}\}$ mit $\mathbf{s} \notin \mathbf{F}$ und $\mathbf{t} \in \mathbf{F}$
3. Teste für jedes noch nicht markierte Paar $\{\mathbf{s},\mathbf{t}\}$ und jedes $\mathbf{a} \in \boldsymbol{\Sigma}$, ob das Paar $\{\boldsymbol{\delta}(\mathbf{s},\mathbf{a}), \boldsymbol{\delta}(\mathbf{t},\mathbf{a})\}$ schon markiert ist. Falls ja, dann markiere das Paar $\{\mathbf{s},\mathbf{t}\}$.
4. Führe Schritt 3 so lange aus, bis sich die Markierungen nicht mehr ändern.
5. Bilde für jeden Zustand $\mathbf{s}$ die Menge
$$\mathbf{S} = \{\mathbf{s}\} \cup \{\mathbf{t} | \{\mathbf{s},\mathbf{t}\} \text{ ist unmarkiert}\}$$
d.h., der Zustand $\boldsymbol{s}$ wird mit allen Zuständen $\boldsymbol{t}$ zu einem Zustand zusammengefasst, für die $\{\mathbf{s},\mathbf{t}\}$ nicht markiert ist. Wir nennen diese Mengen Blöcke.
$\boldsymbol{\Pi}$ sei die Menge dieser Blöcke.
Wir erhalten als minimalen Automaten
$$\boldsymbol{A_{min}} = (\boldsymbol{\Sigma}, \boldsymbol{\Pi}, \boldsymbol{\delta_{min}}, \boldsymbol{S_0}, \{\boldsymbol{S} \in \boldsymbol{\Pi} | \boldsymbol{S} \cap \boldsymbol{F} \neq \emptyset\})$$
mit
$$\boldsymbol{\delta_{min}}(\boldsymbol{S},\boldsymbol{a}) = \bigcup_{\boldsymbol{s} \in \boldsymbol{S}} \boldsymbol{\delta}(\boldsymbol{s},\boldsymbol{a})$$

Automatenminimierung (2)

Beispiel zum Markierungsalgorithmus:

Ausgangsautomat:



Minimierter Automat:

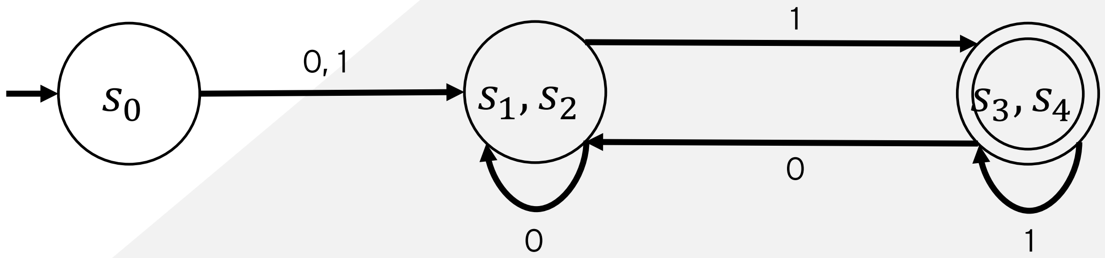


Tabelle:

s_1	2			
s_2	2			
s_3	1	1	1	
s_4	1	1	1	
	s_0	s_1	s_2	s_3

1: Markiert in Schritt 2

2: Markiert in Schritt 3

Zusammenfassung

Sprachen, die sich

- durch DEA

beschreiben lassen, können ebenfalls

- durch NEA,
- Epsilon-EA und
- Verallgemeinerte endliche Automaten

beschrieben werden.

Es gilt also:

$$\boldsymbol{DFA_\Sigma \equiv NFA_\Sigma \equiv \varepsilon FA_\Sigma \equiv GFA_\Sigma}$$

Zudem lässt sich zu jedem Automaten ein minimaler Automat $\boldsymbol{A_{R_L}}$ finden.

NORDAKADEMIE
HOCHSCHULE DER WIRTSCHAFT

NORDAKADEMIE gAG Hochschule der Wirtschaft

Köllner Chaussee 11 · 25337 Elmshorn · Tel.: +49 (0) 4121 4090-0 · E-Mail: info@nordakademie.de · Web: www.nordakademie.de