

Automatentheorie

Eigenschaften endlicher Automaten

Prof. Dr. Franz-Karl Schmatzer
schmatzf@dhbw-loerrach.de

Literatur

- C.Wagenknecht, M.Hielscher; Formale Sprachen, abstrakte Automaten und Compiler; 2.Aufl. Springer Vieweg 2014;
- Sipser M.; Introduction to the Theory of Computation; 2.Aufl.; Thomson Course Technology 2006
- Hopcroft, T. et al; Introduction to Automata Theory, Language, and Computation; 3. Aufl. Pearson Verlag 2006

Agenda

- Eigenschaften endlicher Automaten
- Produktautomat
- Pumping-Lemma
- Entscheidungsprobleme
- Reguläre Ausdrücke und Sprache
- Äquivalenz der regulären Sprachen mit den regulären Ausdrücken.

Eigenschaften regulärer Sprachen

- Seien L_1 und L_2 reguläre Sprachen über das Alphabet Σ . Dann sind auch die Sprachen L mit folgenden Eigenschaften regulär:
 - $L = L_1 \cup L_2, L_1 \cap L_2, L^c$
 - $L = L_1 \bullet L_2$ und
 - $L = L_1^*$

Die Klasse der regulären Sprachen ist abgeschlossen unter den Mengenoperationen Vereinigung, Durchschnitt, Komplement, Konkatination und Kleene-Stern.

Vereinigung

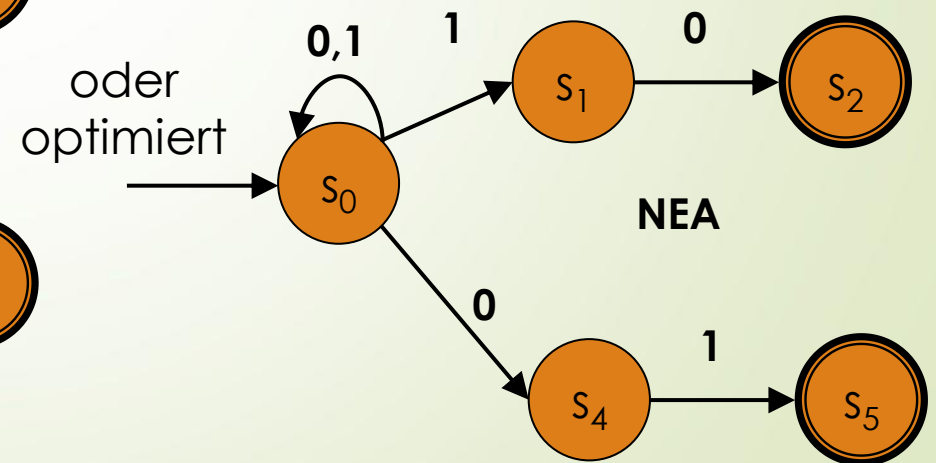
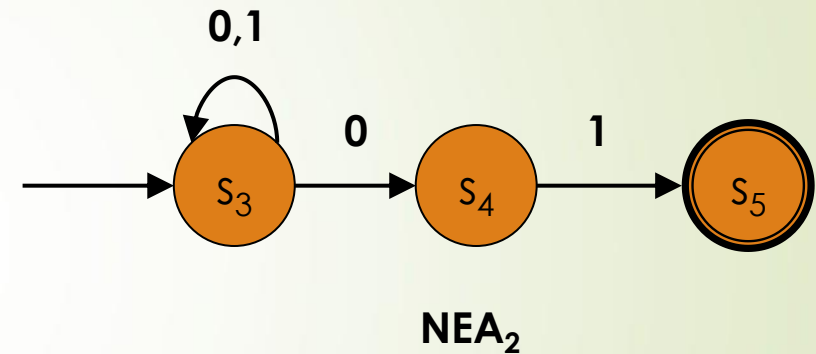
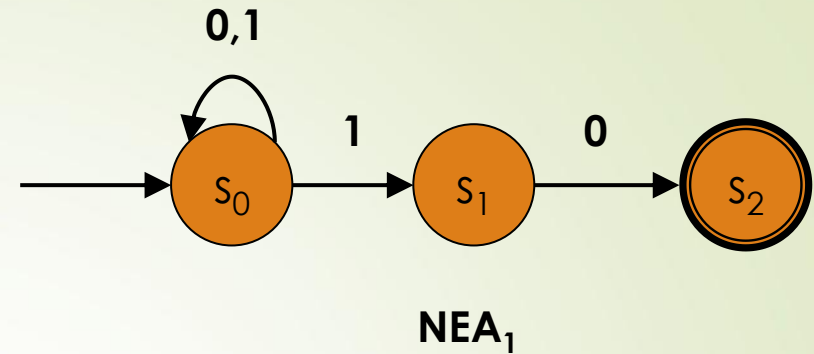
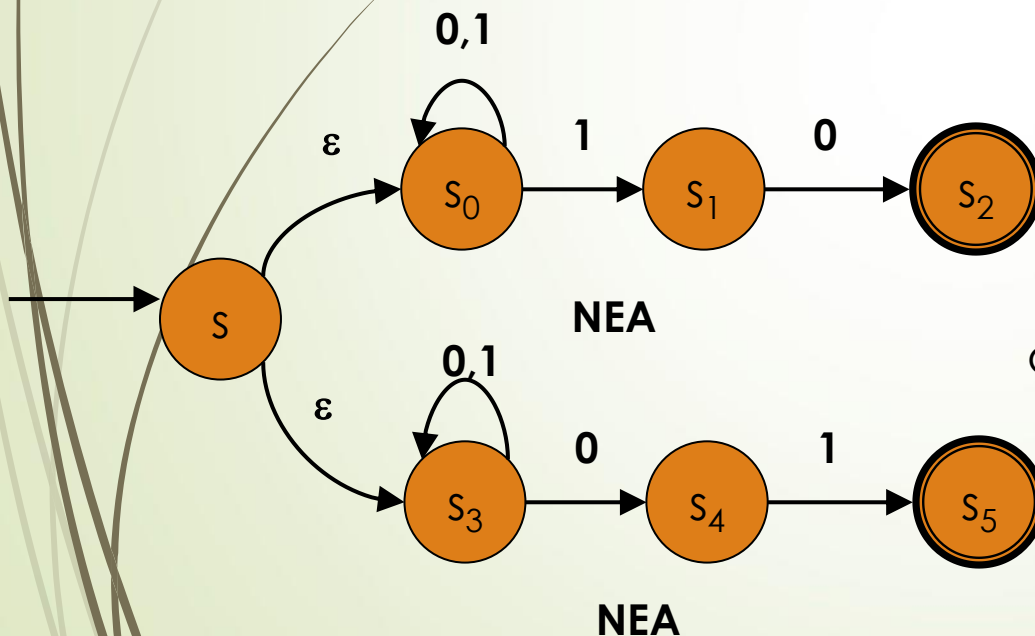
Automaten

- Für die Vereinigung nehmen wir 2 NDA (Nicht deterministische Automaten) mit
 - $NDA_1 = (\Sigma, S_1, \delta_1, s_{01}, F_1)$ und
 - $NDA_2 = (\Sigma, S_2, \delta_2, s_{02}, F_2)$ und
 - $S_1 \cap S_2 = \emptyset$, D.h. die Zustände S sind verschieden.
- Die Idee ist für ein eingegebenes Wort w nicht deterministisch zu entscheiden, mit welchem Automaten man die Worterkennung durchführt.
- Der Automat ist dann:
 - $NDA = (\Sigma, S_1 \cup S_2 \cup S, \delta_1 \cup \delta_2, S, F_1 \cup F_2)$ und die beiden Startzustände s_{01} und s_{02} werden von einem neuen Startzustand S bedient.
 - die zugehörige Überföhrungsfunktion δ ist dann, die Vereinigung der beiden Überföhrungsfunktionen δ_1 und δ_2 :
 - δ von neuen Startzustand ergibt sich zu:
 - $\delta(S, a) = \{s_{01}\}$ falls $\delta_1(s_{01}, a) = \{s_{01}\}$ und $\delta(S, a) = \{s\}$ für alle $\delta_1(s_{01}, a) = \{s\}$
 - $\delta(S, a) = \{s_{02}\}$ falls $\delta_2(s_{02}, a) = \{s_{02}\}$ und $\delta(S, a) = \{s\}$ für alle $\delta_2(s_{02}, a) = \{s\}$

Vereinigung

Automaten: Beispiel

- Automat NEA_1
- Automat NEA_2
- Summenautomat NEA



Aufgabe

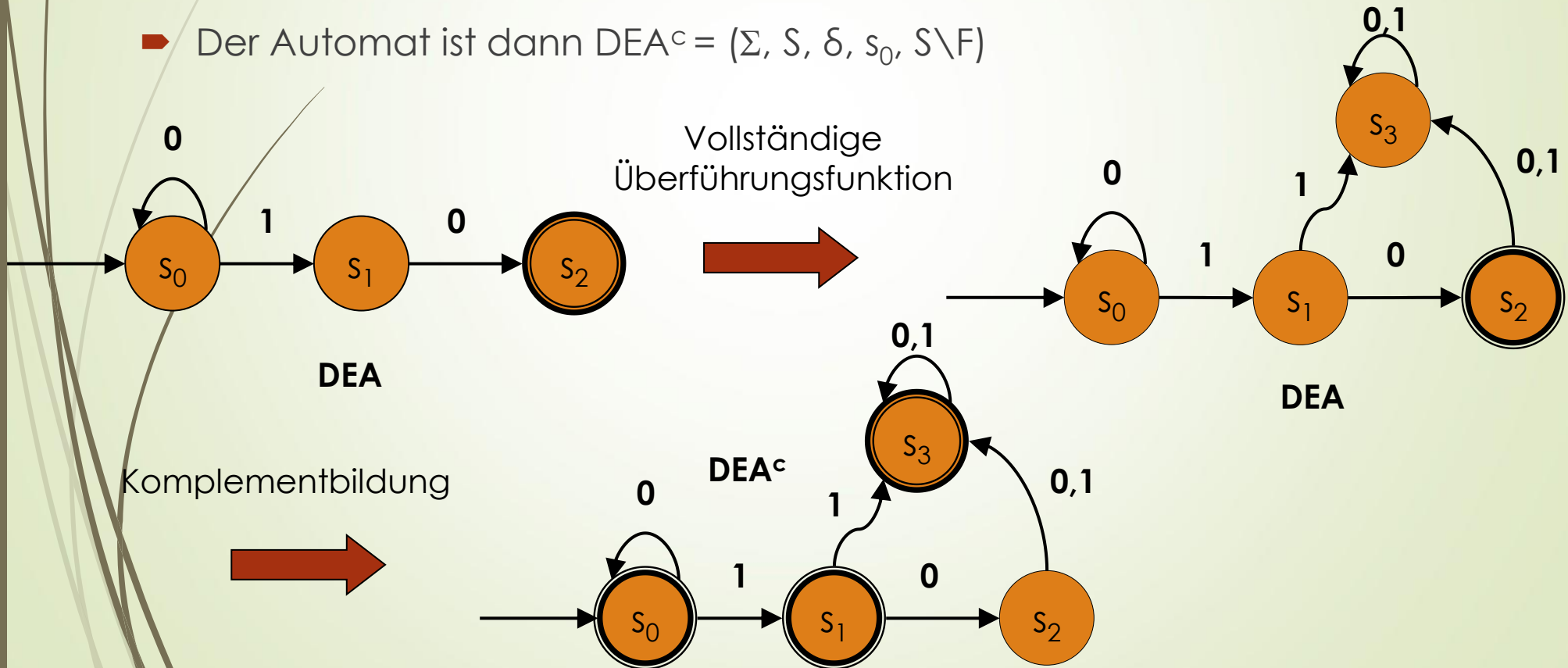
Vereinigung

- Das Alphabet sein $\{a,b,c\}$. Geben Sie den Graphen und die Überföhrungsfunktion für folgende Automaten an.
 - Erstellen Sie einen optimalen Automaten, der sowohl die Zeichenfolge „ab“ als auch die Zeichenfolge „abb“ am Schluss akzeptiert
 - Erstellen Sie einen optimalen Automaten, der die Zeichenfolge „cc“ im Wort als auch die Zeichenfolge „acb“ am Schluss akzeptiert

Komplement

Automaten

- Für das Komplement nehmen wir einen DEA (deterministischer endlicher Automaten) mit $DEA = (\Sigma, S, \delta, s_0, F)$ und vollständiger Übergangsfunktion.
- Die Idee ist, alle Zustände, die bisher keine Endzustände waren, werden Endzustände, und die Endzustände werden normale Zustände.
- Der Automat ist dann $DEA^c = (\Sigma, S, \delta, s_0, S \setminus F)$



Aufgabe

Komplement

- Das Alphabet sein $\{a,b,c\}$. Geben Sie den Graphen und die Überföhrungsfunktion für folgende Automaten an.
 - Erstellen Sie einen Automaten, der die Zeichenfolge „abb“ am Schluss nicht akzeptiert
 - Erstellen Sie einen Automaten, der die Zeichenfolge „cc“ im Wort am Schluss nicht akzeptiert

Durchschnitt

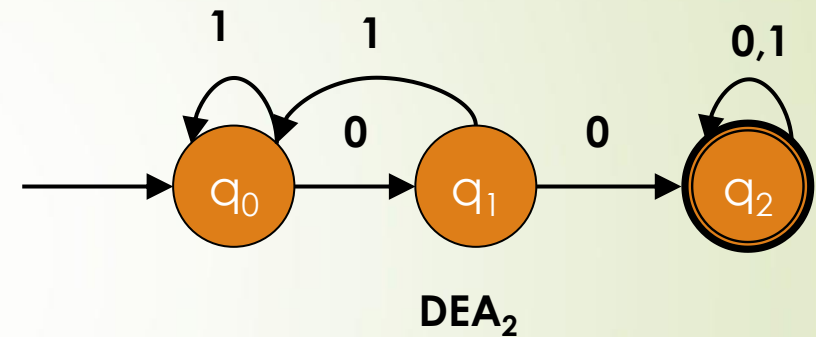
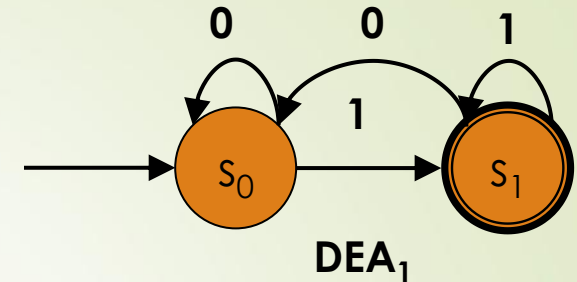
Automaten

- Der Durchschnitt zweier DEA zu dem gleichen Alphabet Σ mit
 - $DEA_1 = (\Sigma, Q_1, \delta_1, q_{01}, F_1)$ und
 - $DEA_2 = (\Sigma, Q_2, \delta_2, q_{02}, F_2)$
- Die Idee ist, die beiden Automaten parallel laufen lassen. Nur solche Worte akzeptieren, die in beiden Automaten zu Endzustände führen.
- Mathematisch ist das durch ein Kreuzprodukt zu erreichen: $DEA_1 \times DEA_2$
- Der Automat $DEA = DEA_1 \times DEA_2$ ist dann:
 - $DEA = (\Sigma, q_1 \times q_2, \delta, s_{01} \times s_{02}, F_1 \times F_2)$ mit
 - $\delta(\langle q_1, q_2 \rangle, a) = \{\langle s_1, s_2 \rangle \mid \text{falls } s_1 \in \delta_1(q_1, a) \text{ und } s_2 \in \delta_2(q_2, a)\}$

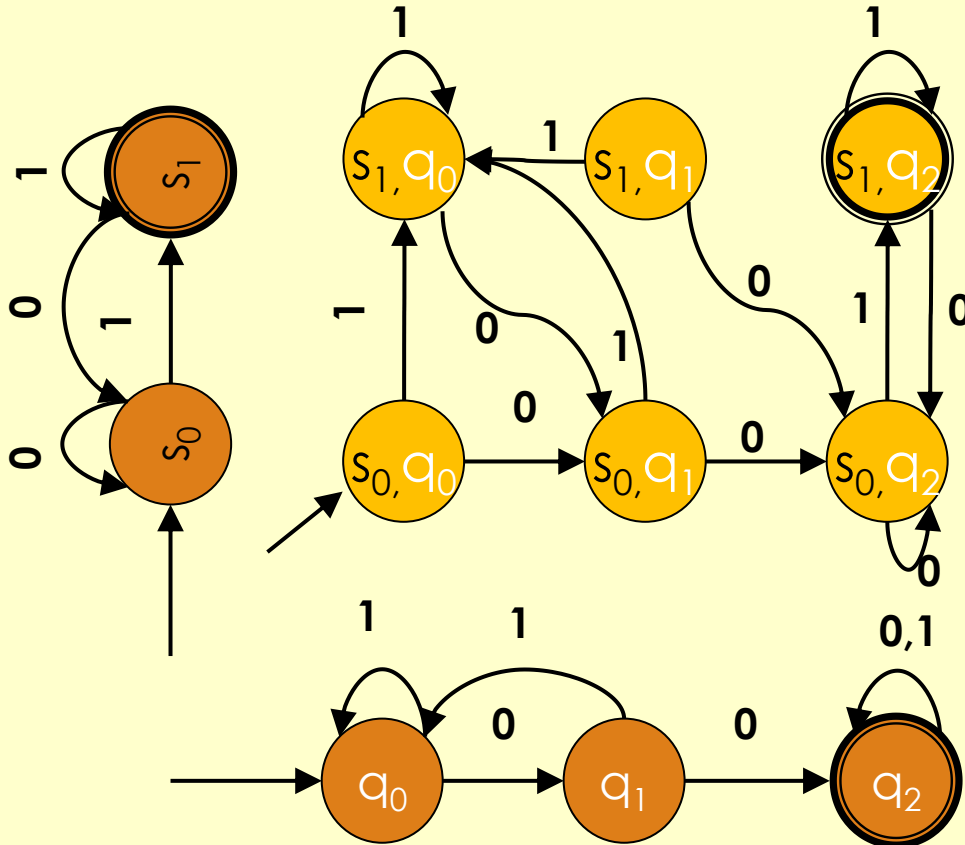
Durchschnitt

Automaten: Beispiel

- Automat DEA_1 (alle Worte enden auf 1)
- Automat DEA_2 (Im Wort kommt 00 vor)
- Summenautomat DEA



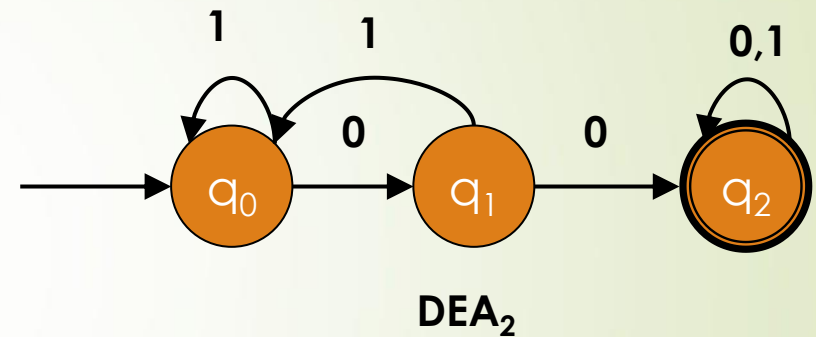
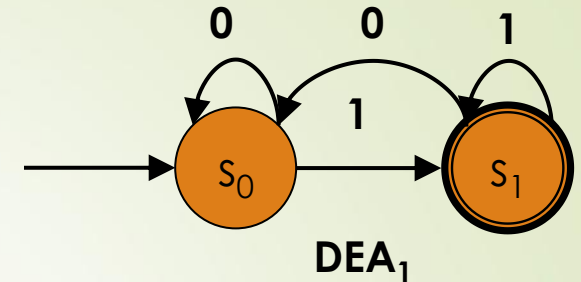
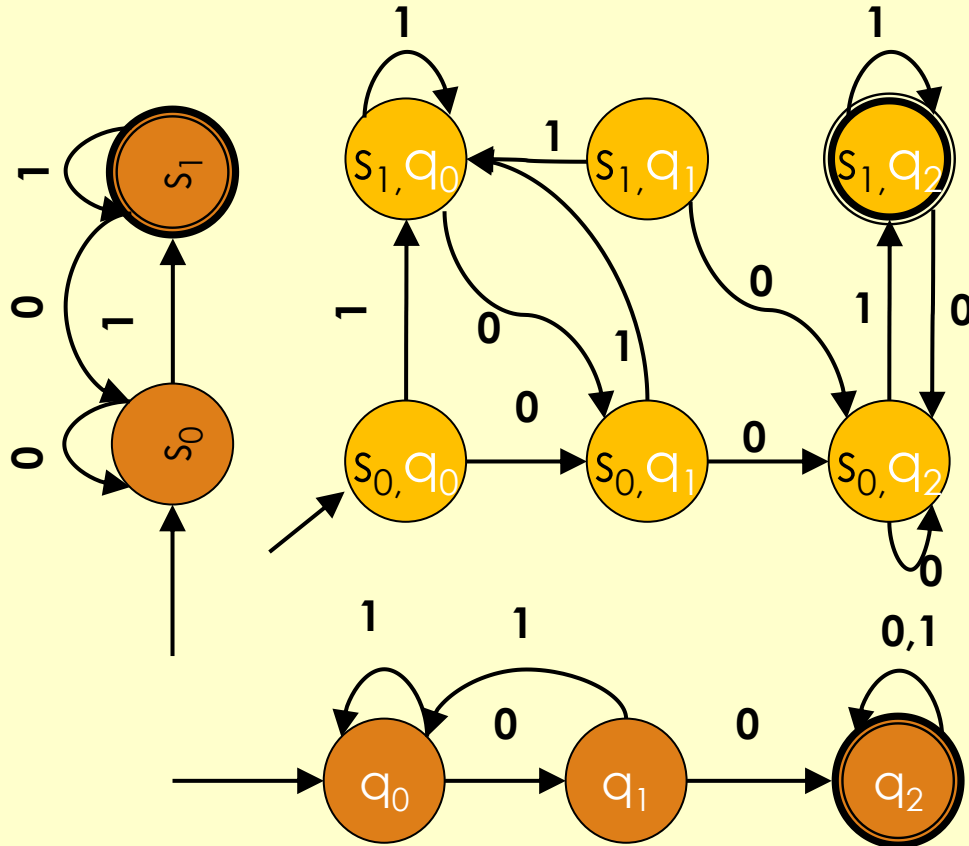
- Summenautomat DEA
- Endzustände sind alle Zustände bei denen beide Automaten zuvor Endzustände hatten.



Durchschnitt

Automaten: Beispiel

- Automat DEA_1
- Automat DEA_2
- Summenautomat DEA



- Summenautomat DEA
- Endzustände sind alle Zustände bei denen beide Automaten zuvor Endzustände hatten.

Aufgabe Produktautomat einzelnen Automaten

Aufgabe:

Ein DEA soll Worte über dem Alphabet $\Sigma = \{a,b,c\}$ Worte erkennen, die sowohl die Teilstrings $s_1 = cc$ als auch die Teilstrings $s_2 = ac$ enthalten.

- Zuerst die beiden Teilautomaten erstellen:

Eigenschaften regulärer Sprachen

Pumping-Lemma Einführung

- Es gibt recht einfach gebaute Sprachen, die nicht regulär sind.
 - $L = \{ 1^k \mid \text{mit } k = 2^n \text{ und } n \geq 0 \} = \{ 1, 11, 1111, 1^8, 1^{16}, \dots \}$
 - $L = \{ 0^n 1^n \mid n \geq 0 \} = \{ \varepsilon, 01, 0011, 000111, \dots \}$
- Wie erkennt man das?
- Ein endlicher Automat hat nur endliche viele Zustände N
- Ein Wort w mit $|w| > N$ muss mindestens einen Zustand 2 mal erreichen. Sei z_i dieser Zustand.
- Wir zerlegen das Wort in 3 Teile u, v, w mit
 - Das Teilwort u führt vom Startzustand s_0 zu s_i
 - Das Teilwort v führt von s_i zu s_i
 - Das Teilwort w führt von s_i zu Endzustand s_F
 - $(s_0, uvw) \rightarrow^* (s_i, vw) \rightarrow^* (s_i, w) \rightarrow^* (s_F, \varepsilon)$
 - Da das Teilwort v immer von s_i zu s_i führt, müssen in dieser Sprache auch alle Worte mit $uv^m w$ mit $m \geq 0$ vorkommen.

Eigenschaften regulärer Sprachen

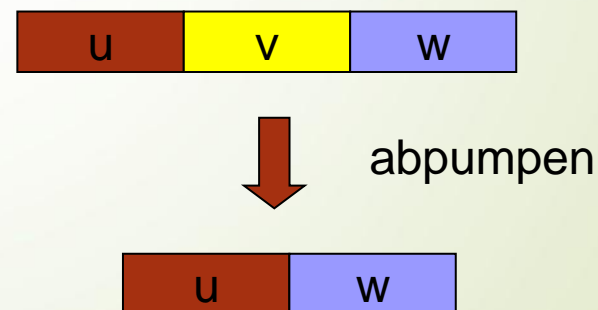
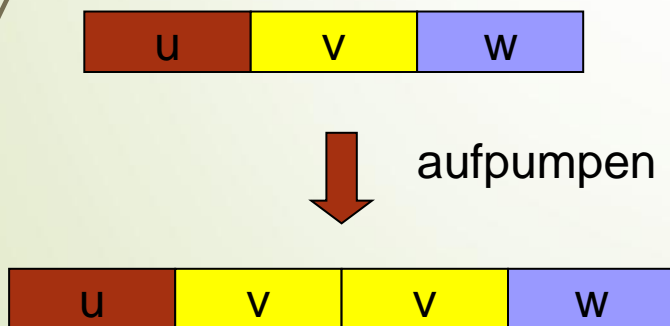
Pumping-Lemma Definition

Pumping-Lemma für reguläre Sprachen

Sei L eine reguläre Sprache. Dann existiert eine Zahl $N \geq 0$, so dass sich jedes Wort $x \in L$ mit $|x| \geq N$ in der folgenden Form schreiben lässt.

$x = uvw$ mit $|v| \geq 1$ und $|uv| \leq N$,

für alle $i \geq 0$ gilt, dass $uv^i w \in L$ ist.



Eigenschaften regulärer Sprachen

Pumping-Lemma Beispiel

- $L = \{ 0^n 1^n \mid n \geq 0 \}$ ist nicht regulär
- Annahme L wäre regulär und $x = 0^n 1^n$ mit $n > N$, dann muss eine Aufspaltung von x in $x = uvw$ geben und $xv^k w$ mit $k \geq 0$ wäre auch ein erlaubtes Wort.
 - Sei v ganz in der ersten Hälfte d.h $v = 0^j$ mit einem $j > 0$ dann müssten auch die Wörter $w = 0^{n-j+kj} 1^n \in L$ sein \Rightarrow Widerspruch!
 - Sei v ganz in der zweiten Hälfte d.h $v = 1^j$ mit einem $j > 0$ dann müssten auch die Wörter $w = 0^n 1^{n-j+kj} \in L$ sein \Rightarrow Widerspruch!
 - Sei v dazwischen mit $v = 0^l 1^m$ mit $l, m > 0$, dann müssten auch die Wörter mit $w = 0^{n-l} (0^l 1^m)^k 1^{n-m} \in L$ mit $k \geq 0$ sein. \Rightarrow Widerspruch!
- $\Rightarrow L$ ist nicht regulär.

Entscheidungsprobleme für reguläre Sprachen

- Der Zweck eines endlichen Automaten ist die Prüfung, ob ein gegebenes Wort zu seiner Sprache gehört. (Wortproblem)
- Auch andere Entscheidungsprobleme sind hier relevant.
- Für reguläre Sprachen können diese alle mit ja beantwortet werden.

Problem	Gegeben	Gefragt	Entscheidbar
Wortproblem	L und $w \in \Sigma^*$	Gilt $w \in L$?	Ja
Leerheitsproblem	L	Gilt $L = \emptyset$?	Ja
Endlichkeitsproblem	L	Gilt $ L < \infty$?	Ja
Äquivalenzproblem	L_1 und L_2	$L_1 = L_2$?	Ja

Grenzen endlicher Automaten

- Endliche Automaten akzeptieren nur reguläre Sprachen
- Sprache wie:
 - $L(P) = \{vv^r \mid v \in E^*\}$ (Palindrome)
mit v^r gespiegelte Version von v
 - $L(P) = \{a^n b^n \mid n \geq 0\}$
 - $L(P) = \{a^p \mid p \text{ eine Primzahl}\}$
- sind nicht regulär und können von einem endlichen Automaten nicht erkannt werden.
- Erweitern der reguläre Sprache und endliche Automaten
 - kontextfreie Sprachen Typ-2-Grammatiken
 - Automaten mit einem unendlichen Speicher, der aber nur das oberste Elemente verändern kann (Stack)
 - Diese beiden Konzepte sind äquivalent

Aufgabe

Grenzen der endlichen Automaten

Zeigen Sie, dass folgende Sprachen nicht regulär sind:

1. $L = \{0^n \mid n \text{ ist eine Quadratzahl}\}$
2. $L = \{0^n \mid n \text{ ist eine Potenz von } 2\}$
3. $L = \{ww \mid \text{mit } w \in \Sigma^* \text{ und } \Sigma = \{0,1\}\}$

Reguläre Ausdrücke

- Reguläre Ausdrücke sind beschreibende Konzepte für reguläre Sprachen.
- Die Menge der reguläre Ausdrücke über E ist die Menge der Wörter über $E \cup \{\Lambda, E, \bullet, |, *, [,]\}$, die nach folgender Regeln gebildet werden.
 - Der Nulloperator Λ ist ein regulärer Ausdruck
 - Der Einsoperator E ist ein regulärer Ausdruck
 - Jedes Zeichen $e \in E$ ist ein regulärer Ausdruck
 - Sind v und w reguläre Ausdrücke, dann auch
 - $v \bullet w = vw$
 - $[v \mid w]$ (v oder w)
 - $v^* = vvvvv$ (beliebig viele v verknüpft)
- Reguläre Ausdrücke und endliche Automaten sind äquivalent

Reguläre Ausdrücke

Beispiele

➤ $\gamma = [0 \cdot [0 \mid 1]^*]$ $w = \{00, 01, 000, 001, 010, 011, \dots\}$

➤ $\gamma = [1^* 0 \cdot 0 \cdot [0 \mid 1]^*] = ?$

Reguläre Sprache

- Die von einem regulären Ausdruck erzeugte Sprache ist:
 - $L(\Lambda) = \emptyset$; Λ definiert die leere Sprache
 - $L(E) = \{\epsilon\}$: L legt die Sprache fest, die nur das leere Wort enthält
 - $L(a) = \{a\}$ für alle $a \in \Sigma$; Sprache die nur das einzige Zeichen a enthält.
 - Sind v und w reguläre Ausdrücke, dann auch
 - $L(v \cdot w) = L(v) \cdot L(w)$
 - $L([v \mid w]) = L(v) \cup L(w)$
 - $L(v^*) = (L(v))^*$
- Beispiel: regulärer Ausdruck: $\gamma = 0 \cdot (0+1)^*$
 - $L(\gamma) = L(0) \cdot L((0+1)^*) = L(0) \cdot (L(0+1))^* =$
 - $L(0) \cdot (L(0) \cup L(1))^* = \{0\}(\{0\} \cup \{1\})^* = \{0\}(\{0,1\})^* = 0(0+1)^*$ (Notation)
 - $0(0+1)^* = \{0, 00, 01, 000, 010, \dots\}$

Reguläre Sprachen

Rechenregeln 1

Algebraische Regeln. Seien L, M, N reguläre Sprachen dann gilt:

1. Kommutativität und Assoziativität

- $L + M = M + L$ (Kommutativgesetz für die Vereinigung)
- $L \cdot M \neq M \cdot L$ (Kommutativgesetz für die Konkationation gilt i.a nicht)
- $(L + M) + N = L + (M + N)$ (Assoziativgesetz für die Vereinigung)
- $(L \cdot M) \cdot N = L \cdot (M \cdot N)$ (Assoziativgesetz für die Vereinigung)

2. Identität und Vernichtung

- $\emptyset + L = L + \emptyset = L$ (Identität für die Vereinigung)
- $\varepsilon L = L \varepsilon = L$ (Identität für die Konkationation)
- $\emptyset L = L \emptyset = \emptyset$ (Vernichtung für die Konkationation)

3. Distributivgesetze

- $L \cdot (M + N) = L \cdot M + L \cdot N$ (Links Distributivität der Konkationation)
- $(M + N) \cdot L = M \cdot L + N \cdot L$ (rechts Distributivität der Konkationation)

4. Idempotenz

- $L + L = L$ (Idempotenz der Vereinigung)

Reguläre Sprachen

Rechenregeln 2

5. Gesetze mit dem Kleeneschen Sternoperator

➤ $(L^*)^* = L^*$

➤ $\emptyset^* = \varepsilon$

➤ $\varepsilon^* = \varepsilon$

➤ $L^+ = LL^*$

➤ $L^* = L^+ + \varepsilon$

➤ $L? = \varepsilon + L$

6. Weitere Folgerungen mit dem Kleeneschen Sternoperator

➤ $(L+M)^* = (L^*M^*)^*$

➤ $(L^*+M)^* = (L+M)^*$

➤ $(L+M)^* = L(L+M)^* + M(L+M)^* + \varepsilon$

➤ $(L+M)^* = (L+M)^*LM(L+M)^* + M^*L^*$

Aufgabe

Umformen von regulären Ausdrücken

Zeigen Sie die Äquivalenz der folgenden regulären Ausdrücke:

1. $(0^*1)^*0^* = 0^*(10^*)^*$
2. $(0^*111)^*0^* = 0^*(1110^*)^*$
3. $(00)^*(\varepsilon + 0) = 0^*$
4. $(0+1)^*0(0+1)^*1(0+1)^* = (0+1)^*01(0+1)^*$
5. $(0+1)^*01(0+1)^* + 1^*0^* = (0+1)^*$

Äquivalenz

endliche Automaten \Leftrightarrow reguläre Ausdrücke

- Die Klasse der Sprachen, welche durch reguläre Ausdrücke erzeugt werden, sind äquivalent zu den Klassen von Sprachen, die durch endliche Automaten erzeugt werden.
- Zu zeigen:
 1. Jede Sprache $L(A)$ eines endlichen Automaten A kann durch die Sprache $L(R)$ eines regulären Ausdruck R erzeugt werden.
 2. Jede Sprache $L(R)$ eines regulären Ausdrucks R kann auch durch die Sprache $L(A)$ eines endlichen Automaten A erzeugt werden.

Äquivalenz

endliche Automaten \Rightarrow reguläre Ausdrücke

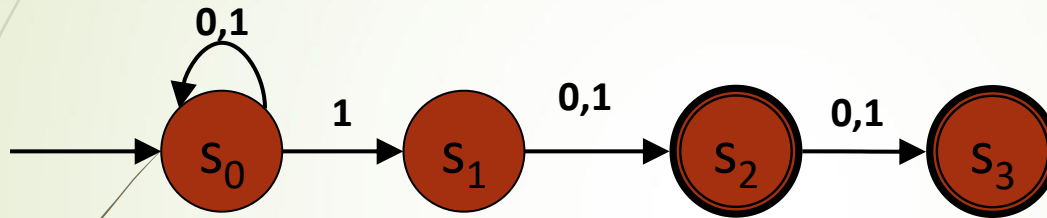
Beweisidee :

- Elimination aller Zustände bis nur noch der Start- und ein Endzustand übrig bleiben.
 - Statt einzelne Zeichen aus dem Alphabet Σ als Übergänge zwischen 2 Zustände werden nun reguläre Ausdrücke als Übergänge zugelassen.
 - Dies erlaubt Schritt für Schritt alle innere Zustände zu entfernen.
 - Endzustände werden am Schluss entfernt.
 - Technisch führt man einen neuen Startzustand S und einen neuen Endzustand S_E ein, welche mit ε -Übergänge mit den entsprechenden Startzustand bzw. mit den entsprechenden Endzuständen verbunden sind.

endliche Automaten \Rightarrow reguläre Ausdrücke

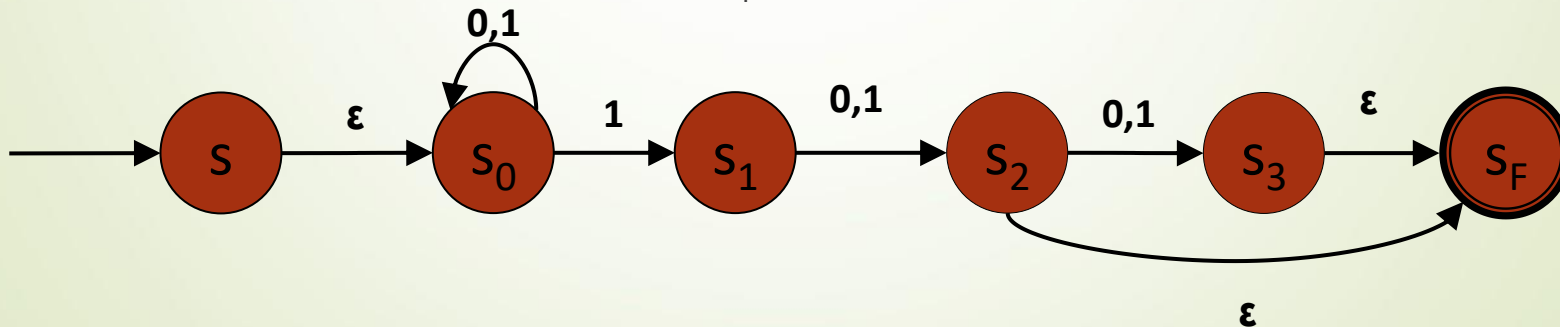
Beispiel

➤ Beispiel: $A = (\{s_0, s_1, s_2, s_3\}, \{0, 1\}, \delta, \{s_0\}, \{s_2, s_3\})$



δ	0	1
s_0	$\{s_0\}$	$\{s_0, s_1\}$
s_1	$\{s_2\}$	$\{s_2\}$
s_2	$\{s_3\}$	$\{s_3\}$
s_3	\emptyset	\emptyset

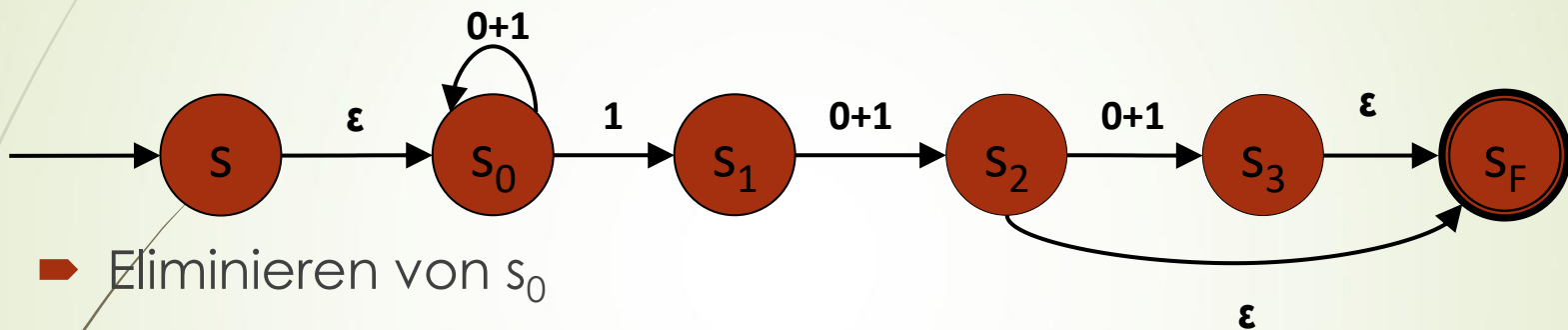
➤ Einführen eines neuen Startzustands S und eines neuen Endzustands S_F



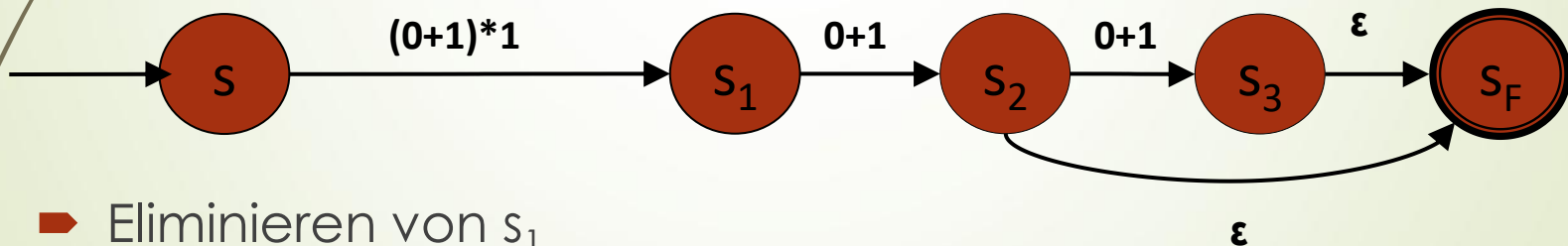
endliche Automaten \Rightarrow reguläre Ausdrücke

Beispiel

- Übergänge in reguläre Ausdrücke umformulieren



- Eliminieren von s_0

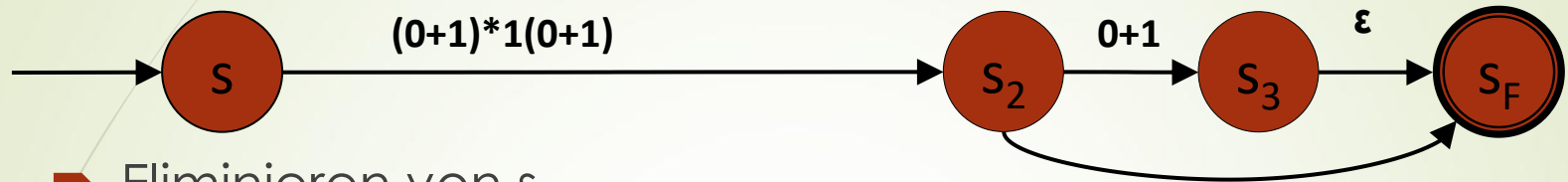


- Eliminieren von s_1

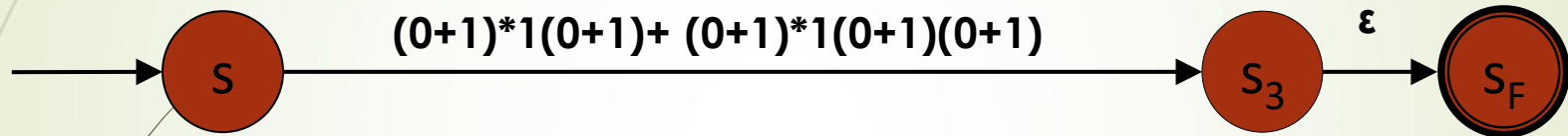


endliche Automaten \Rightarrow reguläre Ausdrücke

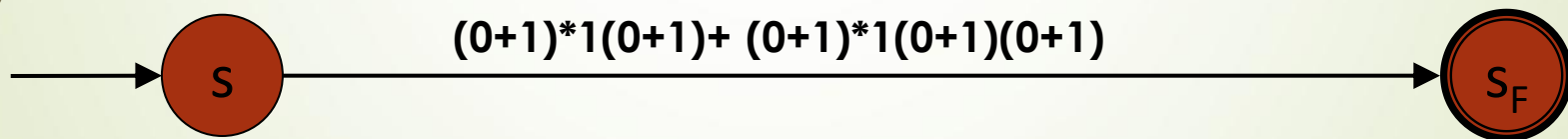
Beispiel



➤ Eliminieren von s_2



➤ Eliminieren von s_3



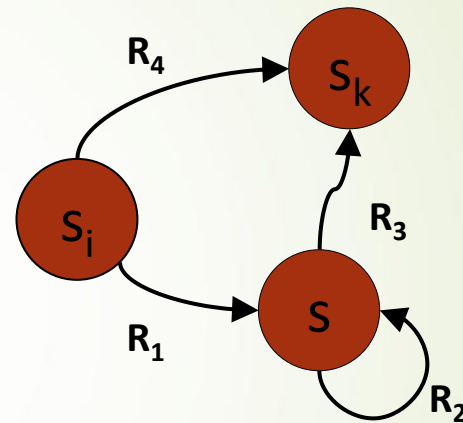
➤ Der reguläre Ausdruck $R = (0+1)^*1(0+1)(0+1) + (0+1)^*1(0+1)$ definiert alle Worte, die die Maschine akzeptiert, d.h. $L(R) = L(A)$

➤ $L = (0+1)^*1(0+1)((0+1)+\epsilon)$ (vereinfacht)

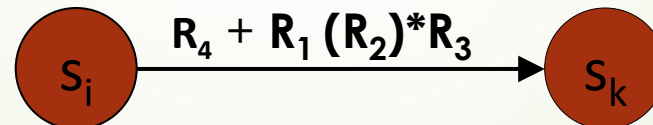
endliche Automaten \Rightarrow reguläre Ausdrücke

Allgemein: Elimination eines Zustandes

➔ Eliminieren von S



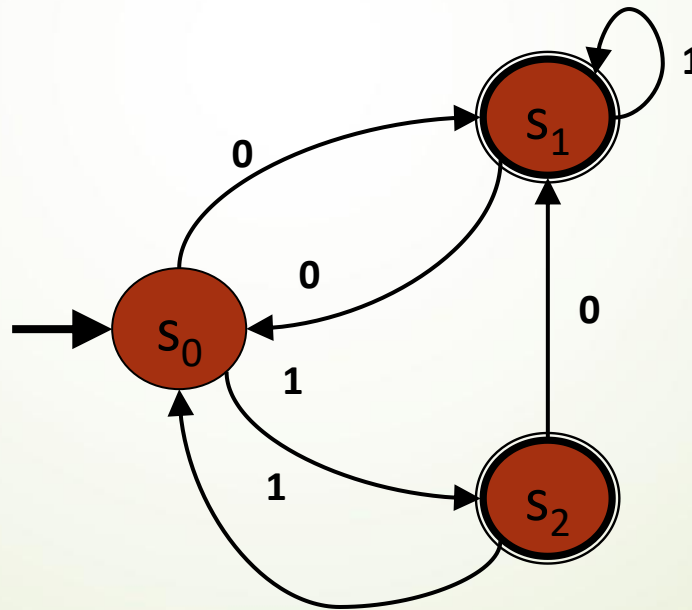
➔ ergibt:



Aufgabe

Umwandeln in einen regulären Ausdruck

- Wandeln Sie die Sprache des folgenden DEA in einen äquivalenten regulären Ausdruck um.

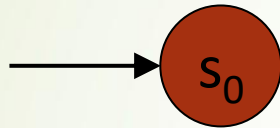


Äquivalenz

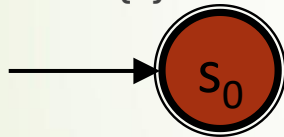
reguläre Ausdrücke \Rightarrow endliche Automaten

➤ Automat der

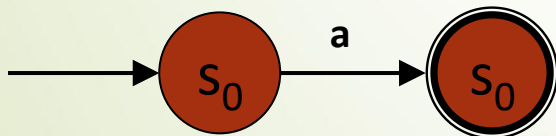
- die leere Sprache akzeptiert



- die Sprache $\{\epsilon\}$ akzeptiert

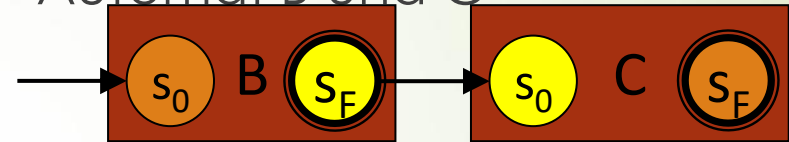


- die Sprache $\{a\}$ akzeptiert



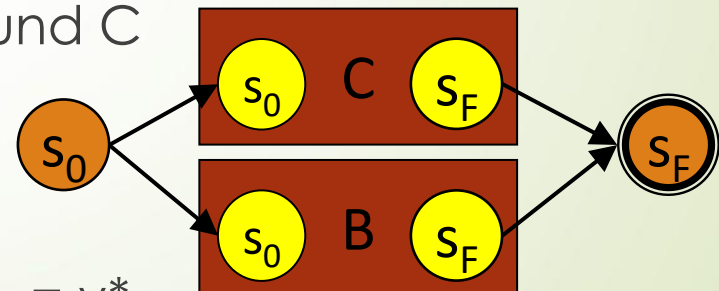
- Sei $\alpha = \beta \gamma$

- Hintereinanderschalten von Automat B und C

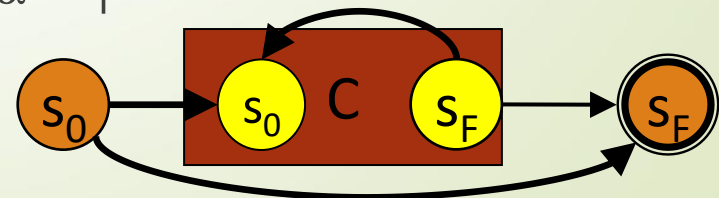


- Sei $\alpha = \beta \mid \gamma$

- Parallelschalten von Automat B und C



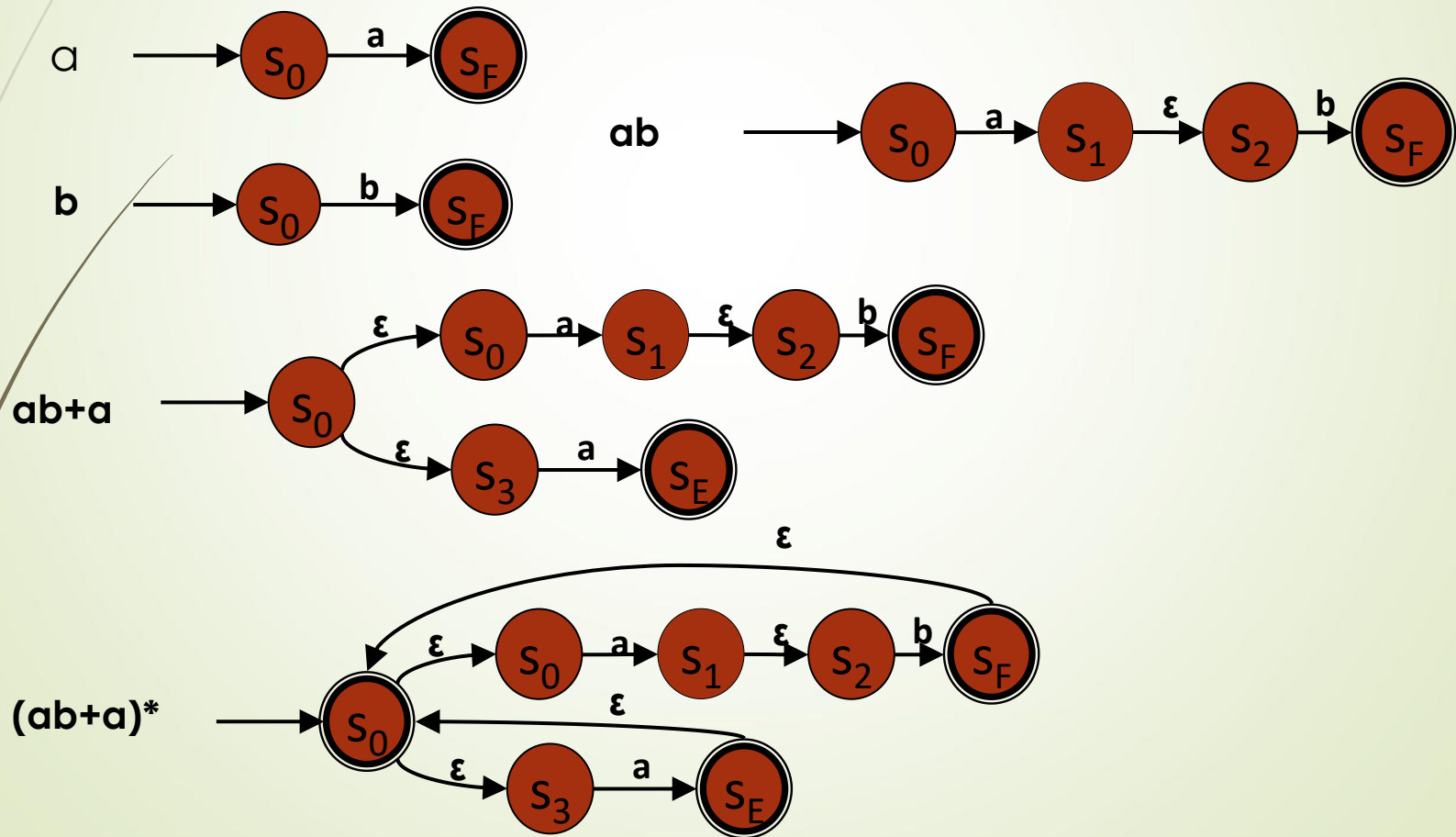
- $\alpha = \gamma^*$



Äquivalenz

Beispiel: reguläre Ausdrücke \Rightarrow endliche Automaten

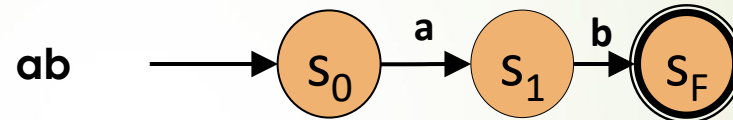
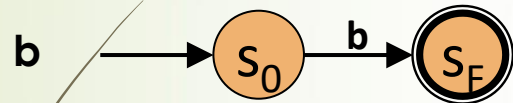
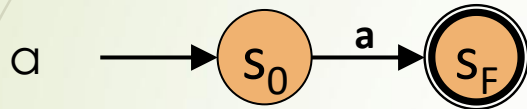
- Sei $R = (ab+a)^*$
- Formale Konstruktion



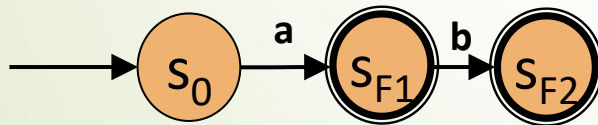
Äquivalenz

Beispiel: reguläre Ausdrücke \Rightarrow endliche Automaten

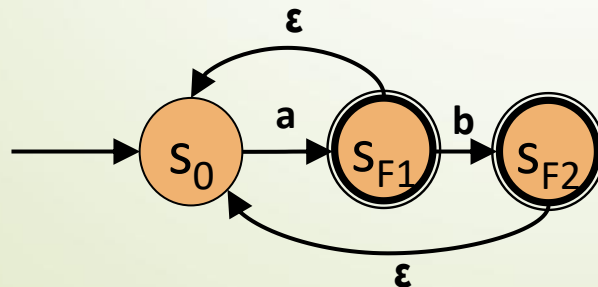
- Sei $R = (ab+a)^*$
- Vereinfachte konstruktion



ab+a



(ab+a)*



Aufgabe

reguläre Ausdrücke

Konstruieren Sie einen deterministischen endlichen Automat der folgende Sprachen über dem Alphabet $\Sigma = \{a,b\}$ akzeptiert:

1. $L = (ab)^*$
2. $L = (a+b)(aa)^*$
3. $L = b^*a(a+b)$