Vorlesung

Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I

Bernhard Beckert

Institut für Informatik

UNIVERSITÄT KOBLENZ · LANDAU

Sommersemester 2007

Dank

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

– Bernhard Beckert, April 2007

Inhalt von Teil III

- Vereinfachtes Modell eines Computers: **endlicher Automat**
- Die von endlichen Automaten erkannten **„rationalen“** Sprachen sind genau die Typ-3-Sprachen (**rechtslinear**, **regulär**)
- **Determinierte** und **indeterminierte** endliche Automaten sind äquivalent
- **Pumping Lemma** erlaubt, eine Sprache als nicht rational nachzuweisen.
- Es gibt Algorithmen, die **Probleme über endlichen Automaten** bzw. Typ-3-Sprachen lösen.
- Typ-3-Sprachen sind genau die, die durch **reguläre Ausdrücke** beschrieben werden können.

Teil III

Endliche Automaten

Beispiel

Beispiel 11.1

Die Sprache

$$L = \{aa\}\{ab\}^*\{c\}$$

ist regulär.

Denn sie wird (z. B.) erzeugt von der rechtslinearen Grammatik

$$G = (\{S, A\}, \{a, b, c\}, R, S),$$

mit Regelmenge R:

$$S \to aaA$$
$$A \to abA \mid c$$

Beispiel

Beispiel 11.2

Die Sprache aller durch 3 teilbaren Dezimalzahlen ist regulär.

Eine erzeugende Grammatik ist

$$G = (\{S, S_0, S_1, S_2\}, \{0, \ldots, 9\}, R, S)$$

mit der Regelmenge R:

$$\begin{aligned}
S &\to 3S_0 \mid 6S_0 \mid 9S_0 \mid 1S_1 \mid 4S_1 \mid 7S_1 \mid 2S_2 \mid 5S_2 \mid 8S_2 \mid 0 \\
S_0 &\to 0S_0 \mid 3S_0 \mid 6S_0 \mid 9S_0 \mid 1S_1 \mid 4S_1 \mid 7S_1 \mid 2S_2 \mid 5S_2 \mid 8S_2 \mid \varepsilon \\
S_1 &\to 0S_1 \mid 3S_1 \mid 6S_1 \mid 9S_1 \mid 1S_2 \mid 4S_2 \mid 7S_2 \mid 2S_0 \mid 5S_0 \mid 8S_0 \\
S_2 &\to 0S_2 \mid 3S_2 \mid 6S_2 \mid 9S_2 \mid 1S_0 \mid 4S_0 \mid 7S_0 \mid 2S_1 \mid 5S_1 \mid 8S_1
\end{aligned}$$

Ohne das ε in der zweiten Regel wäre nur die “0” als Terminalwort herleitbar.

Grammatik vs. Automat

Grammatik: erzeugt Wörter

Automat: analysiert / erkennt Wörter

beide: beschreiben / definieren eine Sprachen

Endlicher Automat: Informell

Endlicher Automat

- Ein endlicher Automat testet, ob ein gegebenes $w \in \Sigma^*$ in einer Sprache L liegt.
- **Lesekopf** erlaubt w zu lesen.
  Bewegt sich nur von links nach rechts.
- Endlich viele mögliche **interne Zustände**,
  immer einer davon ist der aktuelle Zustand
- Automat beginnt in einem **initialen Zustand**.
- Bei jedem gelesenen Buchstaben Übergang zu neuem aktuellen Zustand, in Abhängigkeit vom Buchstaben und dem alten Zustand
- Wenn am Ende von w ein **finaler Zustand** erreicht ist,
  ist w **akzeptiert** als Element von L,
  sonst nicht.
- Automat **stoppt** (auf jeden Fall) nach $|w|$ Schritten

Endlicher Automat: Modell eines einfachen Computers

Endlicher Automat: Computer mit begrenztem Speicher

- Kann vom Band nur lesen
  $\Rightarrow$ kein externer Speicher
- Speichert nur den aktuellen Zustand ($\approx$ Programmzähler)
  $\Rightarrow$ stark begrenzter interner Speicher

Endlicher Automat: Darstellung als Graph

Darstellung als Graph

- ein **Knoten** für jeden **Zustand**
- **Kanten** beschreiben **Zustandsänderungen**,
  sind mit Buchstaben beschriftet
- **initiale** Zustände sind mit einem **Pfeil** gekennzeichnet
- **finale Zustände** mit einem **doppelten Kreis**

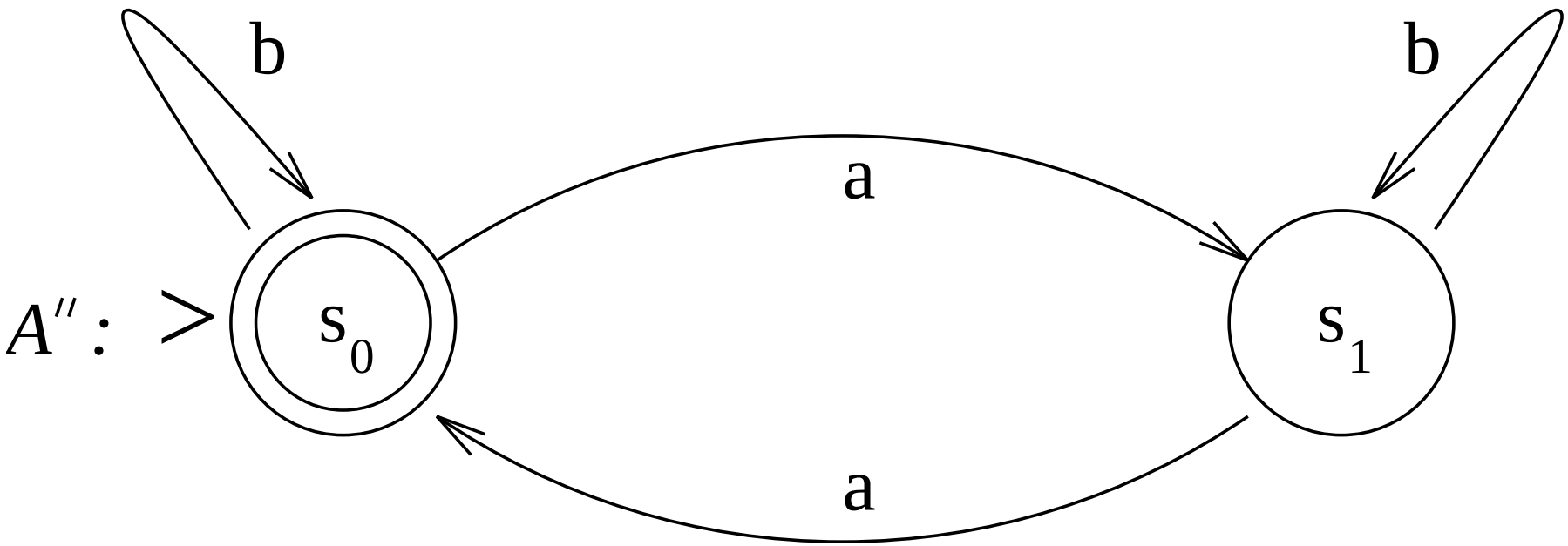
Endlicher Automat: Darstellung als Graph

Beispiel 11.3 (Sprache $\{w \mid \#_a(w)$ *gerade*$\} \subset \{a,b\}^*$)

Der folgende endliche Automat erkennt die Sprache

$$\{w \mid \#_a(w) \text{ gerade}\} \qquad \text{über} \qquad \Sigma = \{a,b\}$$

der Wörter mit gerader Anzahl von „a“s



Endlicher Automat: Definition

Definition 11.4 (Endlicher Automat)

Ein **endlicher Automat (e.a., finite automaton** ist ein Tupel

$$\mathcal{A} = (K, \Sigma, \delta, s_0, F)$$

Dabei ist

- K eine endliche Menge von **Zuständen**
- Σ ein **endliches Alphabet**
  (aus dessen Buchstaben die Eingabewörter bestehen können)
- $\delta : K \times \Sigma \to K$ die totale(!) **Übergangsfunktion**
- $s_0 \in K$ der **Startzustand**
- $F \subseteq K$ die Menge der **finalen Zustände**

Endlicher Automat: Übergangsfunktion

Bedeutung der Übergangsfunktion

$$\delta(q, a) = q'$$

bedeutet:

- Wenn der Automat im Zustand q ist
- und ein a liest,
- dann geht in den Zustand q' über.

Endlicher Automat: Übergangsfunktion

Definition 11.5 (Erweiterung von δ zu δ^*)

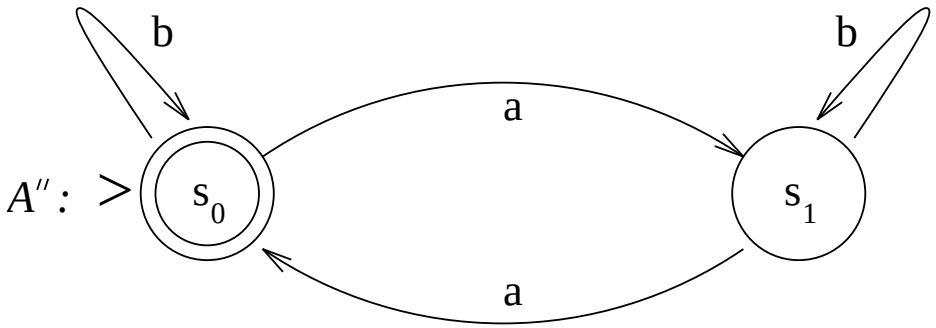
$$\delta^* : K \times \Sigma^* \to K$$

ist strukturell rekursiv über Σ^* definiert durch:

$$\delta^*(q, \varepsilon) := q$$
$$\delta^*(q, wa) := \delta(\delta^*(q, w), a)$$

Endlicher Automat: Beispiel

Beispiel 11.6



Dieser Automat azeptiert die Sprache

$$\{w \mid \#_a(w) \text{ gerade}\} \subset \{a,b\}^*$$

(s. Bsp. 11.3).

Formal hat er die Form:

$$\mathcal{A} = (\{s_0, s_1\}, \{a,b\}, \delta, s_0, \{s_0\})$$

mit

$$\delta(s_0, a) = s_1 \qquad \delta(s_1, a) = s_0$$
$$\delta(s_0, b) = s_0 \qquad \delta(s_1, b) = s_1$$

Endlicher Automat: Übergangsfunktion

Beispiel 11.7 (Beispiel für δ^*)

$$
\begin{aligned}
\delta^*(s_0, aab) &= \delta\big(\delta^*(s_0, aa), b\big) \\
&= \delta\big(\delta\big(\delta^*(s_0, a), a\big), b\big) \\
&= \delta\big(\delta\big(\delta(\delta^*(s_0, \varepsilon), a\big), a\big), b\big) \\
&= \delta\big(\delta\big(\delta(s_0, a), a\big), b\big) \\
&= \delta\big(\delta(s_1, a), b\big) \\
&= \delta(s_0, b) \\
&= s_0
\end{aligned}
$$

Endlicher Automat: Akzeptierte Sprache

Definition 11.8 (Von einem endlichen Automaten akzeptierte Sprache)

Die von einem Automaten $\mathcal{A}$ **akzeptierte** Sprache, ist definiert als

$$L(\mathcal{A}) := \{w \in \Sigma^* \mid \delta^*(s_0, w) \in F\}$$

Definition 11.9 (Von endlichen Automaten akzeptierte Sprachen)

Die Menge

$$\textbf{RAT} := \{L \mid \text{es gibt einen endlichen Automaten } \mathcal{A} \text{ mit } L = L(\mathcal{A})\}$$

der von endlichen Automaten akzeptierten Sprachen
heißt Menge der **rationalen** Sprachen

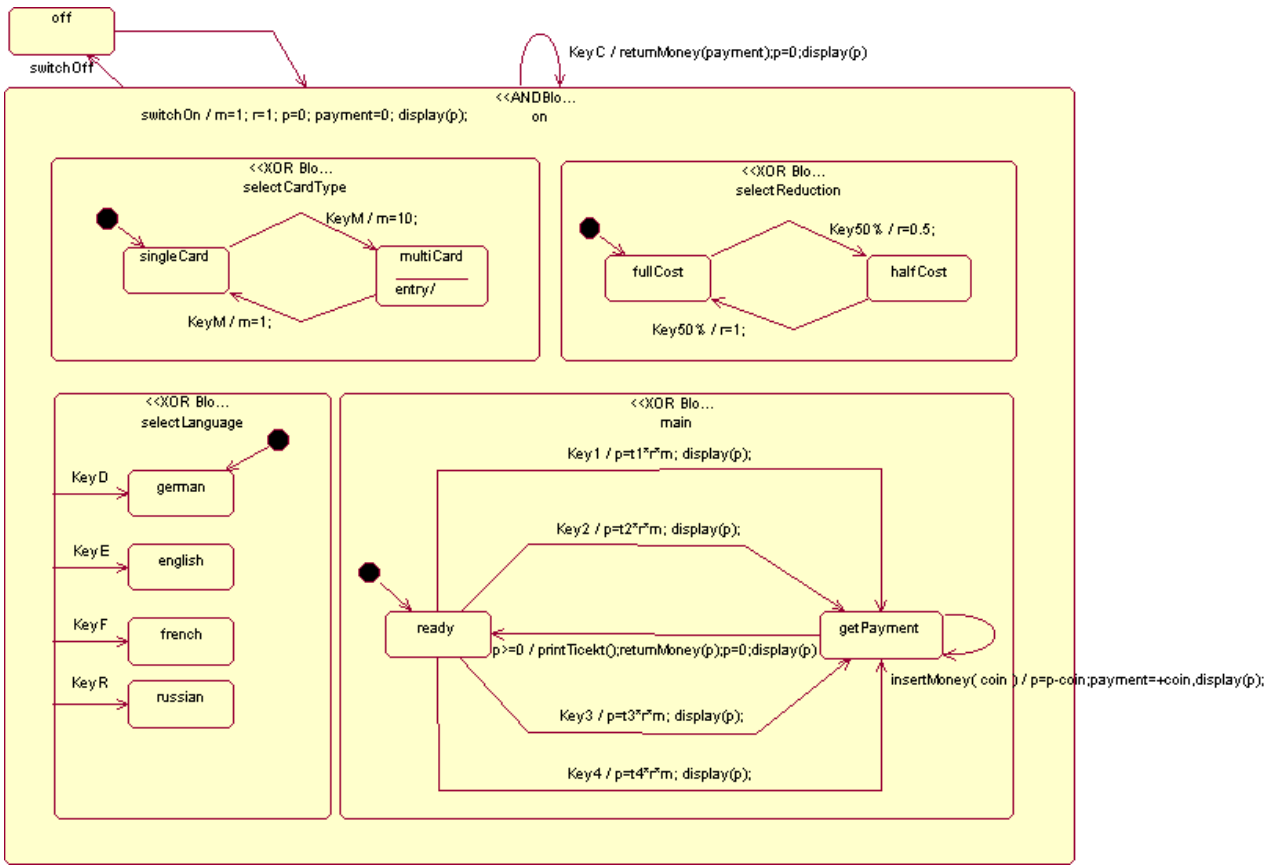
Wir zeigen demnächst:
RAT = Menge der regulären Sprachen

Endliche Automaten: UML State Charts

UML State Charts

UML State Charts sind eine (erweiterte) Form endlicher Automaten

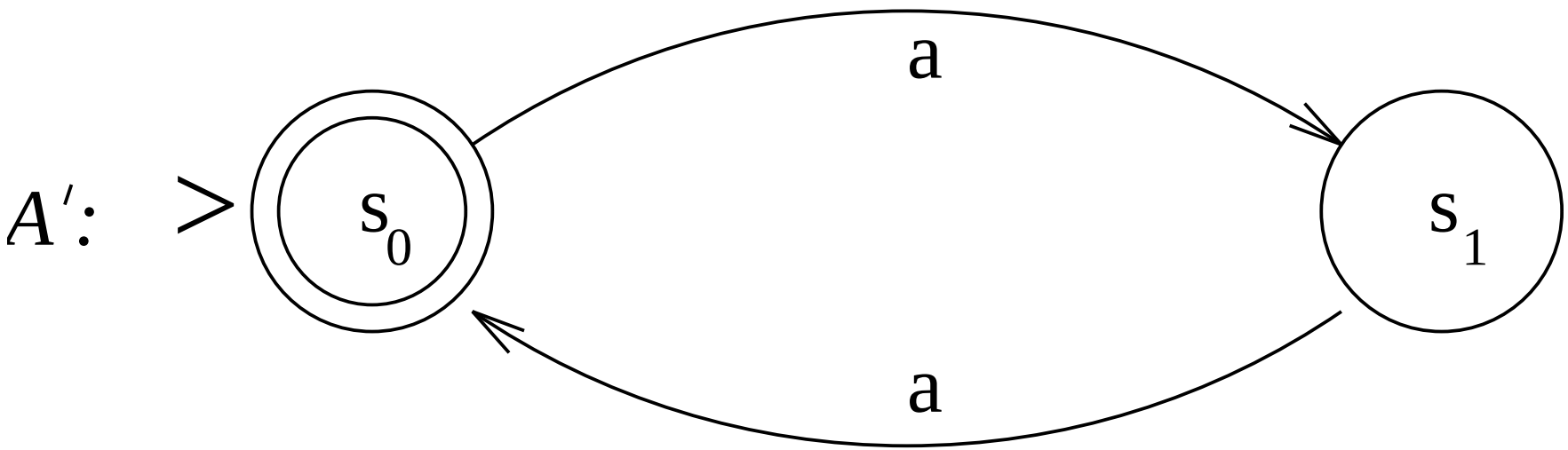
Beispiel 11.10



Endliche Automaten: Weitere Beispiele

Beispiel 11.11

Die Sprache aller Wörter mit gerader Anzahl von a
über dem (kleineren) Alphabet $\Sigma = \{a\}$
wird akzeptiert von:



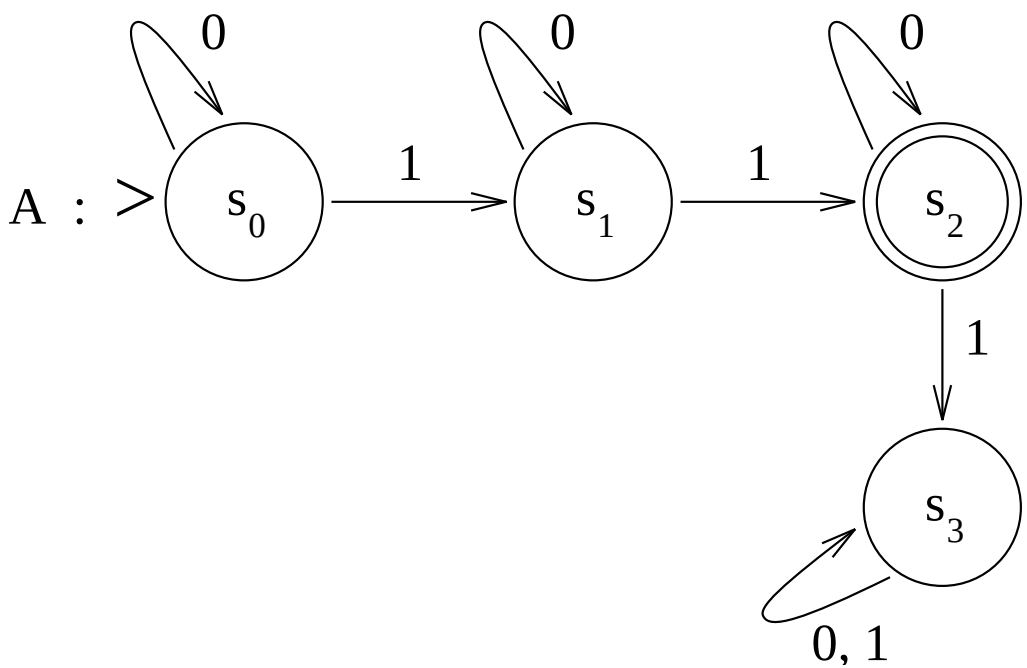
Endliche Automaten: Weitere Beispiele

Beispiel 11.12

Die Sprache

$$L = \{w \in \{0,1\}^* \mid w \text{ enthält genau zwei Einsen}\}$$

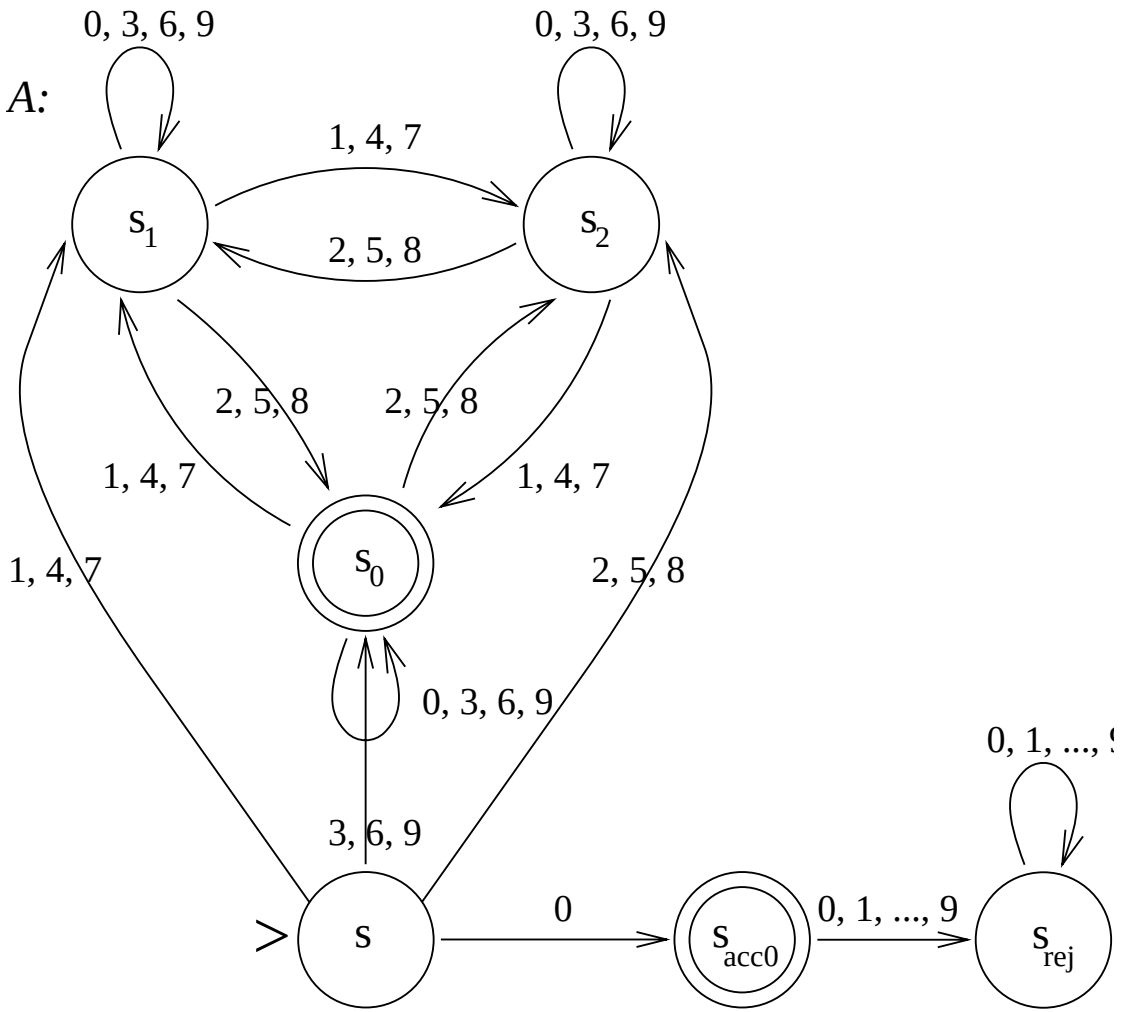
wird akzeptiert von dem folgendenendlichen Automaten:



Endliche Automaten: Weitere Beispiele

Beispiel 11.13

Die Sprache aller durch 3 teilbaren Dezimalzahlen wird akzeptiert durch:



Teil III

Endliche Automaten

Determiniert / indeterminiert

Determinierter endliche Automat

- Für einen Zustand q und eine Eingabe a
  genau ein einziger Nachfolgezustand
- festgelegt durch Übergangs**funktion** δ

Indeterminierter endlicher Automat

- Für einen Zustand q und eine Eingabe a
  evtl. **mehrere Nachfolgezustände** – oder **gar keiner**
- festgelegt durch Übergangs**relation** Δ

Indeterminierter endlicher Automat

Definition 12.1 (Indeterminierter endlicher Automat)

Ein **indeterminierter** endlicher Automat (NDEA) ist ein Tupel

$$\mathcal{A} = (K, \Sigma, \Delta, I, F)$$

Dabei ist

- K eine endliche Menge von Zuständen,
- Σ ein endliches Alphabet,
- $\Delta \subseteq (K \times \Sigma) \times K$ eine Übergangs**relation**,
- $I \subseteq K$ eine **Menge von Startzuständen**,
- $F \subseteq K$ eine Menge von finalen Zuständen.

Indeterminierter endlicher Automat: Übergangsrelation

Definition 12.2 (Erweiterung von Δ zu Δ^*)

$$\Delta^* \subseteq (K \times \Sigma^*) \times K$$

ist definiert durch:

$$\Delta^*(\,(q,\varepsilon),\,q'\,) \quad \underline{\text{gdw}} \quad q' = q$$

$$\Delta^*(\,(q,wa),\,q'\,) \quad \underline{\text{gdw}} \quad \exists q'' \in K \left(\Delta^*(\,(q,w),\,q''\,) \wedge \Delta(\,(q'',a),\,q'\,)\right)$$

NDEA: Akzeptierte Sprache

Wann akzeptiert ein indeterminierter Automat ein Wort?

Ein indeterminierter endlicher Automat $\mathcal{A}$ akzeptiert ein Wort w, wenn

- es **mindestens einen** Weg mit der **Beschriftung** w durch $\mathcal{A}$ gibt,
- der in einem **finalen** Zustand endet.

Definition 12.3 (Von einem NDEA akzeptierte Sprache)

Die von einem indeterminierten endlichen Automaten $\mathcal{A}$ **akzeptierte** Sprache ist

$$L(\mathcal{A}) \quad := \quad \{w \in \Sigma^* \mid \exists s_0 \in I \, \exists q \in F \, \Delta^*(\,(s_0, w),\, q\,)\}$$

NDEA: Beispiel

Beispiel 12.4

Der Automat

$$\mathcal{A} = (\{S_0, S_1, S_2\}, \{a, b\}, \Delta, \{S_0\}, \{S_0\})$$

mit

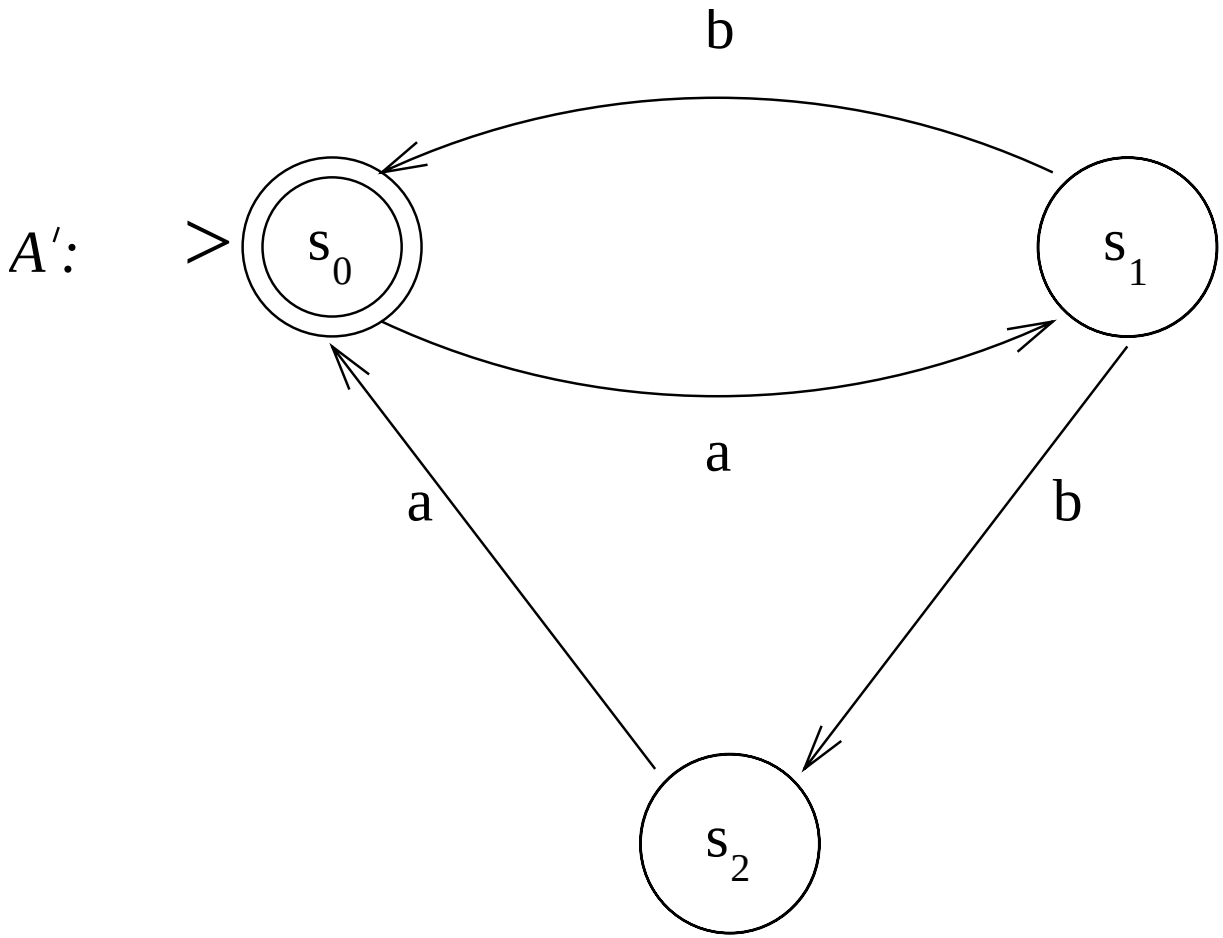
$$\Delta(S_0, a) = \{S_1\}$$
$$\Delta(S_1, b) = \{S_0, S_2\}$$
$$\Delta(S_2, a) = \{S_0\}$$

akzeptiert die Sprache

$$L = \{ab, aba\}^*$$

NDEA: Graphische Darstellung

Der indeterminierte Automat aus Beispiel 12.4



Akzeptiert: $\{ab, aba\}^*$

Indeterminierter endlicher Automat

Vom indeterminierten Automaten zum Algorithmus?

Vom Automaten zum Algorithmus (für das Wortproblem):

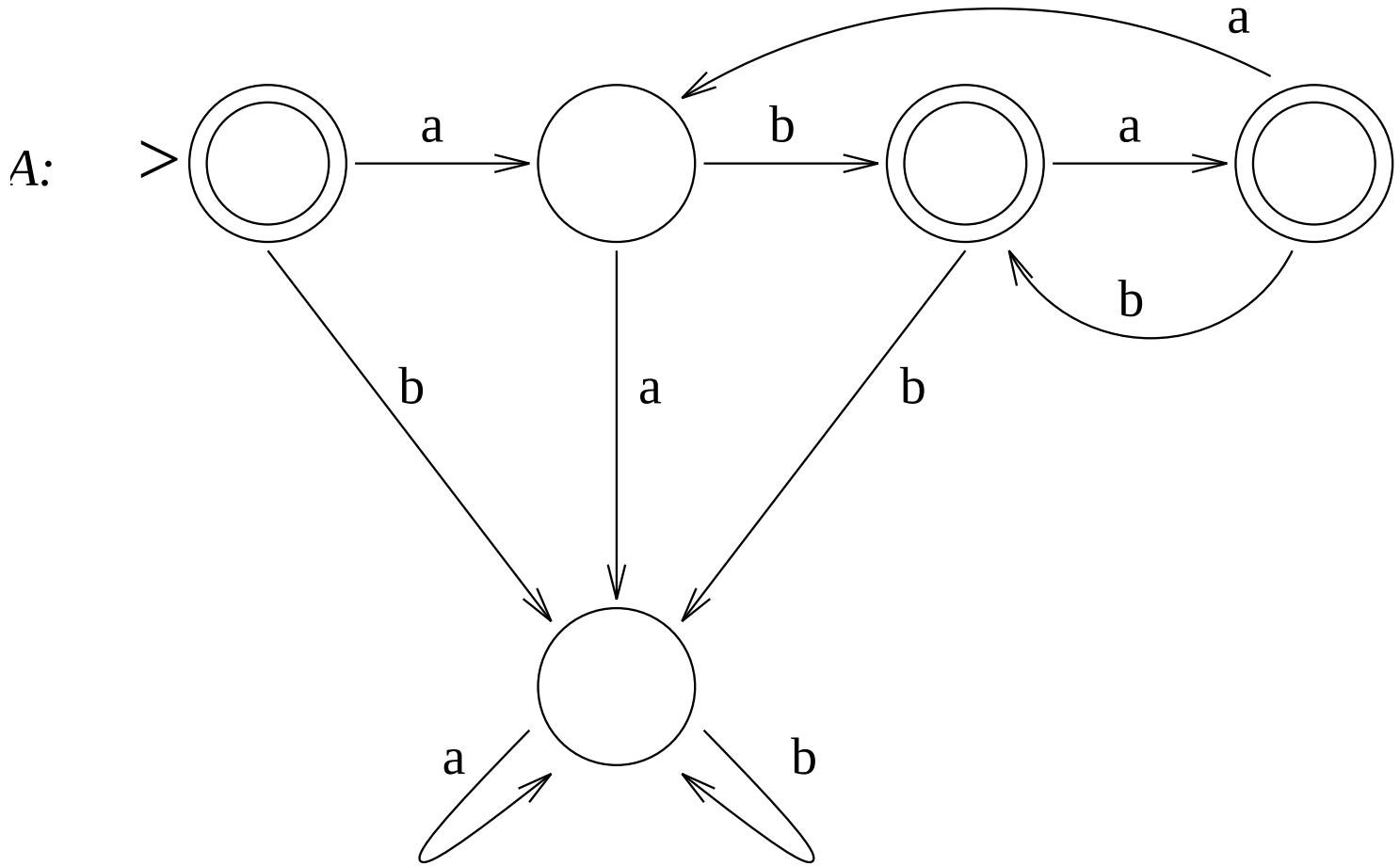
- **DEA** = **Algorithmus**
- **NDEA** + **Suchstrategie** = **Algorithmus**

Zwei Sichtweisen auf indeterminierte Automaten

- Der Automat durchläuft **alle** Wege
  (**parallel** oder mittels **Backtracking**)
- Der Automat **rät**, welcher von mehreren möglichen Folgezuständen der richtige ist

NDEA und DEA: Beispiel

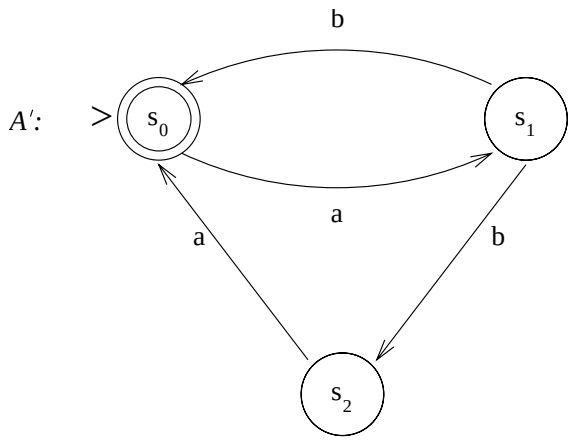
Beispiel 12.5 (DEA für gleiche Sprache wie NDEA aus Bsp. 12.4)



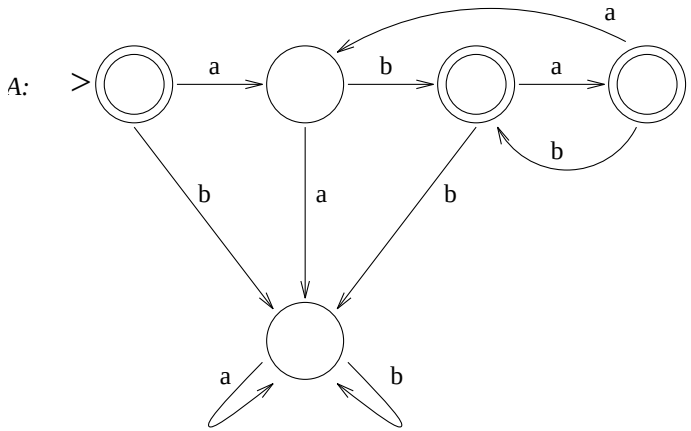
Akzeptiert: $\{ab, aba\}^*$

NDEA und DEA

Vergleich NDEA / DEA



NDEA:



DEA:

- DEA hat mehr Zustände, komplizierter
- DEA muss nicht „raten“
- DEA braucht genauso viele Schritte

NDEA und DEA

Wir zeigen später:

Für jeden indeterminierten Automaten A_{NDEA}
gibt es einen determinierten Automaten A_{DEA} mit

$$L(A_{\mathrm{NDEA}}) = L(A_{\mathrm{DEA}})$$

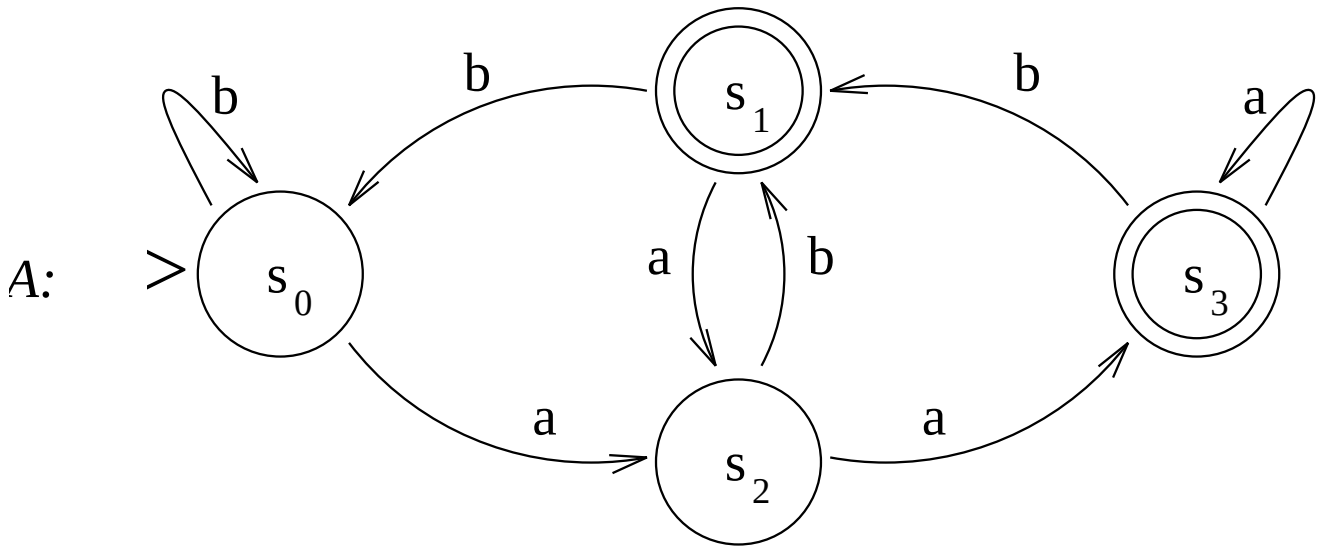
NDEA und DEA: Weiteres Beispiel

Beispiel 12.6

Determinierter Automat für die Sprache

$$L = \{a,b\}^*\{a\}\{a,b\}$$

(die Sprache aller Wörter über $\{a,b\}$, deren zweitletzter Buchstabe ein a ist)



Idee: Im Zustand jeweils die letzten zwei Buchstaben merken

NDEA und DEA: Weiteres Beispiel

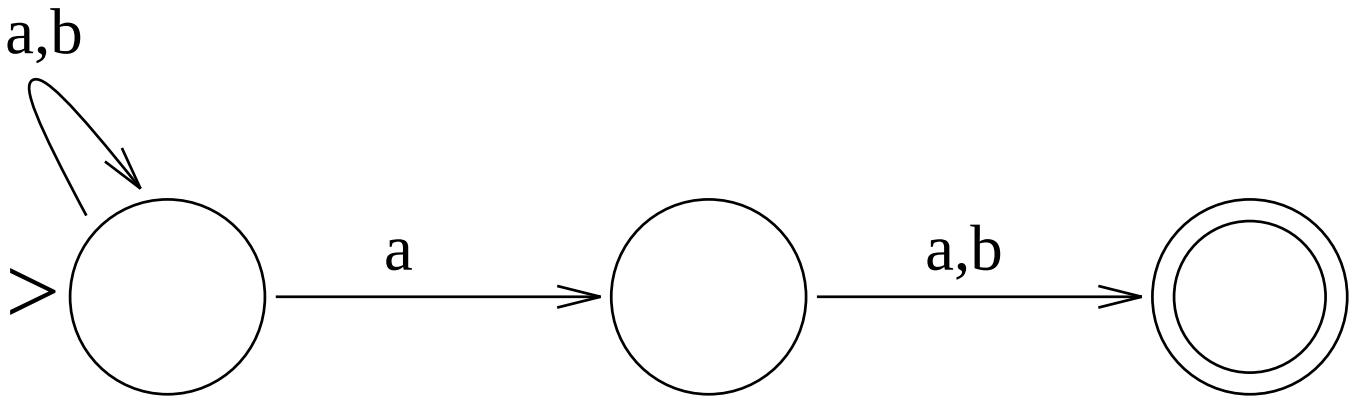
Beispiel 12.7

Indeterminierter Automat für die Sprache

$$L = \{a,b\}^*\{a\}\{a,b\}$$

(die Sprache aller Wörter über $\{a,b\}$, deren zweitletzter Buchstabe ein a ist)

A':



NDEA und DEA: Größenvergleich

Größenvergleich (Worst case)

Sprache über $\{a, b\}$ der Wörter, deren nt-letzter Buchstabe ein a ist

Determinierter Automat: $\mathbf{2^n}$ Zustände

(einen für jede Buchstabenkombination der Länge n)

Indeterminierter Automat: $\mathbf{n+1}$ Zustände

Gleichmächtigkeit von DEA und NDEA

Theorem 12.8 (DEA gleich mächtig wie NDEA)

Eine Sprache ist rational
(es gibt einen determinierten endlichen Automaten, der sie akzeptiert)

gdw

es gibt einen indeterminierten endlichen Automaten, der sie akzeptiert.

Beweis.

„$\Rightarrow$“:

- Sei L eine rationale Sprache.
- Dann gibt es laut Definition einen determinierten endlichen Automaten $\mathcal{A}_{\mathrm{DEA}}$ mit $L = L(\mathcal{A}_{\mathrm{DEA}})$.
- Jeder determinierte endliche Automat ist aber insbesondere auch ein (besonderer) indeterminierter endlicher Automat.

□

Gleichmächtigkeit von DEA und NDEA

Beweis (Forsetzung)

„$\Leftarrow$“:

Sei

$$\mathcal{A}_{\mathrm{NDEA}} = (K, \Sigma, \Delta, I, F)$$

ein (beliebiger) indeterminierter endlicher Automat.
Er akzeptiert die Sprache $L(\mathcal{A}_{\mathrm{NDEA}})$.

Beweisidee:
Konstruiere aus $\mathcal{A}_{\mathrm{NDEA}}$ einen determinierten Automaten $\mathcal{A}_{\mathrm{DEA}}$ mit

$$L(\mathcal{A}_{\mathrm{NDEA}}) = L(\mathcal{A}_{\mathrm{DEA}})$$

mit Hilfe einer Potenzmengenkonstruktion ...

Gleichmächtigkeit von DEA und NDEA

Beweis (Forsetzung)

Fortsetzung ...

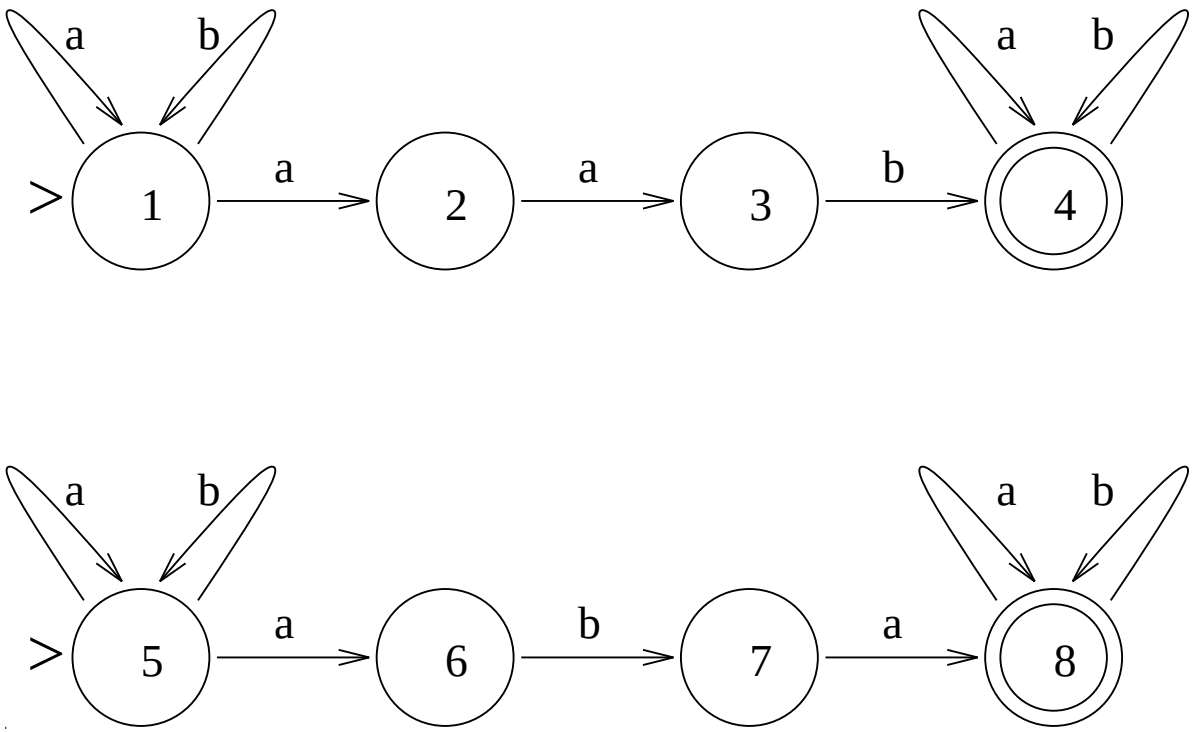
- Zustände in $\mathcal{A}_{\mathrm{DEA}}$ bestehen aus Mengen von Zuständen von $\mathcal{A}_{\mathrm{NDEA}}$
- Wenn man in $\mathcal{A}_{\mathrm{NDEA}}$ mit w nach $q_1, \ldots, q_n$ gelangt,
  dann gelangt man in $\mathcal{A}_{\mathrm{DEA}}$ mit w nach $q' = \{q_1, \ldots, q_n\}$.
- Initialer Zustand von $\mathcal{A}_{\mathrm{DEA}}$:
  Menge aller initialen Zustände von $\mathcal{A}_{\mathrm{NDEA}}$
- Finale Zustände von $\mathcal{A}_{\mathrm{DEA}}$:
  Jede Menge von Zustände, die einen finalen Zustand von $\mathcal{A}_{\mathrm{NDEA}}$ enthält

Gleichmächtigkeit von DEA und NDEA

Zunächst ein konkretes Beispiel

Ein NDEA, der die folgende Sprache akzeptiert:

$$L_{aab/aba} = \{w \in \{a,b\}^* \mid w \text{ hat } aab \text{ oder } aba \text{ als Teilwort}\}$$



Gleichmächtigkeit von DEA und NDEA

Zunächst ein konkretes Beispiel

Startzustand:

Menge der alten Startzustände, also $\{1,5\}$.

Nächster Schritt:

Übergang von $\{1,5\}$ mit a

$$\Delta(1,a) = \{1,2\}$$
$$\Delta(5,a) = \{5,6\}$$

Also, neuer Zustand $\{1,2,5,6\}$ mit

$$\delta_{\mathcal{A}_{\mathrm{DEA}}}(\{1,5\},a) = \{1,2,5,6\}$$

Gleichmächtigkeit von DEA und NDEA

Zunächst ein konkretes Beispiel

Nächster Schritt:

Übergang von $\{1,5\}$ mit b.

$$\Delta(1,b) = \{1\}$$
$$\Delta(5,b) = \{5\}$$

Für den Eingabebuchstaben b bleibt $\mathcal{A}_{\mathrm{DEA}}$ also im Startzustand.

Gleichmächtigkeit von DEA und NDEA

Zunächst ein konkretes Beispiel

Nächster Schritt:

Übergang von $\{1,2,5,6\}$ mit a

$$\Delta(1,a) = \{1,2\}$$
$$\Delta(2,a) = \{3\}$$
$$\Delta(5,a) = \{5,6\}$$
$$\Delta(6,a) = \emptyset$$

Also, neuer Zustand $\{1,2,3,5,6\}$ mit

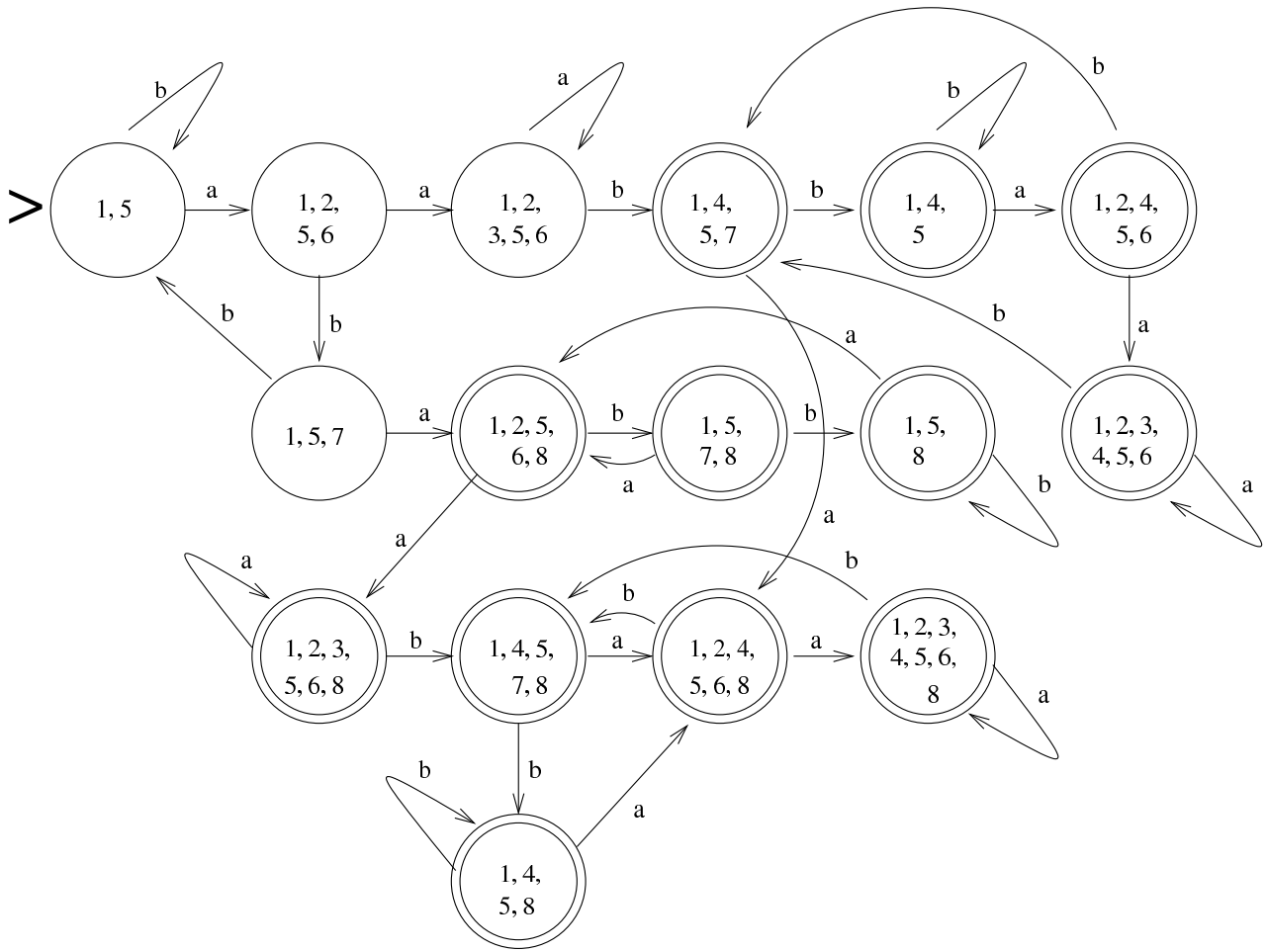
$$\delta_{\mathcal{A}_{\mathrm{DEA}}}(\{1,2,5,6\},a) = \{1,2,3,5,6\}$$

usw.

Gleichmächtigkeit von DEA und NDEA

Zunächst ein konkretes Beispiel

Es ergibt sich folgender determinierter Automat $\mathcal{A}_{\mathrm{DEA}}$:



Gleichmächtigkeit von DEA und NDEA

Beweis (Fortsetzung)

Konstruktion des determinierten endlichen Automaten $\mathcal{A}_{\mathrm{DEA}}$ formal:

Gegeben: Indeterminierter endlicher Automat

$$\mathcal{A}_{\mathrm{NDEA}} = (K, \Sigma, \Delta, I, F)$$

Wir konstruieren: Determinierten endlichen Automaten

$$\mathcal{A}_{\mathrm{DEA}} = (K', \Sigma, \delta', I', F')$$

mit

- $K' = 2^K$ (die Potenzmenge von K)
- Übergangsfunktion

$$\delta' : 2^K \times \Sigma \to 2^K \qquad \text{mit} \qquad \delta'(M, a) := \bigcup_{q \in M} \Delta(q, a)$$

Gleichmächtigkeit von DEA und NDEA

Beweis (Fortsetzung)

Konstruktion des determinierten endlichen Automaten $\mathcal{A}_{\mathrm{DEA}}$ formal:

Gegeben: Indeterminierter endlicher Automa

$$\mathcal{A}_{\mathrm{NDEA}} = (K, \Sigma, \Delta, I, F)$$

Wir konstruieren: Determinierten endlichen Automaten

$$\mathcal{A}_{\mathrm{DEA}} = (K', \Sigma, \delta', I', F')$$

mit

- $I' = I$ (die Menge der initialen Zustände von $\mathcal{A}_{\mathrm{NDEA}}$)
- $F' = \{M \subset K \mid M \cap F \neq \emptyset\}$
  (alle Zustandsmengen von $\mathcal{A}_{\mathrm{NDEA}}$, die einen finalen Zustand enthalten)

Gleichmächtigkeit von DEA und NDEA

Beweis (Fortsetzung)

Konstruktion des determinierten endlichen Automaten $\mathcal{A}_{\mathrm{DEA}}$ formal:

Gegeben: Indeterminierter endlicher Automa

$$\mathcal{A}_{\mathrm{NDEA}} = (K, \Sigma, \Delta, I, F)$$

Wir konstruieren: Determinierten endlichen Automaten

$$\mathcal{A}_{\mathrm{DEA}} = (K', \Sigma, \delta', I', F')$$

Merke:

- $\emptyset \in K'$
- $\delta'(\emptyset, x) = \emptyset$ für alle $x \in \Sigma$

Gleichmächtigkeit von DEA und NDEA

Beweis (Fortsetzung)

Lemma: Es ist

$$\delta'^*(M, w) = \bigcup_{q \in M} \Delta^*(q, w)$$

Beweis durch Induktion über die Länge von w:

Induktionsanfang:

$$\delta'^*(M, \varepsilon) = M = \bigcup_{q \in M} \{q\} = \bigcup_{q \in M} \Delta^*(q, \varepsilon)$$

Gleichmächtigkeit von DEA und NDEA

Beweis (Fortsetzung)

Lemma: Es ist

$$\delta'^*(M,w) = \bigcup_{q\in M} \Delta^*(q,w)$$

Induktionsschritt:

$$\begin{aligned}
& \delta'^*(M,wa) \\
= \;& \delta'(\delta'^*(M,w),a) && \text{Def. von } ^* \\
= \;& \textstyle\bigcup_{p\in\delta'^*(M,w)} \Delta(p,a) && \text{Definition von } \delta' \\
= \;& \textstyle\bigcup_{p\in(\bigcup_{q\in M}\Delta^*(q,w))} \Delta(p,a) && \text{Ind.-Vor. für } \delta'(M,w) \\
= \;& \{q' \mid \exists q\in M\ \exists p\in\Delta^*(q,w)\ q'\in\Delta(p,a)\} \\
= \;& \{q' \mid \exists q\in M\ q'\in\Delta^*(q,wa)\} && \text{Def. von } ^* \\
= \;& \textstyle\bigcup_{q\in M}\Delta^*(q,wa)
\end{aligned}$$

Gleichmächtigkeit von DEA und NDEA

Beweis (Schluss)

Es gilt für alle $w \in \Sigma^*$:

$w \in L(\mathcal{A}_{\mathrm{DEA}})$

gdw	$\delta'^*(I', w) \in F'$	(Def. der Sprache eines Automaten)
gdw	$\delta'^*(I, w) \in F'$	(da $I' = I$ per Def.)
gdw	$\delta'^*(I, w) \cap F \neq \emptyset$	(Def. von F')
gdw	$\bigcup_{q \in I} \Delta^*(q, w) \cap F \neq \emptyset$	(nach Lemma)
gdw	$\exists q \in I \, \exists q' \in F \ \left(q' \in \Delta^*(q, w)\right)$	
gdw	$w \in L(\mathcal{A}_{\mathrm{NDEA}})$	(Def. der Sprache eines Automaten)

Damit: $L(\mathcal{A}_{\mathrm{DEA}}) = L(\mathcal{A}_{\mathrm{NDEA}})$ □

Teil III

Endliche Automaten

Automaten mit ε-Kanten

Vom NDEA zum Automaten mit ε-Kanten

Bisher (NDEA): Kanten mit einem Buchstaben beschriftet

Jetzt (Automat mit ε-Kanten): Kanten mit einem Wort beschriftet

Es darf auch das leere Wort ε sein!

Ein Automaten mit ε-Kanten kann ...

- in einem Schritt ein ganzes Wort verarbeiten
- einen Zustandsübergang machen, ohne dabei einen Buchstaben zu lesen

Automaten mit ε-Kanten: Definition

Definition 13.1 (Automat mit ε-Kanten)

Ein **Automat mit ε-Kanten (ε-NDEA)** A ist ein Tupel

$$\mathcal{A} = (K, \Sigma, \Delta, I, F)$$

Dabei ist

- K eine endliche Menge von Zuständen,
- Σ ein endliches Alphabet,
- $\Delta \subseteq (K \times \Sigma^*) \times K$ eine eine (endliche) Übergangsrelation,
- $I \subseteq K$ eine Menge von Startzuständen,
- $F \subseteq K$ eine Menge von finalen Zuständen

Automaten mit ε-Kanten: Übergangsrelation

Definition 13.2 (Erweiterung von Δ zu Δ^*)

$$\Delta^* \subseteq (K \times \Sigma^*) \times K$$

ist (ähnlich wie für NDEAs) definiert durch:

$$\Delta^*(\,(q,\varepsilon),\,q'\,) \quad \underline{\text{gdw}} \quad q' = q \quad \text{oder} \quad \Delta(\,(q,\varepsilon),\,q'\,)$$

$$\Delta^*(\,(q,w_1w_2),\,q'\,) \quad \underline{\text{gdw}} \quad \exists q'' \in K$$
$$\Delta^*(\,(q,w_1),\,q''\,) \vee \Delta(\,(q,w_1),\,q''\,)$$
$$\wedge$$
$$\Delta^*(\,(q'',w_2),\,q'\,) \vee \Delta(\,(q'',w_2),\,q'\,)$$

Automaten mit ε-Kanten: Akzeptierte Sprache

Definition 13.3 (Von ε-NDEA akzeptierte Sprache)

Die von einem Automaten mit ε-Kanten

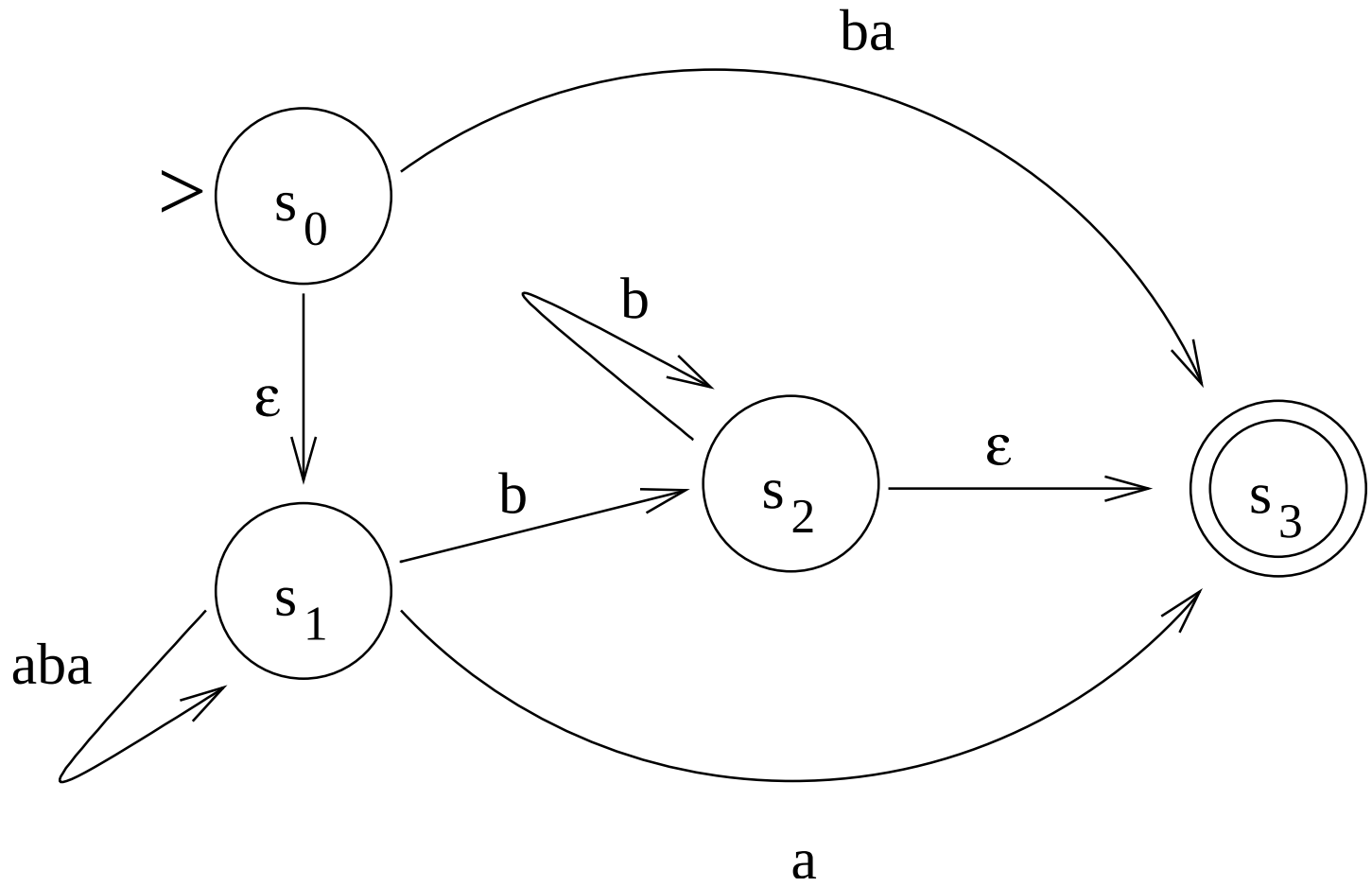
$$\mathcal{A} = (K, \Sigma, \Delta, I, F)$$

akzeptierte Sprache ist

$$L(\mathcal{A}) := \{ w \in \Sigma^* \mid \exists s_0 \in I \exists q \in F \ \Delta^*((s_0, w)\ q) \}$$

Automaten mit ε-Kanten: Beispiel

Beispiel 13.4 (Automaten mit ε-Kanten)



Akzeptiert: $\{aba\}^*\{b\}\{b\}^* + \{aba\}^*\{a\} + \{ba\}$

Gleichmächtigkeit von Automaten mit ε-Kanten und NDEAs

Theorem 13.5 (ε-NDEA gleich mächtig wie NDEA)

Zu jedem Automaten mit ε-Kanten $\mathcal{A}$ existiert ein indeterminierter endlicher Automat $\mathcal{A}'$ mit

$$L(\mathcal{A}) = L(\mathcal{A}')$$

Gleichmächtigkeit: Beweis

Beweis.

Transformation von $\mathcal{A}$ in einen NDEA ohne ε-Kanten

1. Ersetze Übergänge:
   - mit nur einem Buchstaben markiert $\longrightarrow$ beibehalten
   - mit einem Wort w markiert ($|w| = n$) $\longrightarrow$ ersetze durch n Übergänge (verwende $n-1$ neue, zusätzlicher Zustände)
   - ε-Übergänge $\longrightarrow$ statt diesen

     $$\Delta(\,(q,a),\,q''\,)$$

     für jedes Paar

     $$\Delta(\,(q,a),\,q'\,) \quad \text{und} \quad \Delta(\,(q',\varepsilon),\,q''\,)$$

Gleichmächtigkeit von Automaten mit ε-Kanten und NDEAs

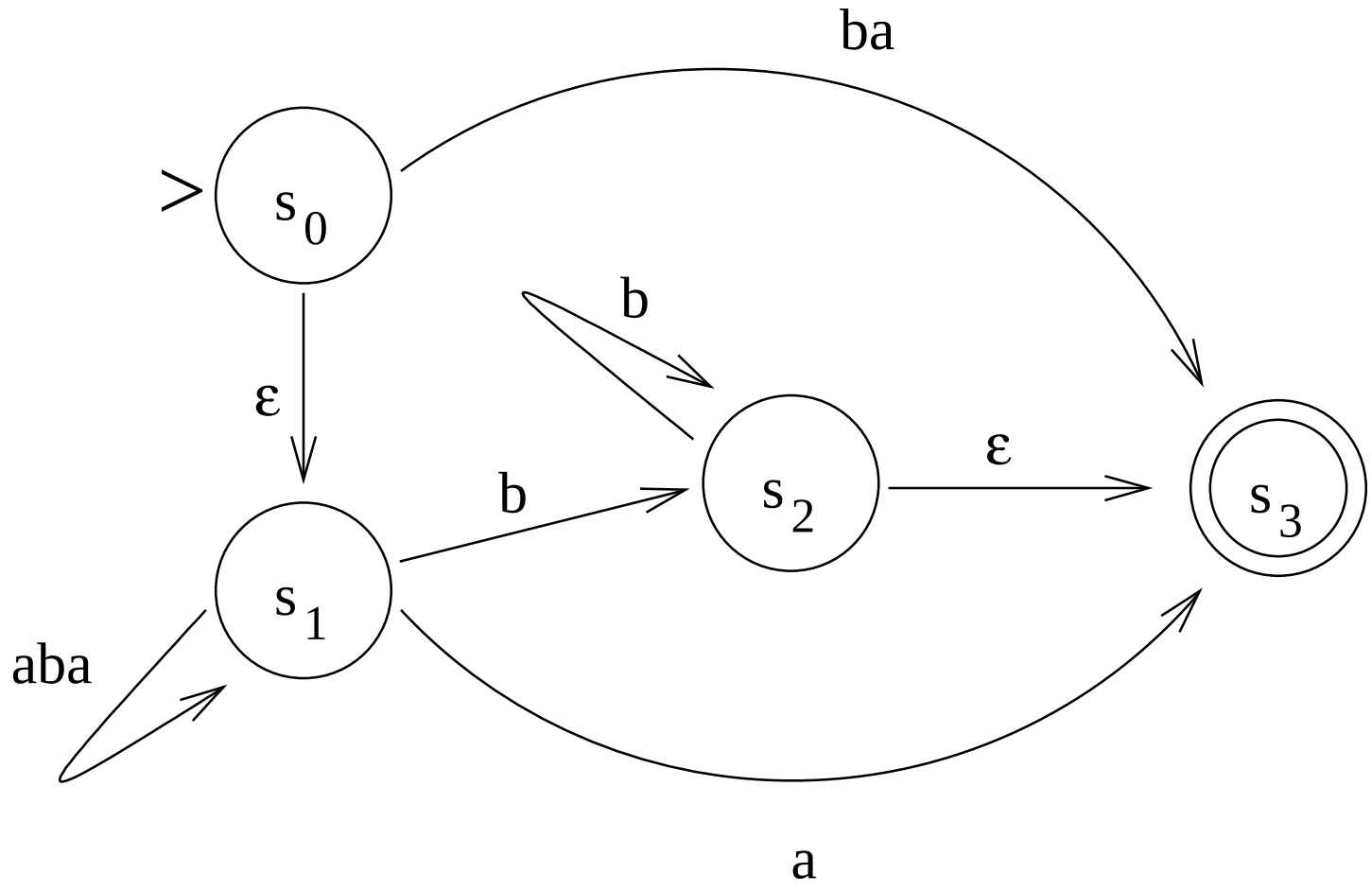
Beweis (Fortsetzugn)

Transformation von $\mathcal{A}$ in einen NDEA ohne ε-Kanten

2. Zusätzliche Initialzustände:
   Falls $q \in I$ und $\Delta^*(\,(q,\varepsilon),\ q'\,)$, dann auch $q' \in I$
3. Finalzustände bleiben unverändert

Gleichmächtigkeit: Beispiel

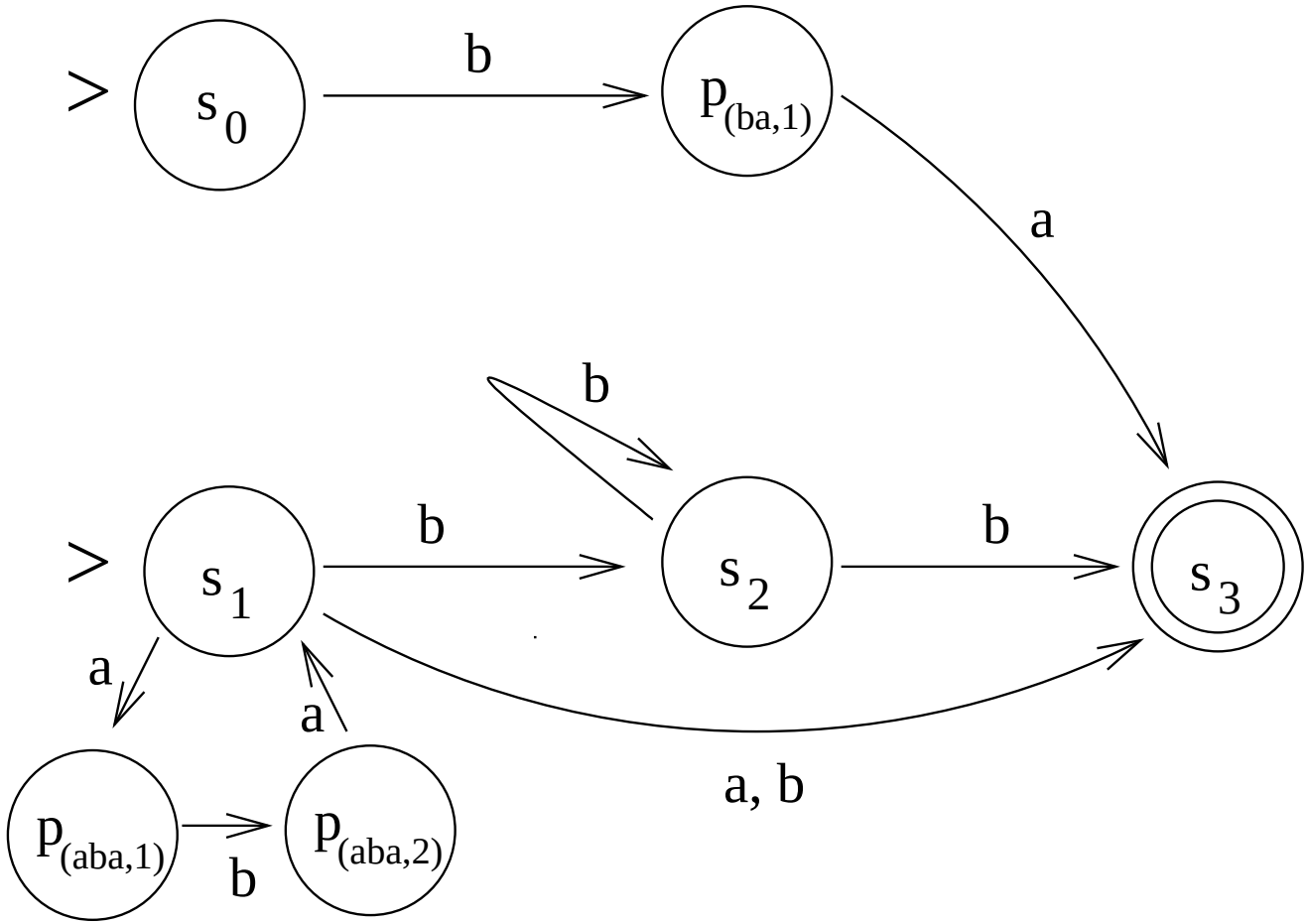
Beispiel 13.6 (Der Automat mit ε-Kanten . . .)



Akzeptiert: $\{aba\}^*\{b\}\{b\}^* + \{aba\}^*\{a\} + \{ba\}$

Gleichmächtigkeit: Beispiel

Beispiel 13.7 (... wird transformiert in den äquivalenten NDEA)



Akzeptiert: $\{aba\}^*\{b\}\{b\}^* + \{aba\}^*\{a\} + \{ba\}$

Teil III

Endliche Automaten

Satz von Kleene

Theorem 14.1 (Satz von Kleene: RAT = L_3)

Eine Sprache L ist rational *gdw* *L ist regulär.*

Merke:

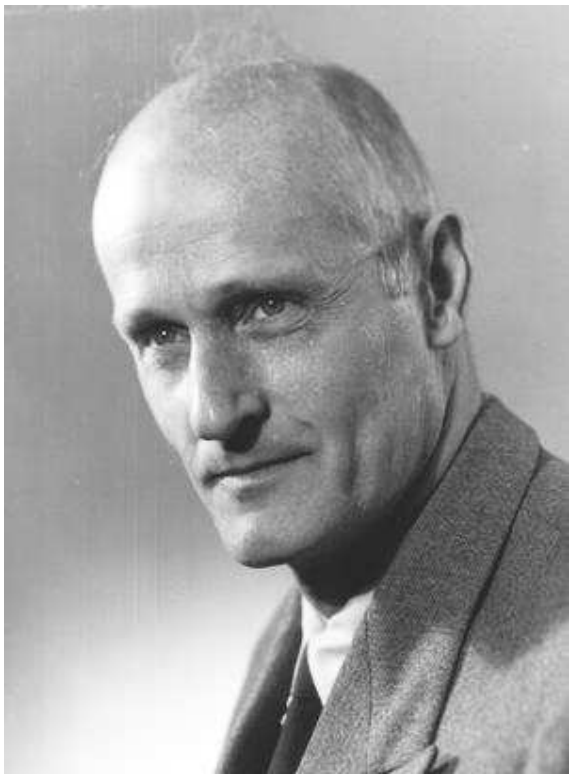
***L* ist rational heißt:** es gibt einen endlichen Automaten, der *L* akzeptiert

***L* ist regulär heißt:** es gibt eine rechtslineare Grammatik für *L*

Satz von Kleene

Stephen Cole Kleene ⋆ 1909, † 1994

- Mathematiker und Logiker
- Promovierte bei Church in Princeton
  (wie Turing und viele andere)
- Professor an der Universität von Wisconsin
- Einer der Begründer der theoretischen Informatik
- Unter anderem:
  Erfinder der Regulären Ausdrücke



Satz von Kleene

Beweis

„$\Rightarrow$“ zu zeigen:

Wenn eine Sprache L von einem endlichen Automaten $\mathcal{A}$ akzeptiert wird, ist sie regulär (wird von einer rechtslinearen Grammatik akzeptiert).

Sei also $L = L(\mathcal{A})$ für einen endlichen Automaten

$$\mathcal{A} = (K, \Sigma, \delta, s_0, F)$$

Dazu konstruieren wir eine Grammatik $G = (V, T, R, S)$:

Automat $\mathcal{A}$:	in **Zustand q**,	**liest a**,	geht in **Zustand q'**
Grammatik:	**Endvariable q**,	**erzeugt a**	neue **Endvariable q'**

Satz von Kleene

Beweis (Fortsetzung)

Formale Definition der Grammatik:

$$V := K$$
$$T := \Sigma$$
$$S := s_0$$
$$R := \{\, q \to aq' \mid \delta(q,a) = q' \,\} \cup \{\, q \to \varepsilon \mid q \in F \,\}$$

Durch Induktion über die Länge eines Wortes w:

$$S \Longrightarrow_G^* wq \quad \underline{\text{gdw}} \quad \delta^*(s_0, w) = q$$

Daraus:
$$\begin{aligned} S \Longrightarrow_G^* w \quad & \underline{\text{gdw}} \quad \exists q \in F \left(S \Longrightarrow_G^* wq \Longrightarrow w \right) \\ & \underline{\text{gdw}} \quad \exists q \in F \left(\delta(s_0, w) = q \right) \\ & \underline{\text{gdw}} \quad w \in L(\mathcal{A}) \end{aligned}$$

Satz von Kleene

Beweis (Fortsetzung)

„$\Leftarrow$“ zu zeigen:

Wenn eine Sprache L regulär ist
(sie wird von einer rechtslinearen Grammatik akzeptiert),
dann gibt es einen endlichen Automaten $\mathcal{A}$ der sie akzeptiert.

Sei also $L = L(G)$ für eine rechtslineare Grammatik

$$G = (V, T, R, S)$$

Dazu konstruieren wir einen ε-NDEA $\mathcal{A} = (K, \Sigma, \Delta, I, F)$ mit:

$$K := V \cup \{q_{stop}\} \qquad (q_{stop} \text{ neu})$$
$$I := \{S\}$$
$$\Sigma := T$$
$$F := \{q_{stop}\}$$

Satz von Kleene

Beweis (Fortsetzung)

Definition von Δ:

$$\Delta(\ (X,u),\ X'\ ) \ :\Longleftrightarrow_{def}\ X \to uX' \in R$$
$$\Delta(\ (X,u),\ q_{stop}\ ) \ :\Longleftrightarrow_{def}\ X \to u \in R$$

für $X, X' \in K$ und $u \in \Sigma^*$

Durch Induktion über die Länge einer Ableitung:

$$S \Longrightarrow_G^* w \quad \underline{\text{gdw}} \quad \Delta^*(\ (S,w),\ q_{stop}\ ) \quad \underline{\text{gdw}} \quad w \in L(\mathcal{A})$$

Wegen Gleichmächtigkeit von ε-NDEA- mit DEA-Automaten gibt es dann auch einen determinierten endlichen Automaten, der L akzeptiert.

□

Satz von Kleene

Beispiel 14.2

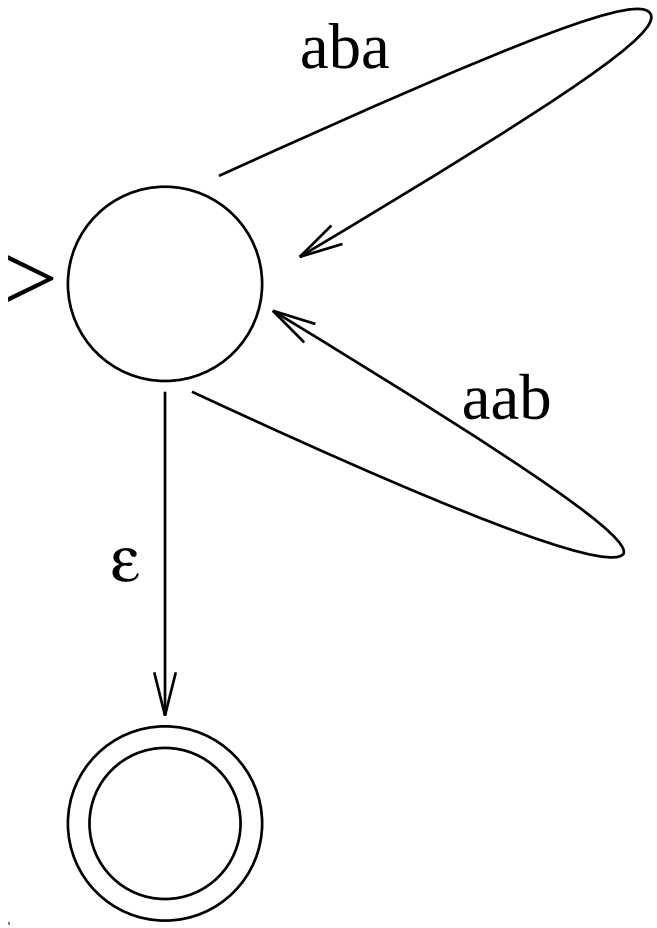
Grammatik G mit Regeln

$$S \to abaS$$
$$S \to aabS$$
$$S \to \varepsilon$$

Sprache

$$L(G) = \{aba, aab\}^*$$

ε-NDEA:



Teil III

Endliche Automaten

Pumping-Lemma

„Aufpumpbarkeit“ (informell)

Lange Wörter $x \in L$ lassen sich zerlegen

$$x = uvw \qquad |v| \geq 1$$

so dass

$$u \underbrace{vv \ldots v}_{i \text{ mal}} w = uv^m w$$

wieder in L liegt (für alle $m \geq 1$)

Pumping-Lemma

Pumping Lemma (informell)

Zu jeder regulären Sprache L gibt es ein $n \in \mathbb{N}$, so dass alle Wörter

$$w \in L \qquad \text{mit } |w| \geq n$$

aufgepumpt werden können

Anwendung

- Wichtige Information über die Struktur regulärer Sprachen
- Nachweis der Nicht-Regularität von Sprachen:
  Wenn das Pumping-Lemma für eine Sprache nicht gilt,
  dann kann sie nicht regulär sein

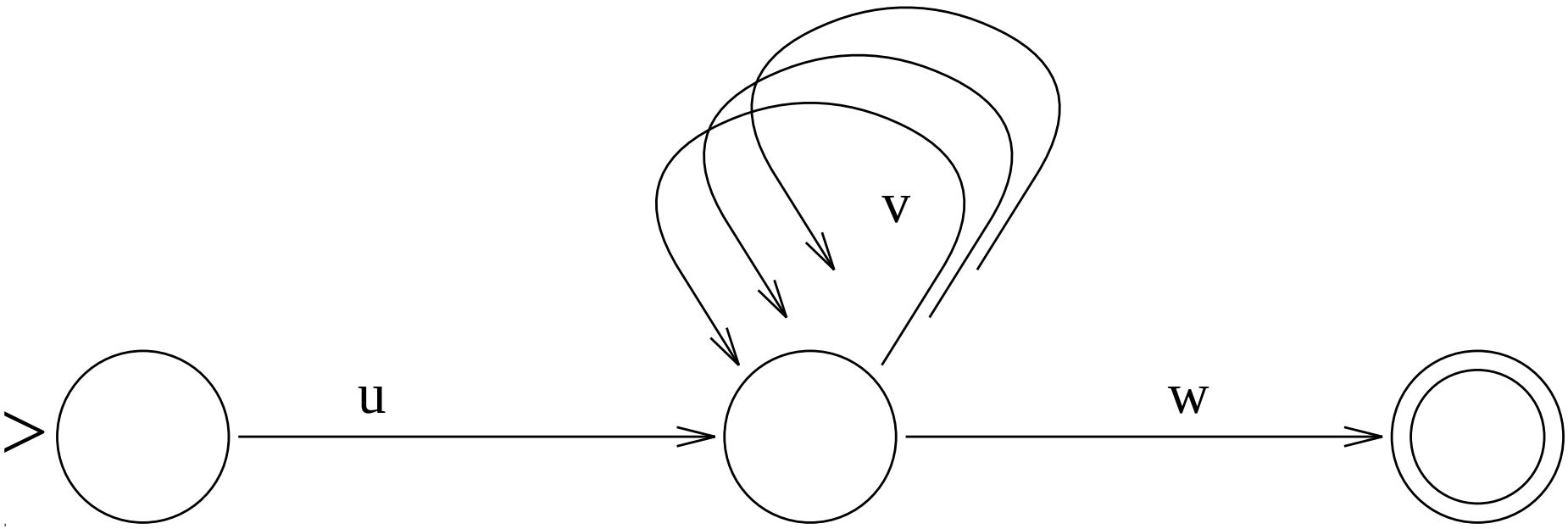
Pumping-Lemma: Intuition

Warum gilt das Pumping-Lemma?

- Zu regulärer Sprache L gibt es einen DEA, der L akzeptiert
- Dieser hat endliche Zustandsmenge K.
  Sei $m := |K|$.
- Wenn $|w| > |K|$, dann muss beim Akzeptieren von w eine Schleife durchlaufen werden.
- Die Schleife kann auf mehrfach durchlaufen werden.
- Das Teilwort v, das der Schleife entspricht kann aufgepumpt werden.

Pumping-Lemma: Intuition

Abstrakt gesehen



Pumping-Lemma: Formal

Theorem 15.1 (Pumping-Lemma für L_3-Sprachen)

*Sei $L \in$ **RAT**.*
Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \textit{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \qquad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L \quad$ *für alle* $m \in \mathbb{N}$

Pumping-Lemma: Beweis

Beweis

Sei L eine reguläre Sprache.

1. Fall: L ist endlich.

Sei w_{max} das längste Wort in L.

Wir setzen

$$n := |w_{max}| + 1$$

Dann gibt es keine Wörter $x \in L$, für die $|x| \geq n$ gilt.

Also gilt dann Pumping-Lemma trivialerweise.

Pumping-Lemma: Beweis

Beweis (Fortsetzung)

2. Fall: L ist unendlich.

Sei

$$\mathcal{A} = (K, \Sigma, \delta, s_0, F)$$

ein endlicher Automat (DEA), der L akzeptiert.

Wir setzen

$$n := |K| + 1$$

Pumping-Lemma: Beweis

Beweis (Fortsetzung)

Wir betrachten ein beliebiges Wort

$$x = x_1 x_2 \dots x_t \in L \quad \text{mit} \quad |x| = t \geq n, \quad x_i \in \Sigma$$

Zu zeigen: x lässt sich aufpumpen.

Seien

$$q_0, q_1, \dots, q_t \in K,$$

die Zustände, die beim Akzeptieren von x durchlaufen werden:

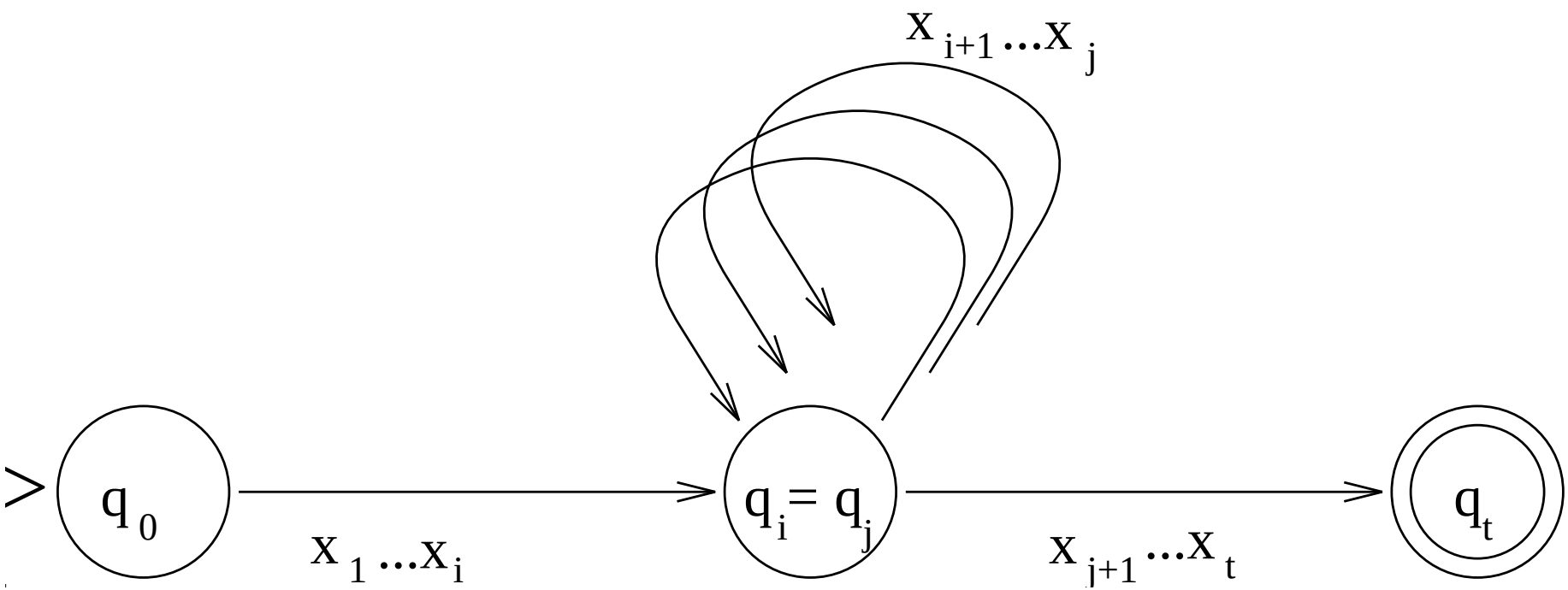
$$q_0 = s_0, \quad q_t \in F, \quad \delta(q_i, x_{i+1}) = q_{i+1} \ \ \forall 0 \leq i \leq t-1$$

Pumping-Lemma: Beweis

Beweis (Fortsetzung)

Da $t \geq |K| + 1$, muss es $0 \leq i < j \leq t - 1$ geben mit

- $q_i = q_j$
- $|j - i| \leq |K|$



Pumping-Lemma: Beweis

Beweis (Fortsetzung)

Wähle nun:

$$\left.\begin{aligned} u &:= x_1 \dots x_i \\ v &:= x_{i+1} \dots x_j \\ w &:= x_{j+1} \dots x_t \end{aligned}\right\} \qquad x = uvw \text{ mit } 1 \leq |v| < n.$$

Damit:

- Für alle $m \geq 0$ gibt es Wege

$$q_0, \dots, q_{i-1}, \underbrace{q_i, \dots, q_j = q_i, \dots, q_j = \cdots = q_i \dots, q_j}_{m \text{ mal}}, q_{j+1}, \dots, q_t$$

- Also: $uv^m w$ wird von $\mathcal{A}$ akzeptiert.
- Also: $uv^m w \in L$

Pumping-Lemma: Umkehrung

Korollar

Wenn für eine Sprache das Pumping-Lemma **nicht** gilt,
dann ist sie **nicht** regulär.

Vorsicht

- Es gibt nicht-reguläre Sprachen, für die das Pumping-Lemma gilt.
- Daraus, dass das Pumping-Lemma für eine Sprache gilt,
  folgt **nicht**, dass sie regulär ist.

Pumping-Lemma: Anwendung der Umkehrung

Beispiel 15.2

Folgende Sprachen sind **nicht regulär**:

1. $L_1 := \{a^i b a^i \mid i \in \mathbb{N}_0\}$
2. $L_2 := \{a^p \mid p \text{ ist Primzahl}\}$

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_1

Zu

$$L_1 := \{a^i b a^i \mid i \in \mathbb{N}_0\}$$

Annahme: L_1 ist regulär.

Dann gilt für L_1 das Pumping-Lemma.

Sei n die Zahl aus dem Pumping-Lemma.

Dann muss sich das Wort

$$a^n b a^n \in L_1$$

aufpumpen lassen (da $|a^n b a^n| \geq n$).

Sei $a^n b a^n = uvw$ eine passende Zerlegung laut Lemma.

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_1 (Forts.)

1. Fall: $u = a^k, v = a^j, w = a^i b a^n$ mit $i, k \geq 0$, $j > 0$ und $k + j + i = n$.
Einmal aufpumpen ($m = 2$) ergibt:

$$uv^2w = a^k a^{2j} a^i b a^n = a^{k+2j+i} b a^n = a^{n+j} b a^n \notin L_1$$

Widerspruch zum Lemma!

2. Fall: $u = a^n b a^i, v = a^j, w = a^k$
Widerspruch zum Lemma! (analog zu Fall 1)

3. Fall: $u = a^k, v = a^j b a^i, w = a^l$ mit $k + j = i + l = n$ und $i, j, k, l \geq 0$
Einmal aufpumpen ($m = 2$) ergibt:

$$uv^2w = a^k a^j b a^i a^j b a^i a^l = a^{k+j} b a^{i+j} b a^{i+l} \notin L_1$$

Widerspruch zum Lemma!

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_1 (Forts.)

Also: Annahme falsch.

Also: L_1 nicht regulär. □

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_2

Zu

$$L_2 := \{a^p \mid p \text{ ist Primzahl}\}$$

Annahme: L_2 ist regulär.

Dann gilt für L_2 das Pumping-Lemma.

Sei n die Zahl aus dem Pumping-Lemma.

Dann muss sich jedes Wort

$$a^p \in L_2 \quad \text{mit} \quad p \geq n$$

aufpumpen lassen.

Sei $a^p = uvw$ eine passende Zerlegung laut Lemma.

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_2 (Forts.)

Sei

$$a^p = uvw = a^i a^j a^k$$

also

$$i + j + k = p \geq n \qquad \text{und} \qquad 0 < j < n$$

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_2 (Forts.)

Fall 1: $i+k>1$.

Pumpe $(i+k)$ mal:

$$uv^{i+k}w = a^i a^{j(i+k)} a^k$$

Nach Pumping-Lemma liegt dieses Wort in L_2, d. h.

$$i+j(i+k)+k \quad \text{prim}$$

Aber **Widerspruch**:

$$\begin{aligned} i+j(i+k)+k &= i+ij+jk+k \\ &= i(1+j)+(j+1)k \\ &= i(1+j)+k(1+j) \\ &= (i+k)(1+j) \end{aligned}$$

Pumping-Lemma: Anwendung der Umkehrung

Beweis der Nichtregularität von L_2 (Forts.)

Fall 2: $i + k = 1$.

Pumpe $(j+2)$ mal:

$$uv^{j+2}w = a^i a^{j(j+2)} a^k$$

Nach Pumping-Lemma liegt dieses Wort in L_2, d. h.

$$i + j(j+2) + k \quad \text{prim}$$

Aber **Widerspruch!**:

$$\begin{aligned} i + j(j+2) + k &= 1 + j(j+2) \\ &= 1 + 2j + j^2 \\ &= (1+j)^2 \end{aligned}$$

Also: Annahme falsch. L_2 nicht regulär. □

Pumping-Lemma: Stärkere Variante

Theorem 15.3 (Pumping-Lemma für L_3-Sprachen, stärkere Variante)

*Sei $L \in$ **RAT**.*
Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \textit{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \qquad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|uv| < n$ *(statt $|v| < n$)*
- $uv^m w \in L$ *für alle $m \in \mathbb{N}$*

Pumping-Lemma: Anwendung der stärkeren Variante

Beispiel 15.4 (Palindrome)

Die Sprache der Palindrome

$$L = \{ww^{-1} \mid w \in \{a,b\}^*\}$$

ist nicht regulär

- Beweis gelingt **nicht** mit der schwächeren Variante des PL (die schwächere Version gilt für die Sprache)
- Beweis **gelingt** mit der stärkeren Varianten des PL

Teil III

Endliche Automaten

Abschlusseigenschaften

Lemma 16.1

Seien zwei reguläre Sprachen L, L' gegeben.
Dann kann folgende endlichen Automaten konstruieren:

- $\mathcal{A}_\neg$ *akzeptiert* $\overline{L} = \Sigma^* \setminus L$
- $\mathcal{A}_\cup$ *akzeptiert* $L \cup L'$
- $\mathcal{A}_\circ$ *akzeptiert* $L \circ L'$
- $\mathcal{A}_*$ *akzeptiert* L^*
- $\mathcal{A}_\cap$ *akzeptiert* $L \cap L'$

Beweis.

An Tafel. □

Abschlusseigenschaften

Theorem 16.2 (Abschlusseigenschaften von L_3)

Wenn L, L' reguäre Sprachen sind, dann sind auch

- $\overline{L}$
- $L \cup L'$
- $L \circ L'$
- L^*
- $L \cap L'$

reguläre Sprachen.

Beweis.

Gemäß Lemma existieren Automaten, die diese Sprachen akzeptieren.
Also sind sie regulär. □

Wortprobleme

Lemma 16.3

Sei $\mathcal{A}$ ein endlicher Automat.
Es ist entscheidbar, ob die Sprache $L(\mathcal{A})$

1. *leer ist.*
2. *unendlich ist.*

Korollar

Sei G eine rechtslineare Grammatik.
Es ist entscheidbar, ob die Sprache $L(G)$

1. leer ist.
2. unendlich ist.

Wortprobleme

Beweis.

Zu (i): $L(\mathcal{A})$ ist nicht leer

gdw

Es gibt einen Weg von einem initialen zu einem finalen Zustand.

Zu (ii): $L(\mathcal{A})$ ist unendlich

gdw

Es gibt einen Weg von einem initialen zu einem finalen Zustand, der einen Zyklus enthält.

Beides ist leicht zu überprüfen. □

Wortprobleme

Lemma 16.4

Seien $\mathcal{A}_1, \mathcal{A}_2$ endliche Automaten.

Es ist entscheidbar, ob

$$L(\mathcal{A}_1) = L(\mathcal{A}_2)$$

Korollar

Für rechtlineare Grammatiken G_1, G_2, G_3 und endliche Automaten $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ ist entscheidbar, ob:

$$L(\mathcal{A}_1) \cap L(\mathcal{A}_2) = \emptyset$$
$$L(G_1) \cup L(G_2) = L(G_3)$$

usw.

Beweis

Seien $\mathcal{A}_1$, $\mathcal{A}_2$ endliche Automaten.

Man kann zu $\mathcal{A}_1$ und $\mathcal{A}_2$ einen endlichen Automaten $\mathcal{A}_=$ konstruieren mit

$$L(\mathcal{A}_=) = (L(\mathcal{A}_1) \cap \overline{L(\mathcal{A}_2)}) \cup (\overline{L(\mathcal{A}_1)} \cap L(\mathcal{A}_2))$$

Es gilt

$$L(\mathcal{A}_=) = \emptyset \quad \underline{\text{gdw}} \quad L(\mathcal{A}_1) = L(\mathcal{A}_2)$$

Dies ist entscheidbar.

Teil III

Endliche Automaten

Hauptsatz von Kleene

Theorem 17.1 (Hauptsatz von Kleene)

Die durch endliche Automaten akzeptierten Sprachen sind genau die, die man durch reguläre Ausdrücke beschreiben kann.

Hauptsatz von Kleene: Beweis

Beweis

„$\Rightarrow$“ (schwierigere Richtung)

Gegeben ein DEA $\mathcal{A}$.

Zustände von $\mathcal{A}$ seien $q_1, \ldots, q_n$.

O.B.D.A. sei q_1 der initiale Zustand von $\mathcal{A}$

Induktion über die **Kompliziertheit** des Akzeptierens

Dafür sei genauer:

$$R^k_{i,j} := \{w \in \Sigma^* : \quad \delta^*(q_i, w) = q_j \text{ und für alle Präfixe } u \text{ von } w \text{ mit } \varepsilon \neq u \neq w \text{ gilt } \delta^*(q_i, u) \in \{q_1, q_2, \ldots, q_k\}\}$$

Hauptsatz von Kleene: Beweis

Beweis (Forts.)

Offensichtlich ist

$$L(\mathcal{A}) = \bigcup_{q_f \in F} R^n_{1,f}$$

Es genügt zu zeigen:

Alle Mengen $R^n_{1,f}$ sind durch reguläre Ausdrücke beschreibbar.

Dazu: Durch Induktion über k (an der Tafel):

Für alle $1 \leq i, j \leq n$: $R^k_{1,j}$ is durch einen regulären Ausdruck beschreibb.

Hauptsatz von Kleene: Beweis

Beweis

„$\Leftarrow$“ (einfacherer Richtung)

Durch **Induktion** über den **Aufbau** regulärer Ausdrücke:

Zu jedem regulären Ausdruck gibt es einen äquivlanten ε-NDEA

(an der Tafel)