

Automaten und Formale Sprachen

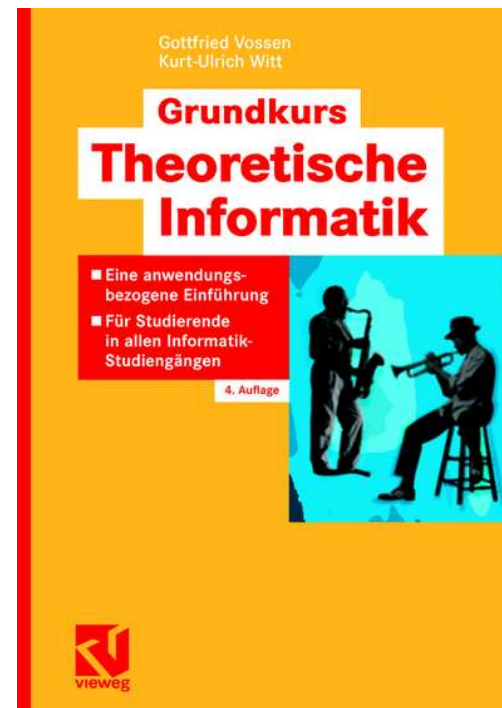
ϵ -Automaten und Minimierung

Ralf Möller

Hamburg Univ. of Technology

Literatur

- **Gottfried Vossen, Kurt-Ulrich Witt:**
Grundkurs Theoretische Informatik,
Vieweg Verlag



Danksagung

- Kurs basiert auf Präsentationsmaterial von
 - ◆ Thomas Ottmann (Uni Freiburg)

Deterministische endliche Automaten

Ein **deterministischer endlicher Automat (DFA)** ist gegeben durch

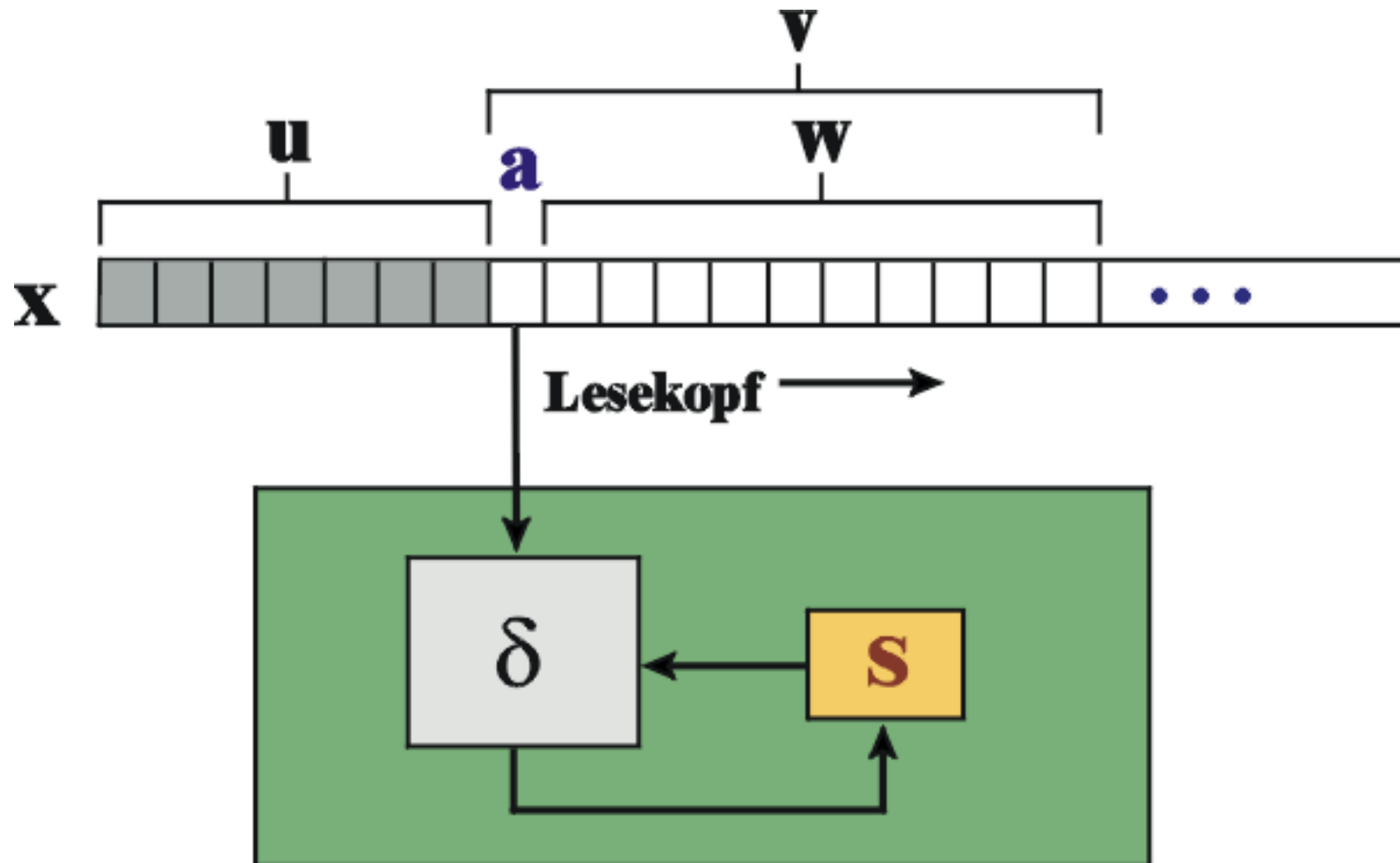
- eine endliche Menge S von **Zuständen**
- eine endliche Menge Σ von **Eingabezeichen**
- einen **Anfangszustand** $s_0 \in S$
- eine **Endzustandsmenge** $F \subseteq S$
- eine **Übergangsfunktion** $\delta : S \times \Sigma \rightarrow S$

Kurz: $A = (\Sigma, S, \delta, s_0, F)$

δ kann auch durch einen **Zustandsübergangs Graphen** oder als Menge von Tripeln (s, a, t) mit $\delta(s, a) = t$ gegeben sein

δ ist manchmal nicht total (überall definiert)

Konfiguration eines endlichen Automaten



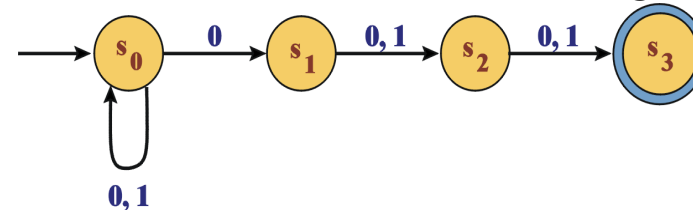
Nichtdeterministische endliche Automaten

Ein Nichtdeterministischer endlicher Automat (NFA) besteht aus

- einer endlichen Menge S von **Zuständen**
- einer endlichen Menge Σ von **Eingabezeichen**
- Einer Menge von **Anfangszuständen** $S_0 \subseteq S$
- einer **Endzustandsmenge** $F \subseteq S$
- einer **Zustandsübergangsrelation** $\delta \subseteq S \times \Sigma \times S$

Kurz: $A = (S, \Sigma, \delta, S_0, F)$

δ kann als Menge von Tripeln (s, a, t) oder als Tabelle mit Mengeneinträgen notiert werden, Bsp.:



$\delta = \{(s_0, 0, s_0), (s_0, 0, s_1), (s_0, 1, s_0), (s_1, 1, s_2), (s_2, 0, s_3), (s_2, 1, s_3)\}$

Äquivalenz von DFA und NFA

Satz: Zu jedem NFA $A = (S, \Sigma, \delta, S_0, F)$ kann man einen DFA A' konstruieren, so dass $L(A) = L(A')$.

Beweis: Mit Hilfe der
Potenzmengenkonstruktion.

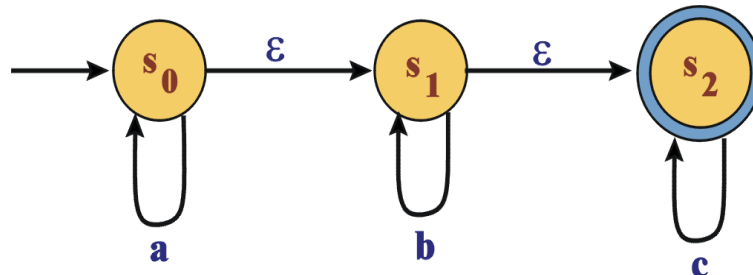
Motivation endlicher Automaten mit ε -Übergängen

- Bei DFA und NFA wird bei jedem Zustandsübergang immer genau ein Zeichen aus Σ verbraucht
- Für bestimmte Aufgaben lassen sich die Automaten einfacher entwerfen, wenn es möglich ist, Nachfolgezustände durch ε -Transitionen zu erreichen

Bsp.: Entwerfe Automaten $A_{\varepsilon abc}$ für die Sprache

$$L_{abc} = \{a^i b^j c^k; i, j, k \geq 0\}$$

- $A_{\varepsilon abc}$ darf beliebig viele a lesen, dann beliebig viele b, anschließend beliebig viele c
- Problem: Woher weiß $A_{\varepsilon abc}$, wann eine Folge gleicher Zeichen zu Ende ist?
- Lösung: $A_{\varepsilon abc}$ könnte irgendwann ohne zu lesen in neuen Zustand umschalten



Endlicher Automat mit ε -Übergängen

- Ein **endlicher ε -Automat** (ε FA) ist ein Quintupel $A = (S, \Sigma, \delta, S_0, F)$;
- Dabei sind S, Σ, S_0 und F wie bei NFA definiert, und δ ist die **Zustandsübergangsrelation**, die auch ε -Transitionen zulässt:
$$\delta \subseteq S \times (\Sigma \cup \{\varepsilon\}) \times S$$
- Konfigurationen $k = (s, w)$ sind wie bei NFA definiert
- Für Konfigurationsübergänge gilt:
 $(s, aw) \vdash (t, w)$ gdw. $(s, a, t) \in \delta, a \in \Sigma \cup \{\varepsilon\}, w \in \Sigma^*$
- Die von einem ε FA A **akzeptierte Sprache** ist dann wieder
$$L(A) = \{w \in \Sigma^*; (s_0, w) \vdash^* (s, \varepsilon), s_0 \in S_0, s \in F\}$$

Äquivalenz von ϵ -Automaten und NFA

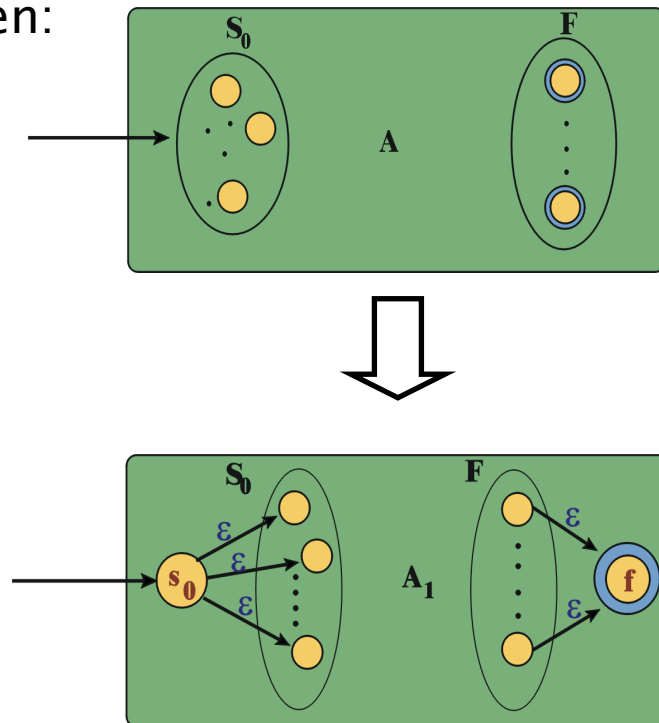
Satz: Die Klasse der jeweils von ϵ -Automaten und NFA akzeptierbaren Sprachen sind gleich.

Beweis:

1. Jeder NFA ist ein spezieller ϵ FA (ohne ϵ -Übergänge)
2. Wir konstruieren zu einem beliebig gegebenen ϵ FA $A = (S, \Sigma, \delta, S_0, F)$ in vier Schritten einen NFA A' , der dieselbe Sprache akzeptiert, für den also $L(A) = L(A')$ ist.

Transformation von ϵ FA in NFA, 1. Schritt

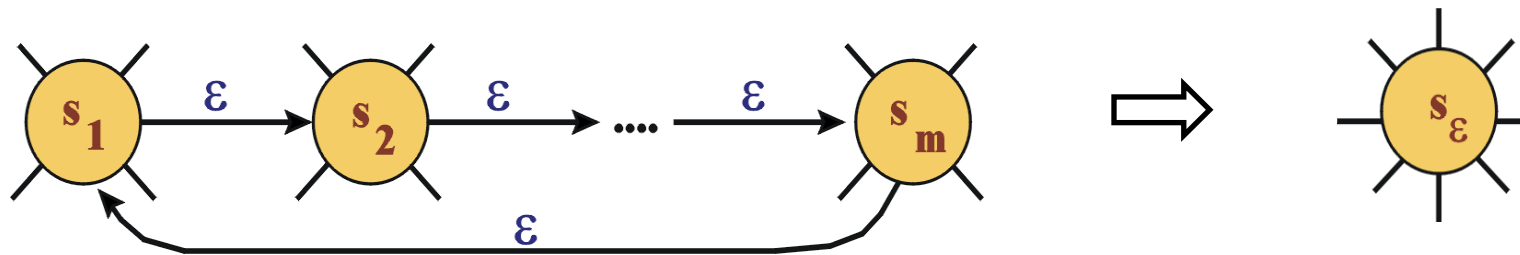
A erhält einen neuen Startzustand s_0 und einen neuen Endzustand f , die mit den bisherigen Start- bzw. Endzuständen über ϵ -Transitionen verbunden werden:



(Falls A nur einen Start- oder Endzustand hat, kann auf das Schaffen neuer Zustände verzichtet werden.)

Transformation von ϵ FA in NFA, 2. Schritt

Entferne alle ϵ -Zykel, Zustandsfolgen s_1, s_2, \dots, s_m , die zyklisch über ϵ -Transitionen durchlaufen werden können, ersetze sie durch neuen Zustand s_ϵ . Zu und von s_ϵ führen alle Transitionen der ersetzten Zustände.

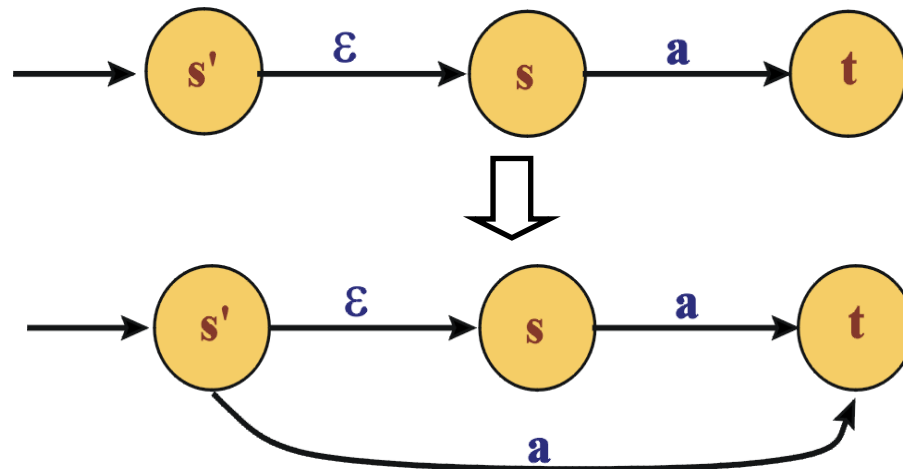


Anschließend werden alle Transitionen der Art (s, ϵ, s) entfernt.

Transformation von ϵ FA in NFA, 3. Schritt

Jetzt noch enthaltene ϵ -Transitionen werden durch alternative Übergänge ergänzt, die Zeichen aus Σ benötigen. (Ziel ist es, die ϵ -Transitionen überflüssig zu machen.)

Kommt man mit (erst ϵ dann a) von s' nach t , dann auch mit (nur a).



Diese Operation wird so oft angewandt, bis sich keine Änderungen mehr ergeben.

Transformation von ϵ FA in NFA, 4. Schritt

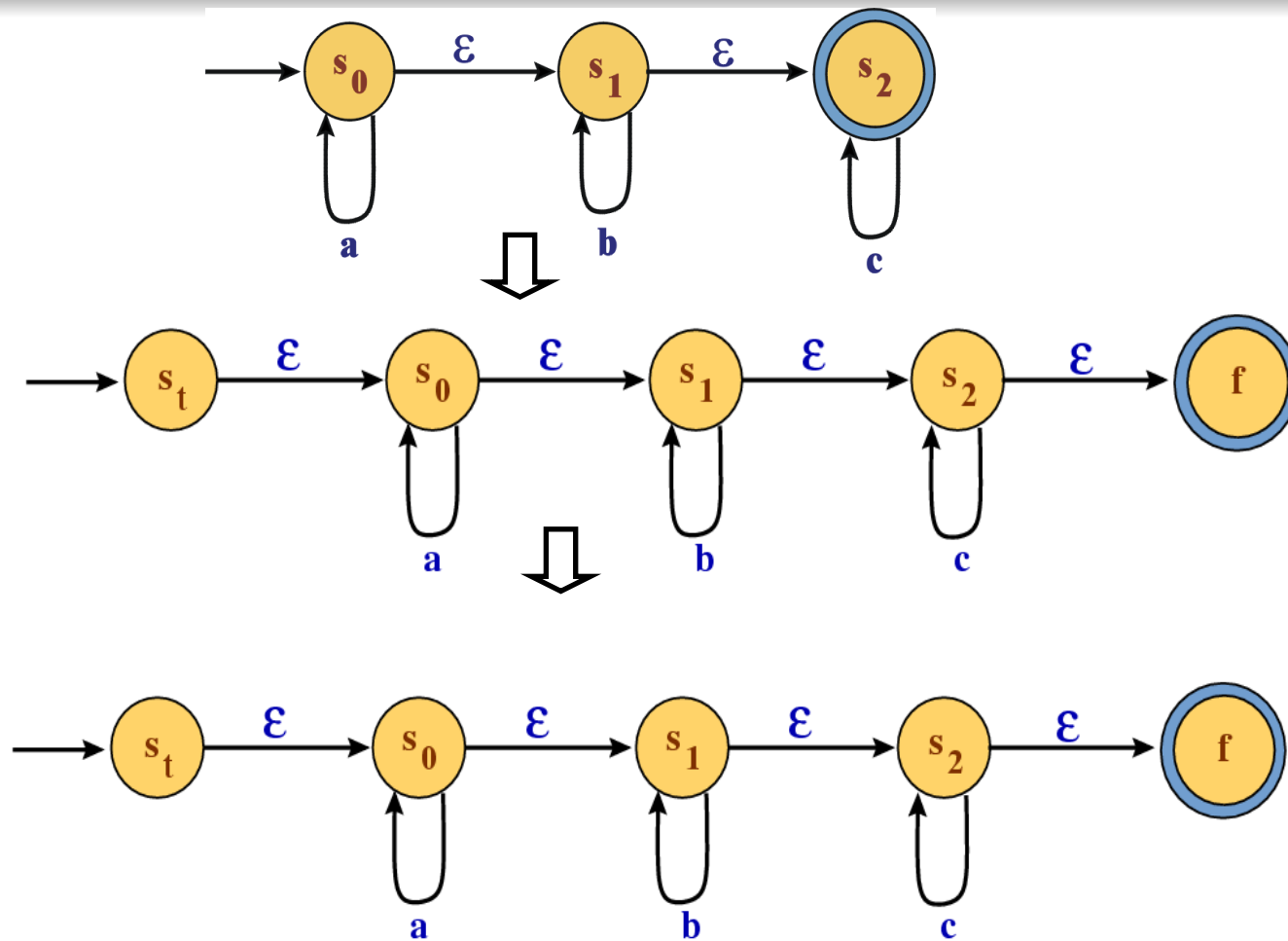
Ein neu eingefügter Endzustand f (der nur das Ziel von ϵ -Transitionen ist) wird wieder entfernt und in die Endzustandsmenge F' werden alle die Zustände aufgenommen, von denen aus f per ϵ -Transitionen erreichbar war.

$$F' = \{t \in S; (t, \epsilon) \vdash^* (f, \epsilon)\}$$

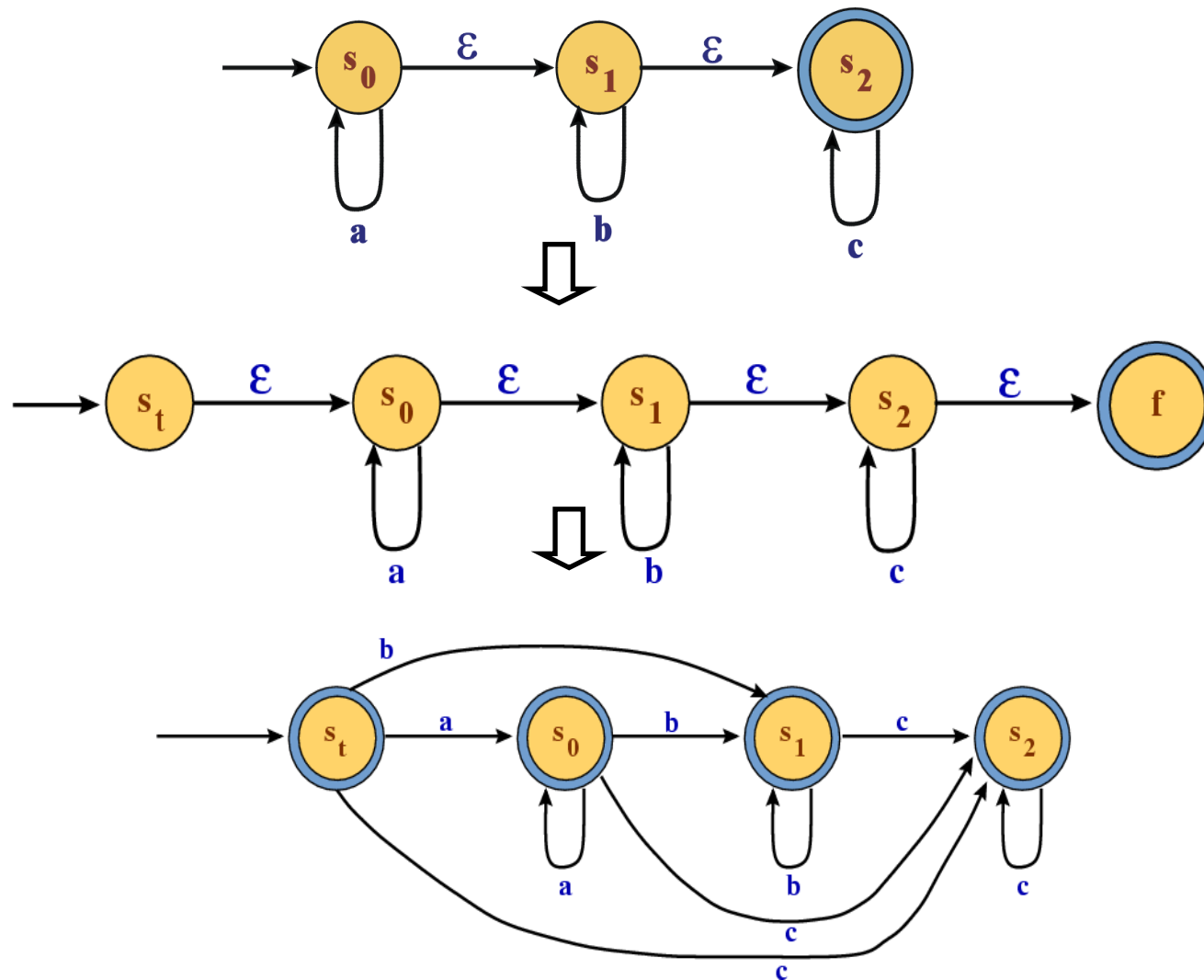
Anschließend können alle ϵ -Transitionen entfernt werden.

Da alle Transformationsschritte äquivalenzerhaltend waren, akzeptiert der so konstruierte Automat dieselbe Sprache wie der gegebene ϵ FA A .

Beispiel: Transformation von $A_{\epsilon abc}$ in einen NFA



Beispiel: Transformation von $A_{\epsilon abc}$ in einen NFA

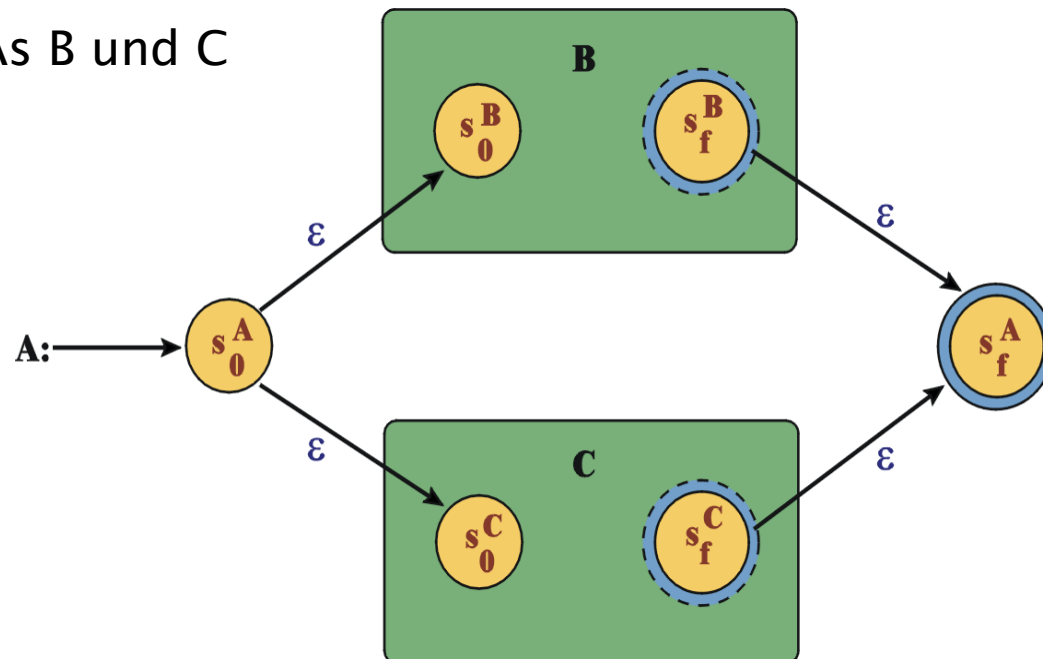


Abgeschlossenheitseigenschaften

Satz: Die Klasse der regulären Sprachen ist abgeschlossen gegenüber den Operationen **Vereinigung** \cup , **Verkettung** \cdot , **Kleene-Stern** $*$.

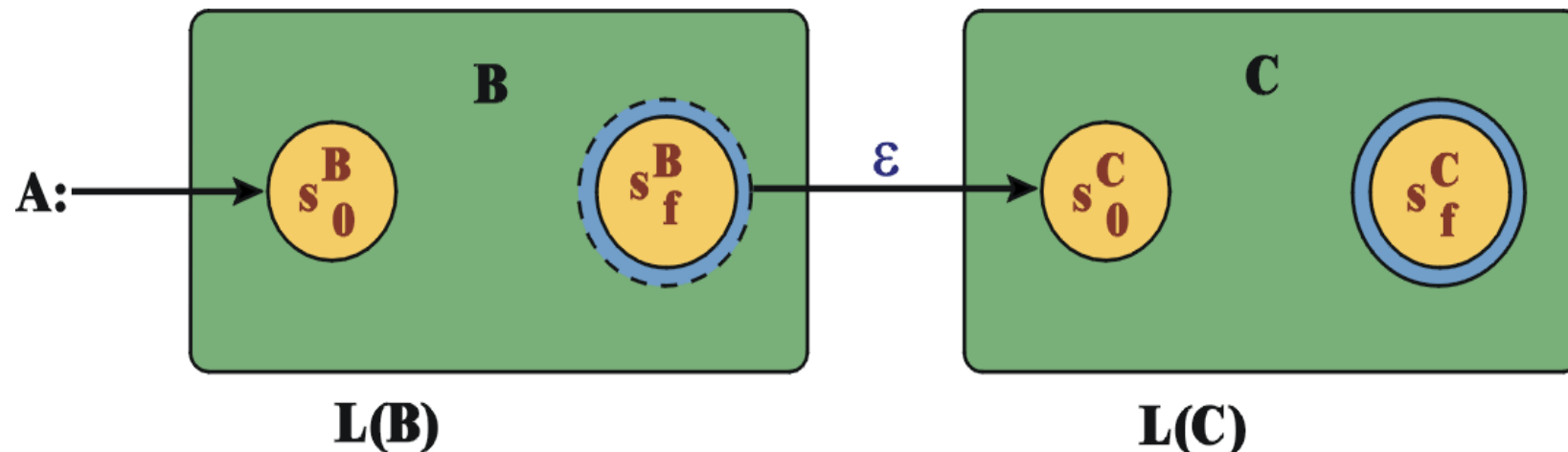
Bew.: Abgeschlossenheit gegenüber booleschen Operationen wurde schon bewiesen! Neuer, einfacherer Beweis für Abgeschlossenheit gegen \cup :

Konstruiere zu zwei DFAs B und C den ϵ FA wie folgt:



Verkettung

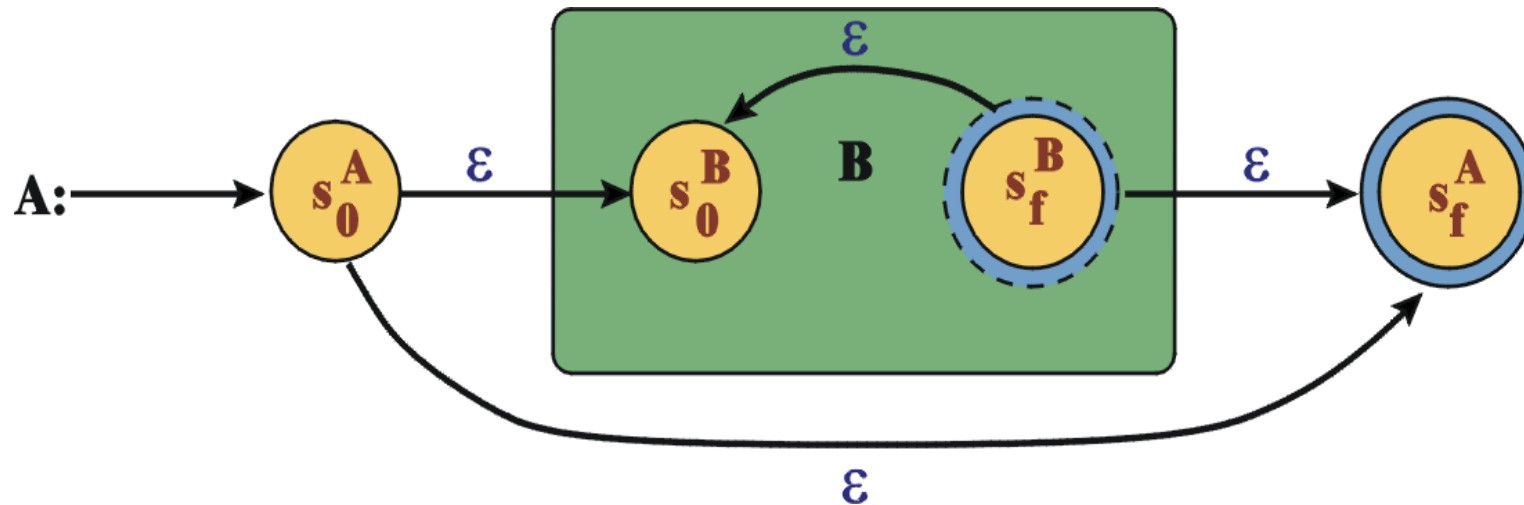
Seien zwei DFAs B und C gegeben, die die Sprachen $L(B)$ und $L(C)$ akzeptieren.
Konstruiere einen ϵ FA A wie folgt:



18 Dann ist $L(A) = L(B) L(C)$

Kleene Stern

Sei ein DFA B gegeben, der die Menge $L(B)$ akzeptiert.
Dann konstruiere den ϵ FA A wie folgt:



Offenbar ist $L(A) = (L(B))^*$

Bemerkung

Im Nachweis der Abgeschlossenheit regulärer Mengen gegenüber \cup , \cdot , $*$ steckt implizit die Potenzmengenkonstruktion zur Umwandlung von NFAs in DFAs. Daher können die konstruierten DFAs u.U. sehr viele Zustände haben!

Minimierungsproblem: Konstruiere zu einem gegebenen DFA A einen äquivalenten DFA A' mit minimaler Zustandszahl!

Reduzierter Automat

Sei $A = (\Sigma, S, \delta, s_0, F)$ ein DFA. Dann heißen zwei Zustände $s, s' \in S$ **äquivalent** bzgl. A , $s \sim_A s'$, wenn für alle Worte $w \in \Sigma^*$ gilt:

$$\delta(s, w) \in F \text{ gdw. } \delta(s', w) \in F$$

Der Automat A heißt **reduziert (oder zustands-minimal)**, wenn für alle Zustände s, s' aus $s \sim_A s'$ schon $s = s'$ folgt und wenn alle Zustände aus S vom Anfangszustand s_0 aus **erreichbar** sind. (D.h.: Für jedes $s \in S$ gibt es ein Wort $w \in \Sigma^*$, so dass $s = \delta(s_0, w)$ ist.)

“ \sim_A “ ist eine Äquivalenzrelation auf S

Minimaler endlicher Automat

Satz: Gegeben sei eine reguläre Sprache $L \subseteq \Sigma^*$. Dann ist der (Zustands-) minimale Automat A_{\min} , für den $L = L(A_{\min})$ gilt, bis auf Isomorphie eindeutig bestimmt.

Dabei sind zwei Automaten $A_1 = (\Sigma, S_1, \delta_1, s_1, F_1)$ und $A_2 = (\Sigma, S_2, \delta_2, s_2, F_2)$ genau dann isomorph, wenn $|S_1| = |S_2|$ ist und es eine bijektive Abbildung $f: S_1 \rightarrow S_2$ gibt mit

$f(\delta_1(s, a)) = \delta_2(f(s), a)$, für alle $s \in S_1$
und alle $a \in \Sigma$.

$$f(F_1) = F_2$$

Konstruktion des Minimalautomaten

Sei $A_1 = (S, \Sigma, \delta, s_0, F)$ ein DFA, der $L = L(A)$ erkennt.

Zu A wird in mehreren Schritten ein äquivalenter Automat A_{\min} konstruiert:

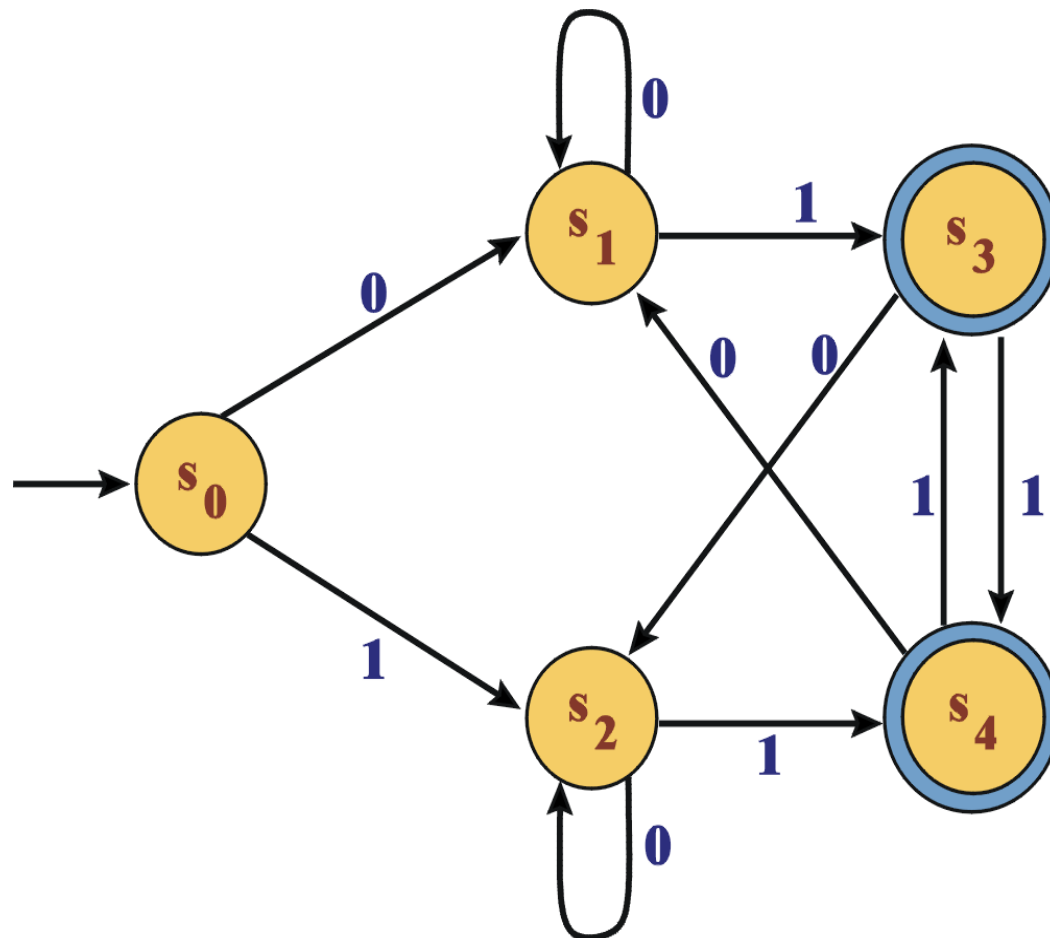
1. Vereinfache A , so dass alle Zustände von s_0 aus erreichbar sind.
2. Zerlege die Zustandsmenge disjunkt in zwei Teile: $\pi_1 = \{F, S - F\}$
3. Verfeinere die aktuelle Zerlegung $\pi_i = \{S_1, \dots, S_k\}$: In der neuen Zerlegung π_{i+1} gehören Zustände s, s' genau dann zur gleichen Menge, wenn $s \in S_i$ und $s' \in S_i$ sowie $\delta(s, a) \in S_j$ und $\delta(s', a) \in S_j$ für alle $a \in \Sigma$ und $i, j \in \{1, \dots, k\}$.

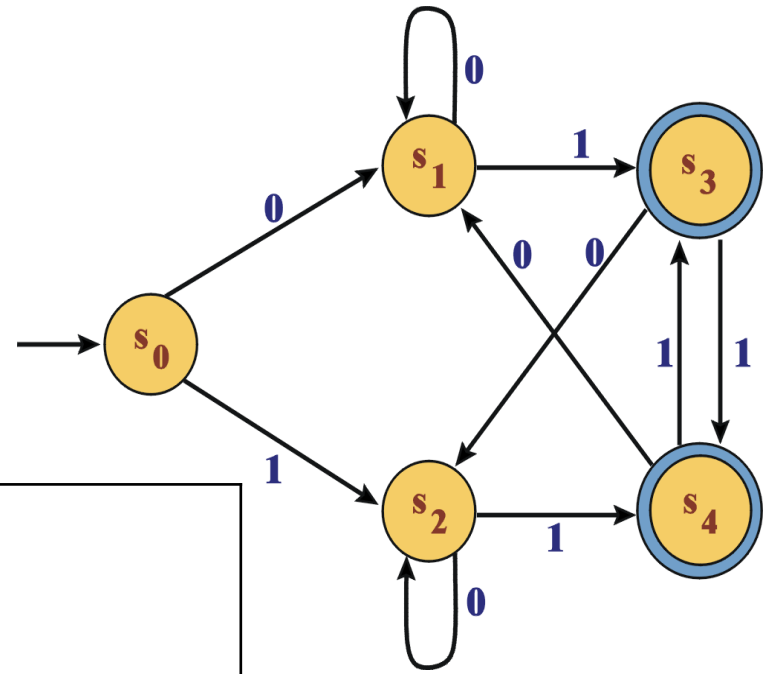
Aufgeteilt werden muss S_i , wenn für $s, s' \in S_i$ gilt:

$$\delta(s, a) \neq \delta(s', a)$$

4. Ergab die letzte Verfeinerung mehr Mengen, gehe zurück zu 3; sonst sind die Mengen der letzten Zerlegung die Zustände von A_{\min} .

Ein zu minimierender DFA





π_1	M_1	M_2
	s_3 s_4	s_0 s_2 s_1

Reduzierter DFA

