Grundlagen der Theoretischen Informatik

3. Endliche Automaten

30.04.2015

Viorica Sofronie-Stokkermans

Matthias Horbach

e-mail: sofronie@uni-koblenz.de, horbach@uni-koblenz.de

Bis jetzt

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Jetzt

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

3. Endliche Automaten

3. Endliche Automaten

Inhalt von Teil 3

- Vereinfachtes Modell eines Computers: **endlicher Automat**
-
-
-
-
-

3. Endliche Automaten

Inhalt von Teil 3

- Vereinfachtes Modell eines Computers: **endlicher Automat**
- Die von endlichen Automaten erkannten **“rationalen”** Sprachen sind genau die Typ-3-Sprachen (**rechtslinear**, **regulär**)
-
-
-
-

3. Endliche Automaten

Inhalt von Teil 3

- 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
-
-
-

3. Endliche Automaten

Inhalt von Teil 3

- 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.
-
-

3. Endliche Automaten

Inhalt von Teil 3

- 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.
-

3. Endliche Automaten

Inhalt von Teil 3

- 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.

3. Endliche Automaten

- Determinierte endliche Automaten (DEAs)
- Indeterminierte endliche Automaten (NDEAs)
- Automaten mit epsilon-Kanten
- Endliche Automaten akzeptieren genau die Typ-3-Sprachen
- Pumping Lemma
- Abschlusseigenschaften und Wortprobleme
- Rational $=$ Reguläre Ausdrücke

3. Endliche Automaten

- Determinierte endliche Automaten (DEAs)
- Indeterminierte endliche Automaten (NDEAs)
- Automaten mit epsilon-Kanten
- Endliche Automaten akzeptieren genau die Typ-3-Sprachen
- Pumping Lemma
- Abschlusseigenschaften und Wortprobleme
- Rational $=$ Reguläre Ausdrücke

Determinierte endliche Automaten (DEAs)

Beispiel

Beispiel 1. Die Sprache

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

ist regulär.

Beispiel

Beispiel 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:

$$\begin{array}{rcl} S & \to & aaA \\ A & \to & abA \mid c \end{array}$$

Beispiel

Beispiel 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:

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

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

Grammatiken und Automaten

Grammatik vs. Automat

Grammatiken **erzeugen** Wörter

Automaten **analysieren / erkennen** Wörter

beide beschreiben / definieren Sprachen

Endlicher Automat: Informell

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 möglichen **Zustand**,
- **Kanten** sind mit Buchstaben beschriftet: Sie beschreiben **Zustandsänderungen**.
- **Initiale** Zustände werden mit einem **Pfeil** gekennzeichnet,
- **finale Zustände** mit einem **doppelten Kreis**.

Endlicher Automat: Darstellung als Graph

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

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

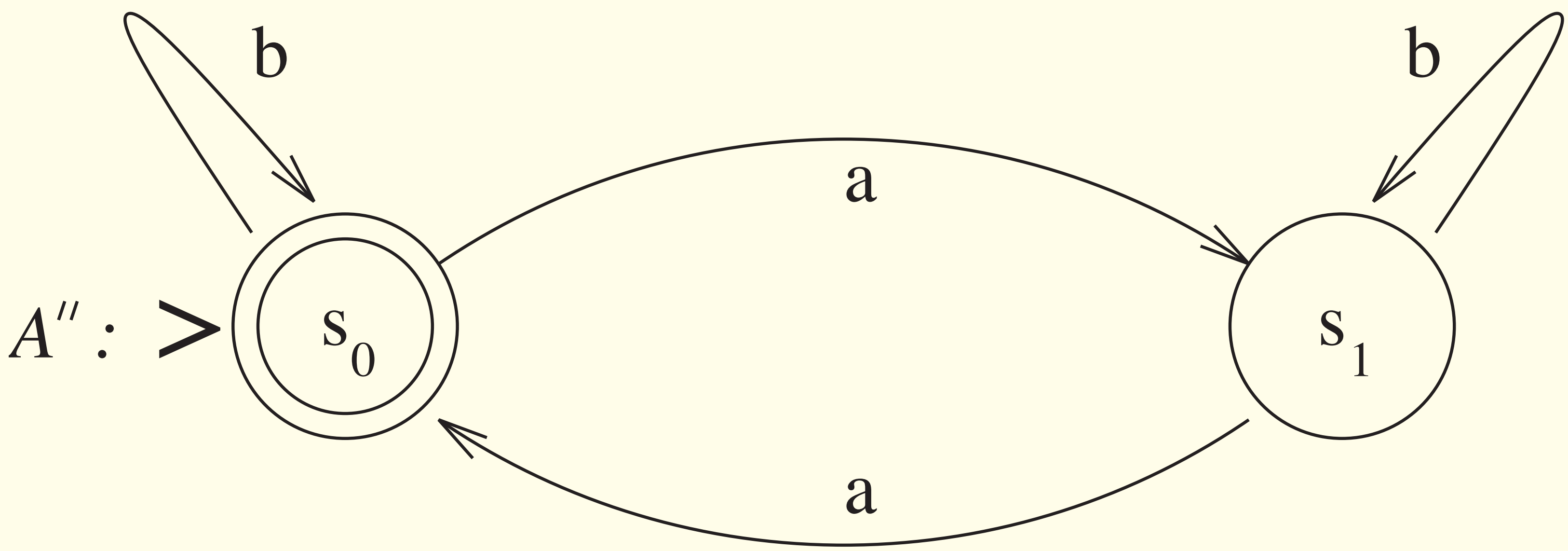
Endlicher Automat: Darstellung als Graph

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

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. 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**, und
- $F \subseteq K$ die Menge der **finalen Zustände**.

Endlicher Automat: Übergangsfunktion

Bedeutung der Übergangsfunktion

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

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

Endlicher Automat: Übergangsfunktion

Bedeutung der Übergangsfunktion

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

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

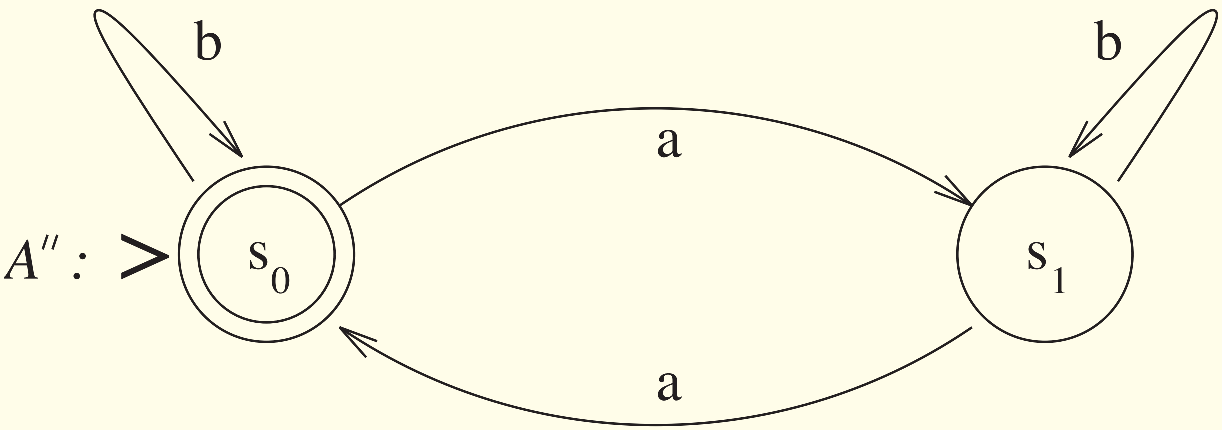
Wir erweitern δ zu δ^*

$\delta^* : K \times \Sigma^* \to K$ ist strukturell rekursiv über Σ^* definiert:

$$\begin{array}{rcl} \delta^*(q, \varepsilon) & := & q \\ \delta^*(q, wa) & := & \delta(\delta^*(q, w), a) \end{array}$$

Wenn klar ist, was gemeint ist, wird δ^* auch einfach als δ geschrieben.

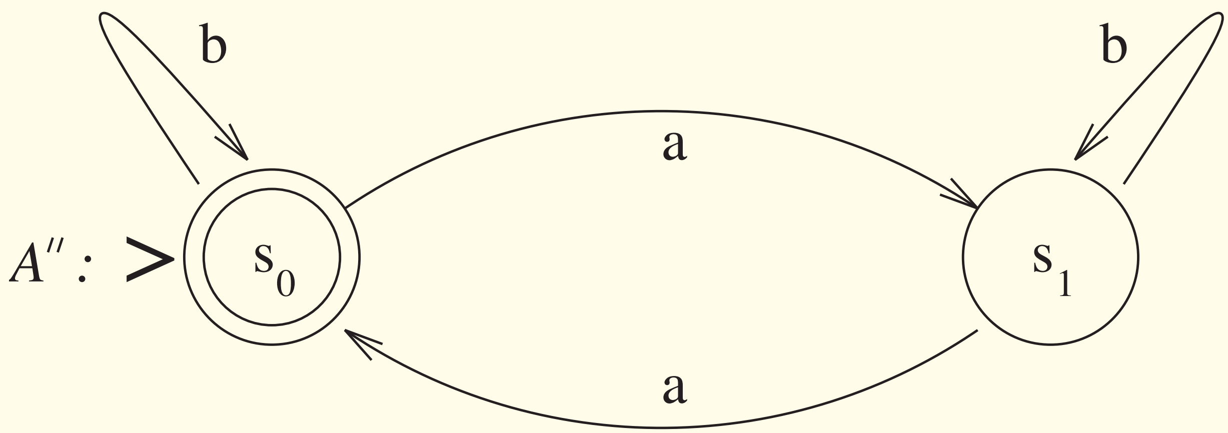
Endlicher Automat: Beispiel



Dieser Automat azeptiert die Sprache

$$\{w \mid \#_a(w) \text{ gerade}\} \subset \{a, b\}^* \text{ (Beispiel Seite 22)}$$

Endlicher Automat: Beispiel



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 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 (Von einem endlichen Automaten akzeptierte Sprache)

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

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

Endlicher Automat: Akzeptierte Sprache

Definition (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 (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

Endlicher Automat: Akzeptierte Sprache

Definition (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 (Von endlichen Automaten akzeptierte Sprachen)
Die Menge

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

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

Wir zeigen demnächst: **RAT** $=$ Menge der regulären Sprachen

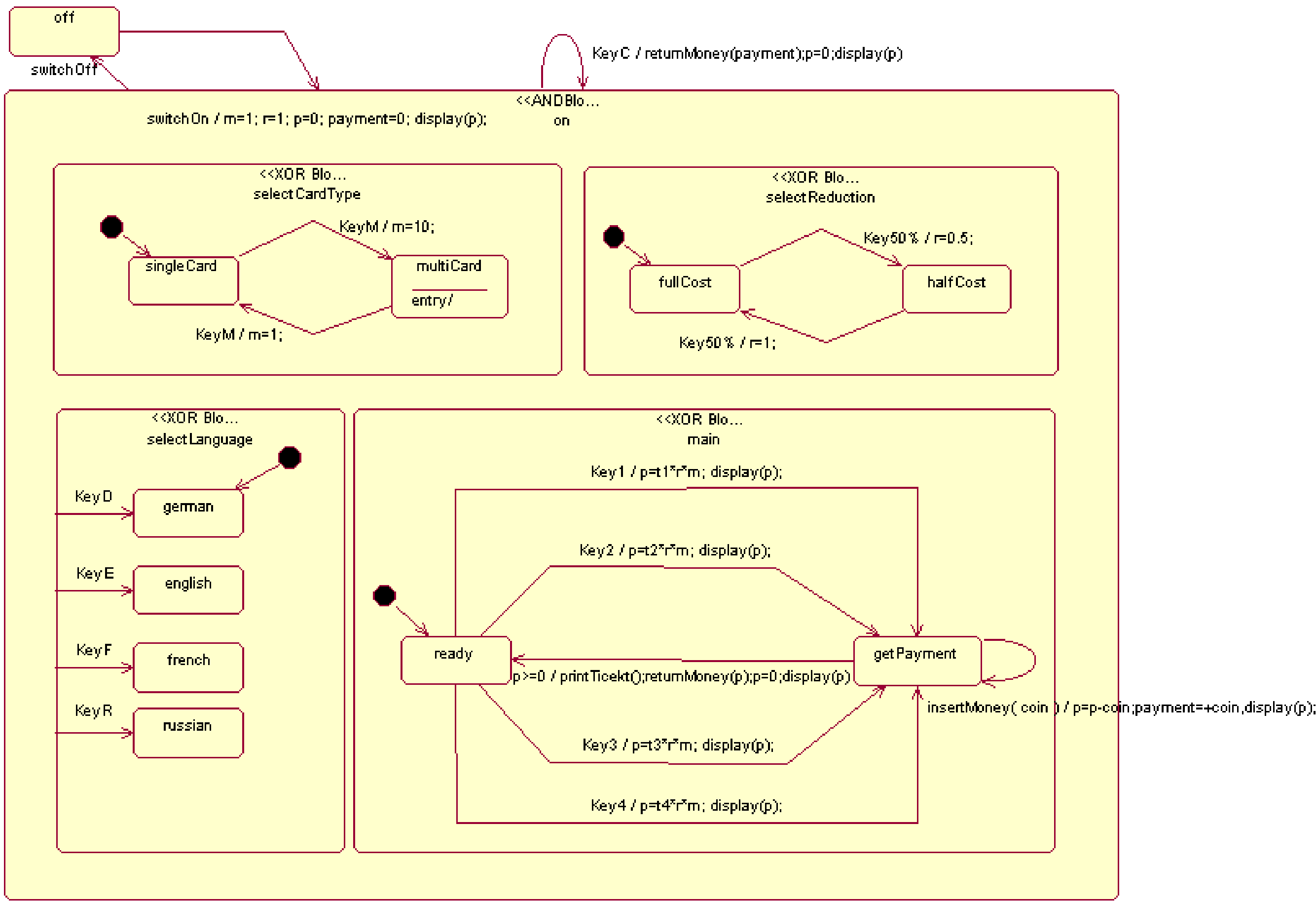
Endlicher Automat: UML State Chart

UML State Charts sind eine (erweiterte) Form endlicher Automaten

Endlicher Automat: UML State Chart

UML State Charts sind eine (erweiterte) Form endlicher Automaten

Beispiel:

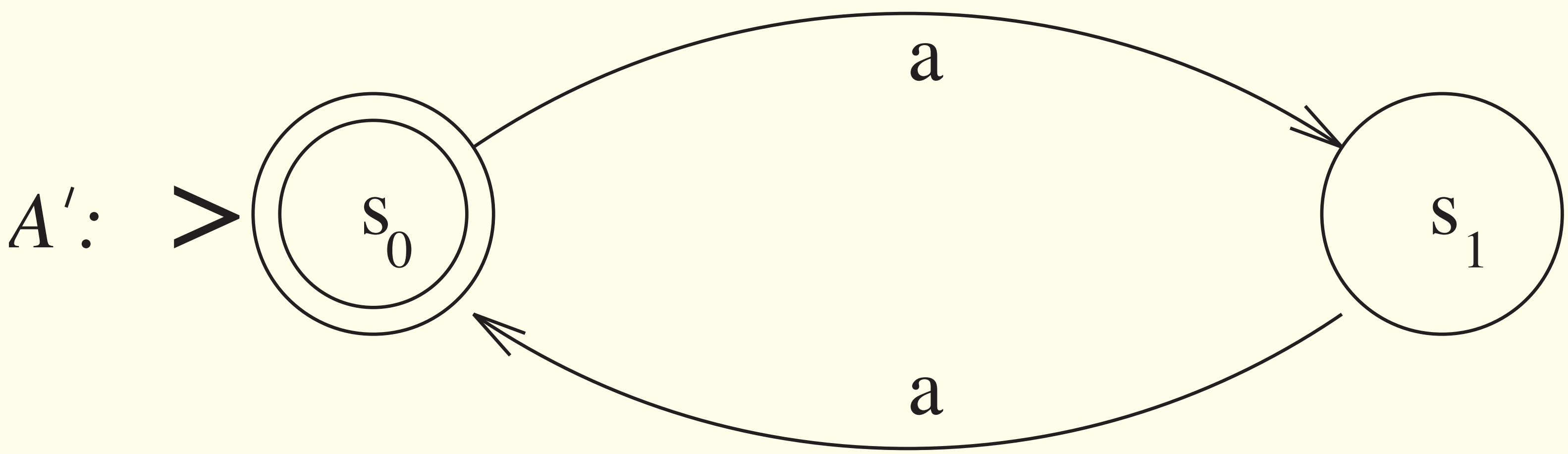


Endliche Automate: Weitere Beispiele

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

Endliche Automate: Weitere Beispiele

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



Endliche Automate: Weitere Beispiele

Beispiel. Die Sprache

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

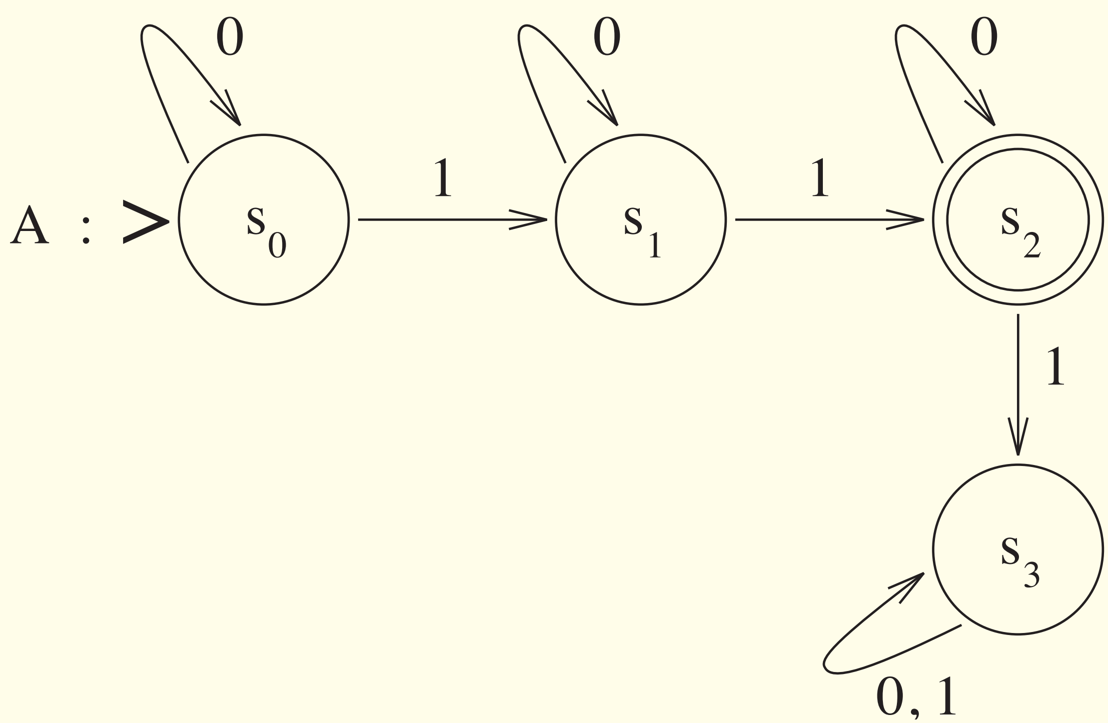
wird akzeptiert von dem folgenden endlichen Automaten:

Endliche Automate: Weitere Beispiele

Beispiel. Die Sprache

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

wird akzeptiert von dem folgenden endlichen Automaten:



Endliche Automate: Weitere Beispiele

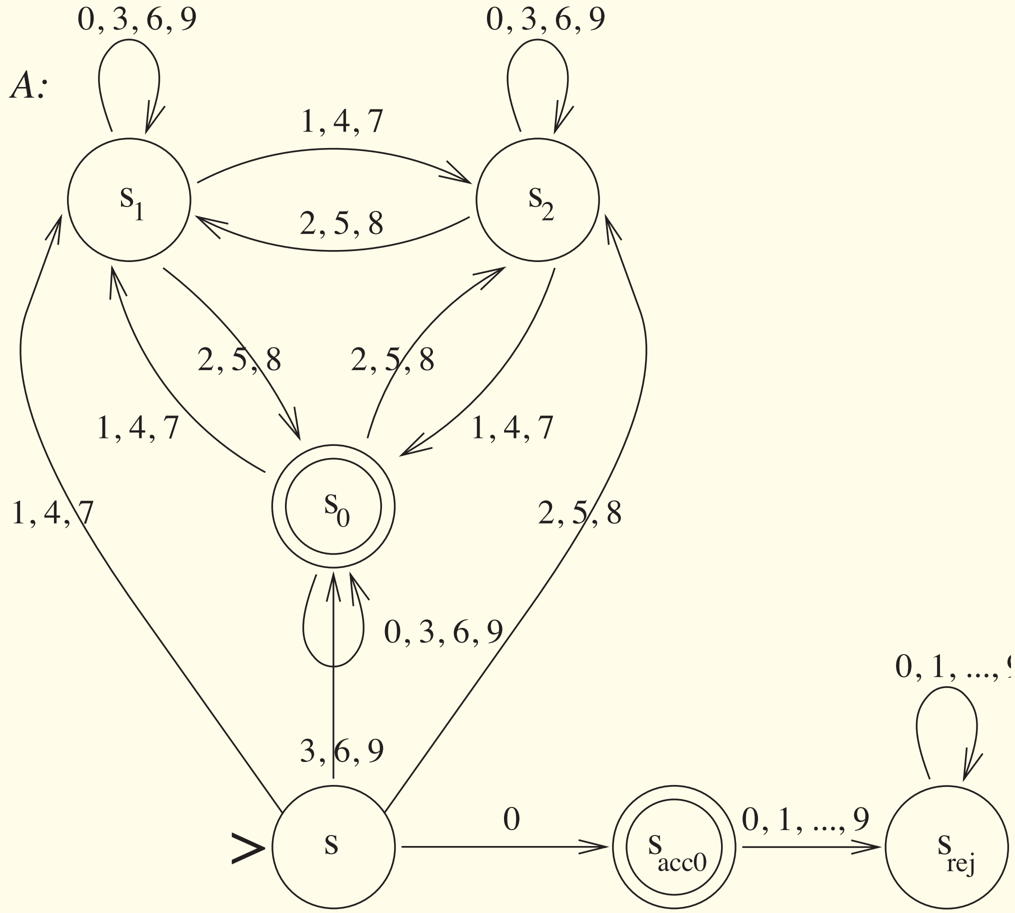
Beispiel

Die Sprache aller durch 3 teilbaren Dezimalzahlen wird akzeptiert durch:

Endliche Automate: Weitere Beispiele

Beispiel

Die Sprache aller durch 3 teilbaren Dezimalzahlen wird akzeptiert durch:



3. Endliche Automaten

- Determinierte endliche Automaten (DEAs)
- Indeterminierte endliche Automaten (NDEAs)
- Automaten mit epsilon-Kanten
- Endliche Automaten akzeptieren genau die Typ-3-Sprachen
- Pumping Lemma
- Abschlusseigenschaften und Wortprobleme
- Rational = Reguläre Ausdrücke

Nächste Vorlesung

- Determinierte endliche Automaten (DEAs)
- Indeterminierte endliche Automaten (NDEAs)
- Automaten mit epsilon-Kanten
- Endliche Automaten akzeptieren genau die Typ-3-Sprachen
- Pumping Lemma
- Abschlusseigenschaften und Wortprobleme
- Rational $=$ Reguläre Ausdrücke