3.0 VU Formale Modellierung

Gernot Salzer

Arbeitsbereich Theoretische Informatik und Logik
Institut für Computersprachen

SS 2016

Inhalt

The Simpsons – Modellierung als Automat

Systemkomponenten: Maggie (M), Knecht Ruprecht (K), Gift (G), Homer+Boot (H), linkes/rechtes Flussufer

Situationsbeschreibung (Systemzustand): $\frac{\text{Lebewesen/Dinge links}}{\text{Lebewesen/Dinge rechts}}$

(Wer/Was befindet sich momentan auf welcher Seite des Flusses?)

Anfangszustand: $\frac{MKGH}{}$

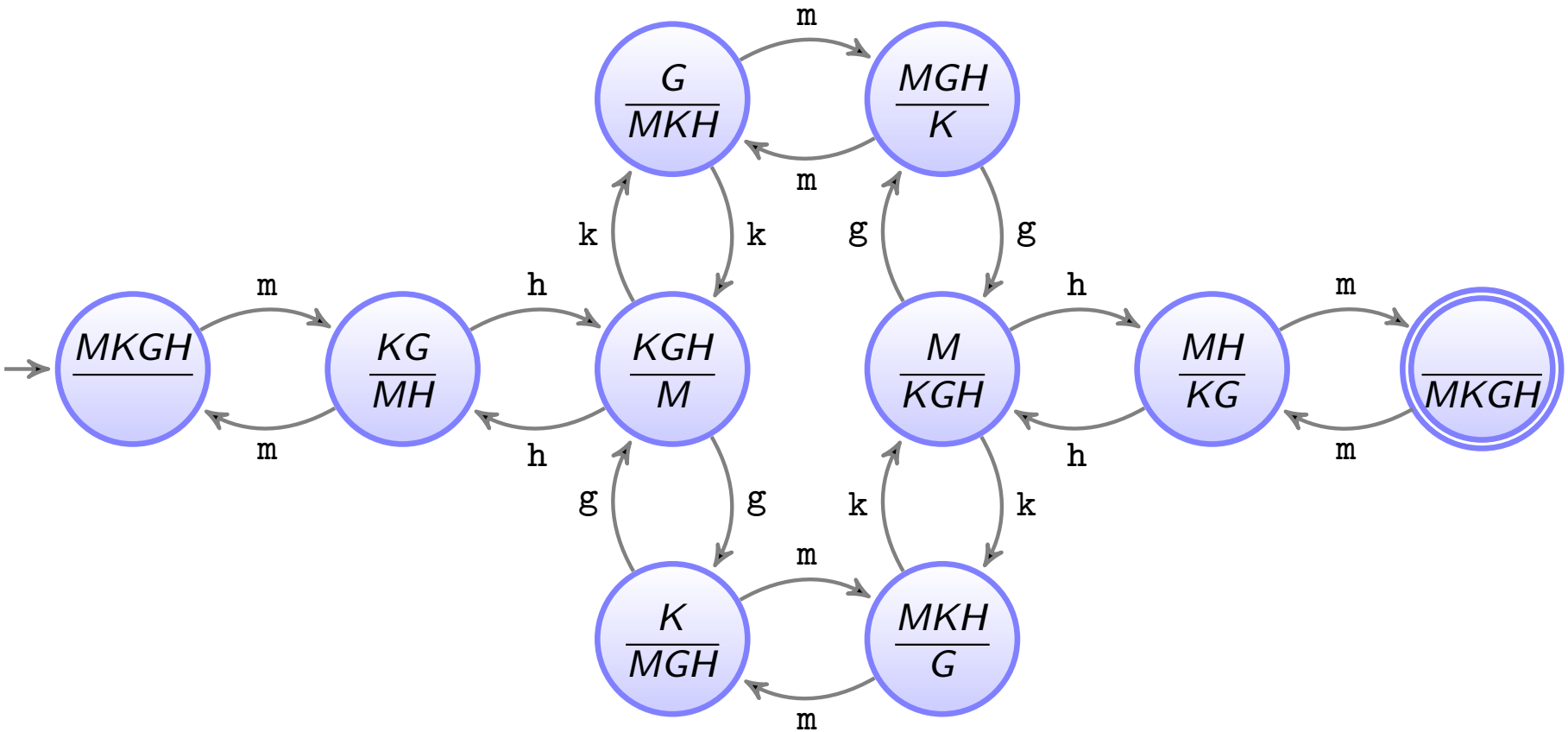
Endzustand (Ziel): $\frac{}{MKGH}$

Verbotene Zustände: $\frac{GH}{MK}$, $\frac{MK}{GH}$, $\frac{KH}{MG}$, $\frac{MG}{KH}$, $\frac{H}{MKG}$, $\frac{MKG}{H}$

Zustandsübergänge:

`h`, `m`, `k`, `g` ... Homer fährt alleine/mit Maggie/KR/Gift über den Fluss.

Automat (ohne verbotene Zustände und Übergänge):



Mögliche Lösungen: $\{\texttt{mhkmghm}, \texttt{mmmhhhgmkhm}, \ldots, \texttt{mhkmgkmgkmghm}, \ldots\}$

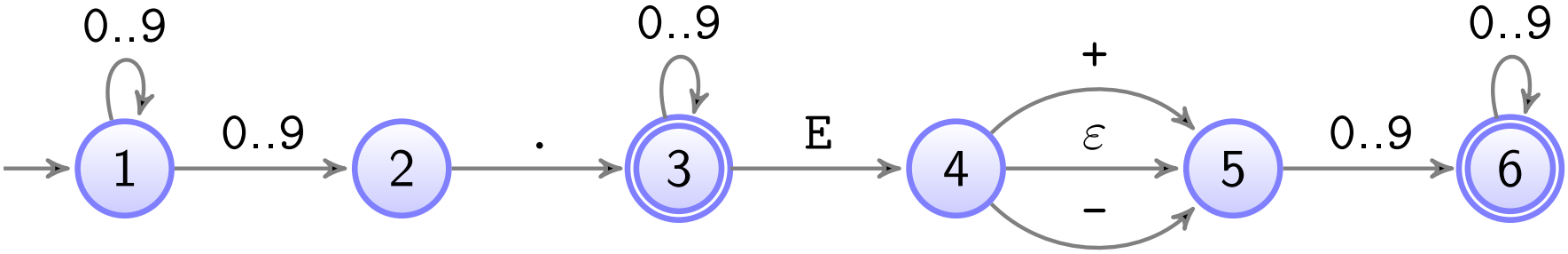
„Sprache des Automaten“

Beispiel: Reelle Numerale mit Exponentialteil

Z.B. `3.14`, `0.314E1` ($= 0.314 \cdot 10^1$), `314.E-2` ($= 314 \cdot 10^{-2}$)

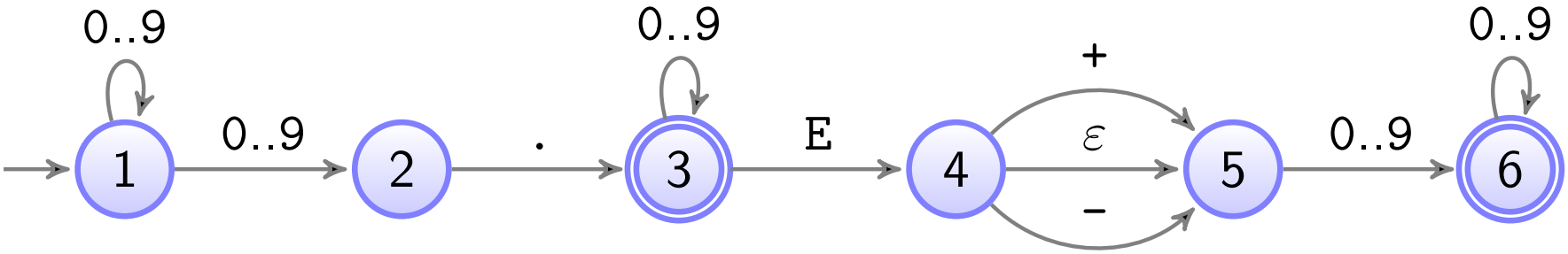
- Mindestens eine Ziffer vor Dezimalpunkt
- Dezimalpunkt
- Nachkommastellen optional
- Exponentialteil optional:
  - eingeleitet durch `E`
  - Vorzeichen optional
  - mindestens eine Ziffer

Endlicher Automat für die reellen Numerale



`0..9` ... Abkürzung für 10 parallele Übergänge beschriftet mit `0` bis `9`.

ε ... Leerwort, „Nichts“



- Zustandsbeschriftungen 1–6 dienen nur der Bezugnahme, irrelevant für das Verhalten des Automaten
- Kanten sind mit Symbolen beschriftet, die gelesen/geschrieben werden.
- Anfangszustand (1) ist durch einen Pfeil aus dem Nichts markiert.
- Endzustände (3, 6) sind durch einen Doppelkreis markiert.

Zwei Sichtweisen:

- Akzeptor: Der Automat liest Symbole und akzeptiert alle Zeichenketten, die vom Anfangs- zu einem der Endzustände führen.
- Generator: Der Automat schreibt Symbole und generiert jene Zeichenketten, die vom Anfangs- zu einem der Endzustände führen.

Inhalt

Endliche Automaten modellieren Systeme bzw. Abläufe, die nur eine begrenzte, feste Zahl an unterscheidbaren Zuständen besitzen.

Kennzeichen:

- endliche Menge von Zuständen
- Übergänge zwischen Zuständen
- Eingaben, die die Übergänge steuern.
- Ausgaben oder Aktionen, die in den Zuständen oder während der Übergänge getätigt werden.
- Anfangszustand
- Endzustände (optional)
- deterministisch: Der momentane Zustand und die nächste Eingabe bestimmen eindeutig den Folgezustand.
  nichtdeterministisch: Es gibt Zustände, die bei manchen Eingaben mehrere mögliche Folgezustände besitzen.

Arten endlicher Automaten

(Klassischer) Endlicher Automat:

- Anfangs- und Endzustände
- nur Eingaben (bzw. nur Ausgaben)
- Ein-/Ausgaben verknüpft mit Zustandsübergängen
- verarbeitet endliche Symbolfolgen
- Unterarten: deterministisch, nichtdeterministisch mit/ohne ε-Übergängen

Transducer: wie endlicher Automat, aber mit Ein- und Ausgaben.

Mealy-Automat: deterministischer Transducer; in der Regel keine Endzustände, verarbeitet daher unendliche Symbolfolgen.

Moore-Automat: wie Mealy-Automat, die Ausgaben sind aber mit den Zuständen verknüpft.

Büchi-Automat: wie endlicher Automat, verarbeitet aber unendliche Symbolfolgen

Weitere Typen: Verallgemeinerter endlicher Automat, Muller-Automat, Rabin-Automat, Baumautomaten, ...

Englische Begriffe: automaton/automata, finite state machine, DFA (Deterministic Finite Automaton), NFA (Non-Determinstic FA)

Weitere (nicht-endliche) Automatenarten: Kellerautomaten (Push-down automata), Turing-Maschinen, Registermaschinen etc. können Ausgaben wieder lesen $\Rightarrow$ zusätzlicher Speicher, mächtiger als endliche Automaten.

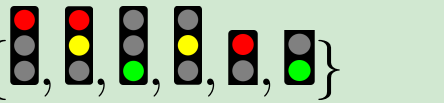
Spezifikation von Automaten:

- Graphisch: Zustände sind Knoten, Übergänge sind Kanten, Ein- und Ausgaben sind Beschriftungen von Knoten und Kanten.
- Tabellarisch: Zu jedem Zustand und Eingabesymbol gibt es einen Eintrag mit zugehöriger Ausgabe und den Folgezuständen.

Inhalt

Formale Sprachen

Alphabet (Σ): endliche, nicht-leere Menge atomarer Symbole

- Menge aller lateinischen Buchstaben, Ziffern und Sonderzeichen
- Menge aller ägyptischen Hieroglyphen
- {  }
- $\{0, \ldots, 9, ., \mathtt{E}, +, -\}$
- $\{0, 1\}$
- $\{00, 01, 10, 11\}$

Wort über Σ: (endliche) Folge von Zeichen aus dem Alphabet Σ

ε ... Leerwort

$\Sigma^+ = \{ s_1 \cdots s_n \mid s_i \in \Sigma, 1 \leq i \leq n \}$ ... Menge aller nicht-leeren Wörter über Σ

$\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$ ... Menge aller Wörter über Σ (inklusive Leerwort)

$w_1 \cdot w_2 = w_1 w_2$ ... Verkettung der Wörter $w_1, w_2 \in \Sigma^*$

$\langle \Sigma^*, \cdot, \varepsilon \rangle$ bildet ein Monoid

D.h.: Für alle Wörter $u, v, w \in \Sigma^*$ gelten folgende Gleichungen:

$(u \cdot v) \cdot w = u \cdot (v \cdot w)$ Assoziativität

$w \cdot \varepsilon = \varepsilon \cdot w = w$ Neutralität

$\Sigma = \{0, 1\}$

$\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$

$10 \cdot \varepsilon \cdot 11101 \cdot 000 = 1011101000$ (Klammerung irrelevant, Assoziativität!)

$\varepsilon \cdot \varepsilon \cdot \varepsilon = \varepsilon$

Formale Sprache über Σ: beliebige Teilmenge von Σ^*

- die Menge aller deutschen Sätze (Alphabet: Buchstaben+Satzzeichen)
- die Menge aller Java-Programme (Alphabet: ASCII-Zeichen)
- $\{\}$, $\{\varepsilon\}$, Σ^*

2^{Σ^*} ... Menge aller Sprachen über Σ = Menge aller Teilmengen von Σ^*

Inhalt

Deterministische endliche Automaten

Deterministischer endlicher Automat (DEA)

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$, wobei

- Q ... endliche Menge der Zustände
- Σ ... Eingabealphabet (*input alphabet*)
- $\delta\colon Q \times \Sigma \mapsto Q$ ... Übergangsfunktion (total) (*transition function*)
- $q_0 \in Q$ ... Anfangszustand (*initial state*)
- $F \subseteq Q$ ... Menge der Endzustände (*final states*)

δ ist eine totale Funktion: Folgezustand $\delta(q, s)$ ist für jeden Zustand $q \in Q$ und jede Eingabe $s \in \Sigma$ eindeutig definiert. $\Longrightarrow$ „deterministisch“

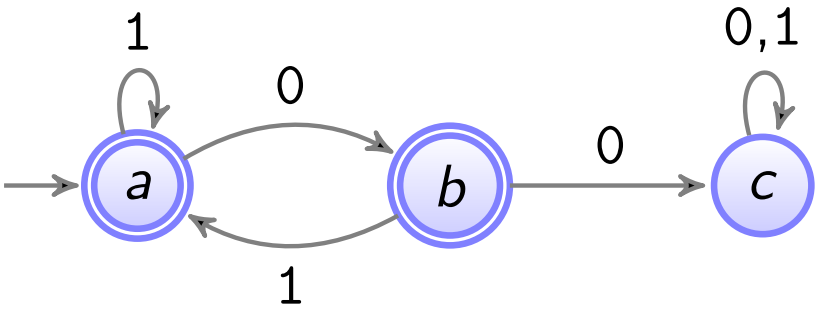
Erweiterte Übergangsfunktion $\delta^*\colon Q \times \Sigma^* \mapsto Q$

$\delta^*(q, \varepsilon) = q, \quad \delta^*(q, sw) = \delta^*(\delta(q, s), w)$ für alle $q \in Q$, $s \in \Sigma$, $w \in \Sigma^*$.

Akzeptierte/Generierte Sprache

$\mathcal{L}(\mathcal{A}) = \{\, w \in \Sigma^* \mid \delta^*(q_0, w) \in F \,\}$

Beispiel: 00-freie Binärstrings



c ... „Falle“, Fehlerzustand
wird oft auch weggelassen

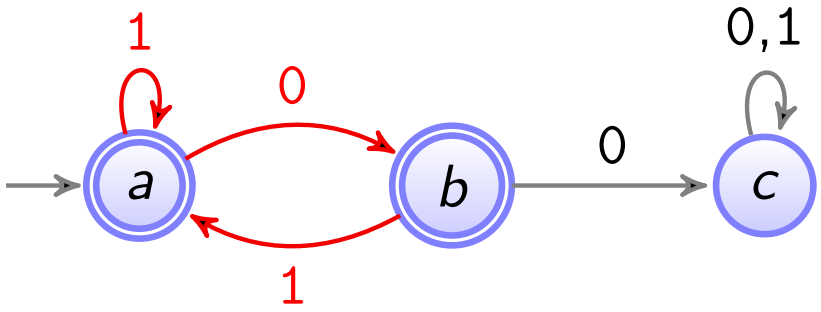
$\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$, wobei

- $Q = \{a, b, c\}$ ... Zustandsmenge
- $\Sigma = \{0, 1\}$ ... Eingabealphabet
- $\delta\colon Q \times \Sigma \mapsto Q$ ... Übergangsfunktion definiert durch:

δ	0	1
a	b	a
b	c	a
c	c	c

- $q_0 = a$ ... Anfangszustand
- $F = \{a, b\}$ ... Endzustände

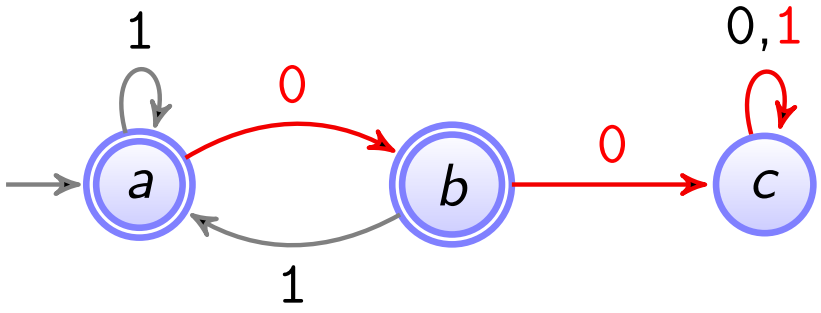
Beispiel: 00-freie Binärstrings



$$
\begin{aligned}
\delta^*(a, 101) &= \delta^*(\delta(a, 1), 01) \\
&= \delta^*(a, 01) \\
&= \delta^*(\delta(a, 0), 1) \\
&= \delta^*(b, 1) \\
&= \delta^*(\delta(b, 1), \varepsilon) \\
&= \delta^*(a, \varepsilon) \\
&= a
\end{aligned}
\qquad
\begin{aligned}
&\delta^*(q, sw) = \delta^*(\delta(q, s), w) \\
&\\
&\\
&\\
&\\
&\delta^*(q, \varepsilon) = q
\end{aligned}
$$

Das Wort 101 wird von $\mathcal{A}$ akzeptiert/generiert, d.h., $101 \in \mathcal{L}(\mathcal{A})$, weil $\delta^*(a, 101) = a$ ein Endzustand ist.

Beispiel: 00-freie Binärstrings



$$\begin{aligned}\delta^*(a,001) &= \delta^*(\delta(a,0),01) \\ &= \delta^*(b,01) \\ &= \delta^*(\delta(b,0),1) \\ &= \delta^*(c,1) \\ &= \delta^*(\delta(c,1),\varepsilon) \\ &= \delta^*(c,\varepsilon) \\ &= c\end{aligned}$$

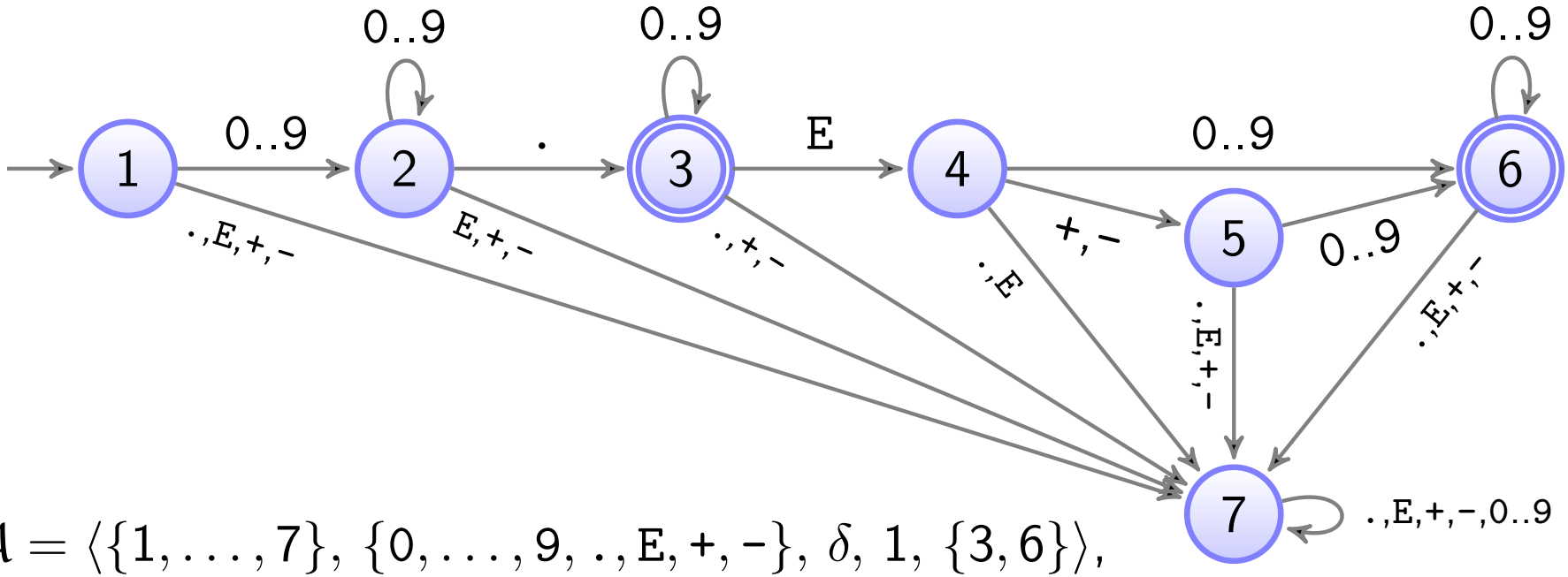
$\delta^*(q,sw) = \delta^*(\delta(q,s),w)$

$\delta^*(q,\varepsilon) = q$

Das Wort 001 wird von $\mathcal{A}$ nicht akzeptiert/generiert, $001 \notin \mathcal{L}(\mathcal{A})$, weil $\delta^*(a,001) = c$ kein Endzustand ist.

$\mathcal{L}(\mathcal{A}) = \{\, w \in \{0,1\}^* \mid 00 \text{ kommt nicht in } w \text{ vor} \,\}$

Beispiel: reelle Numerale

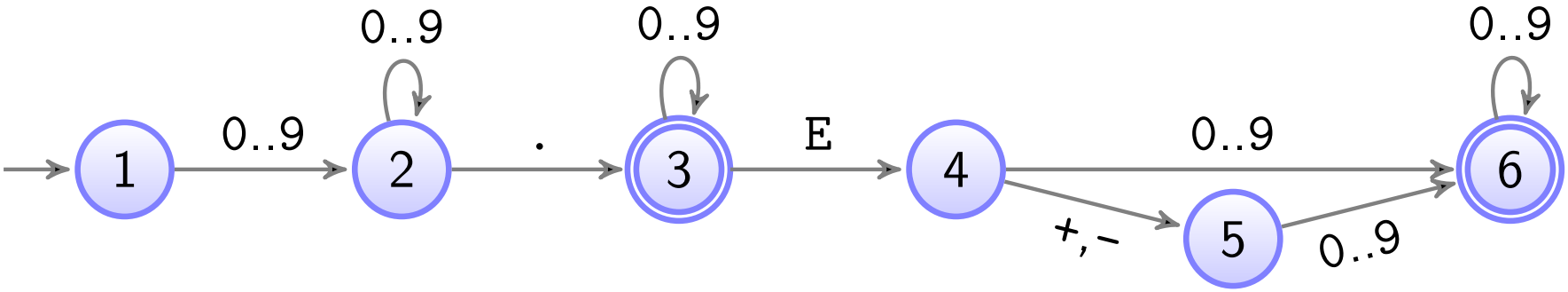


$\mathcal{A} = \langle \{1, \ldots, 7\},\ \{0, \ldots, 9, .\, , \mathtt{E}, +, -\},\ \delta,\ 1,\ \{3, 6\} \rangle,$

wobei

δ	$0 \cdots 9$	.	E	+	−
1	$2 \cdots 2$	7	7	7	7
2	$2 \cdots 2$	3	7	7	7
3	$3 \cdots 3$	7	4	7	7
4	$6 \cdots 6$	7	7	5	5
5	$6 \cdots 6$	7	7	7	7
6	$6 \cdots 6$	7	7	7	7
7	$7 \cdots 7$	7	7	7	7

Beispiel: reelle Numerale



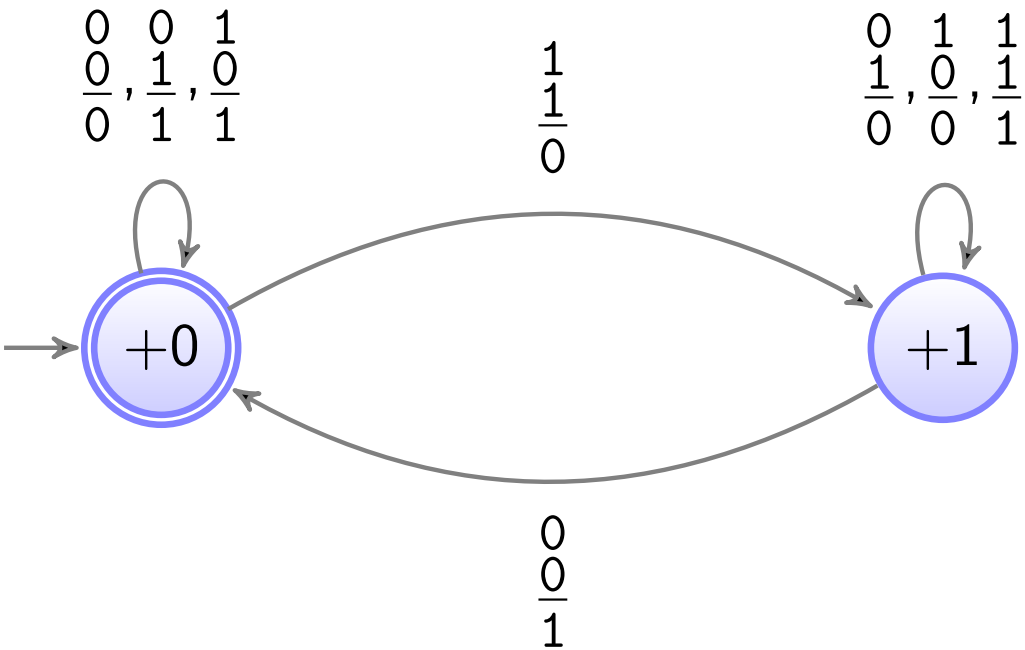
	δ	0 ··· 9	.	E	+	−
AZ	1	2 ··· 2	7	7	7	7
	2	2 ··· 2	3	7	7	7
EZ	3	3 ··· 3	7	4	7	7
	4	6 ··· 6	7	7	5	5
	5	6 ··· 6	7	7	7	7
EZ	6	6 ··· 6	7	7	7	7
	7	7 ··· 7	7	7	7	7

Beispiel: Binäraddition von rechts nach links (Kontrolle)

$$\begin{array}{cccccc|l} 0 & 0 & 1 & 0 & 1 & 1 & = 11_{10} \\ 0 & 0 & 0 & 1 & 0 & 1 & = 5_{10} \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & = 16_{10} \end{array}$$

$\longleftarrow$

$Q = \{+0, +1\}$ $\quad$ $\Sigma = \{\frac{0}{0}, \frac{0}{1}, \frac{1}{0}, \frac{1}{1}, \frac{0}{0}, \frac{0}{1}, \frac{1}{0}, \frac{1}{1}\}$



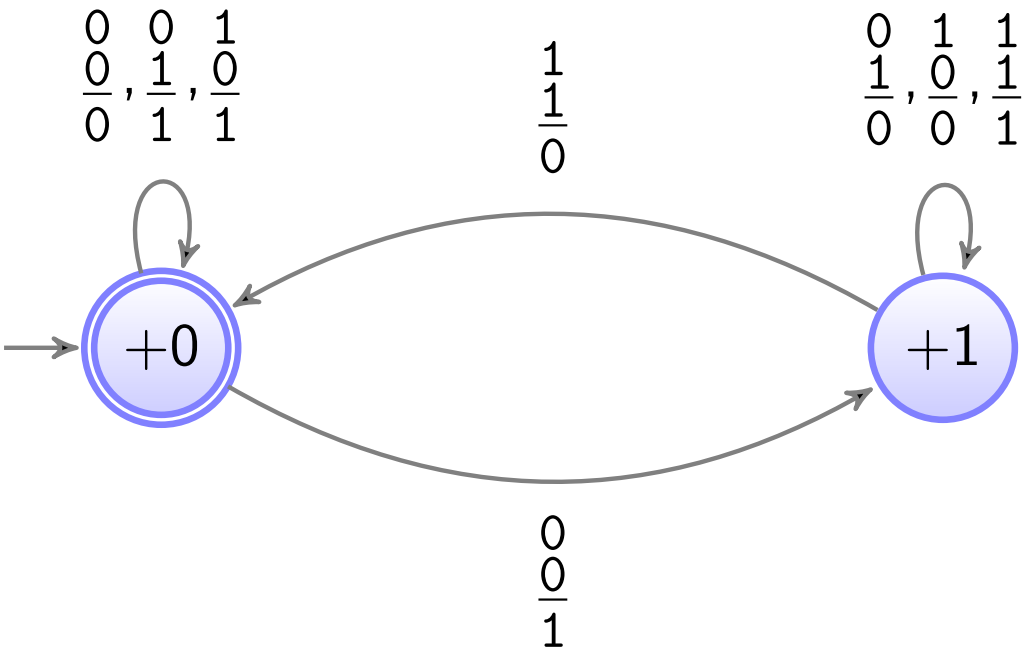
+0 . . . kein Übertrag
+1 . . . Übertrag

Beispiel: Binäraddition von links nach rechts (Kontrolle)

$$\begin{array}{cccccc|l} 0 & 0 & 1 & 0 & 1 & 1 & = 11_{10} \\ 0 & 0 & 0 & 1 & 0 & 1 & = 5_{10} \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & = 16_{10} \end{array}$$

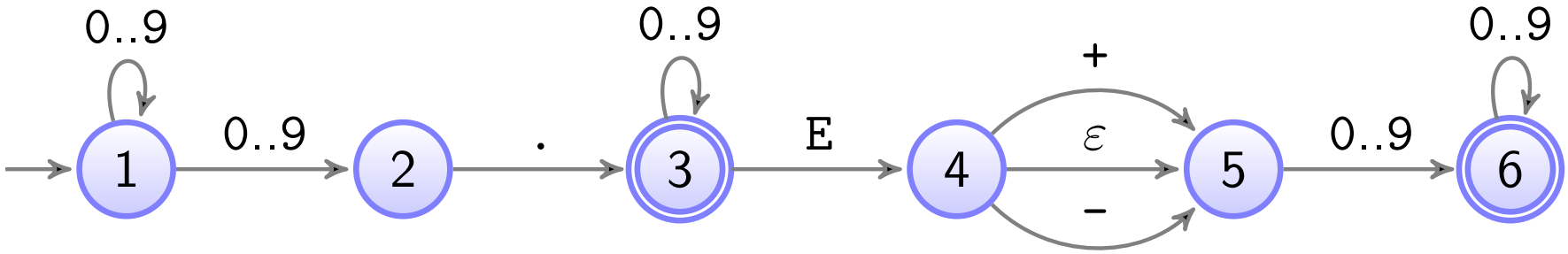
$\longrightarrow$

$Q = \{+0, +1\}$ $\quad$ $\Sigma = \{\frac{0}{0}, \frac{0}{1}, \frac{1}{0}, \frac{1}{1}, \frac{0}{0}, \frac{0}{1}, \frac{1}{0}, \frac{1}{1}\}$



+0 . . . kein Übertrag
+1 . . . Übertrag

Inhalt



Kein deterministischer Automat!

- $\delta(1, 0) = 1$?
  $\delta(1, 0) = 2$?
  Die Übergangsfunktion muss ein eindeutiges Ergebnis besitzen.
- $\delta(4, \varepsilon) = 5$?
  Die Übergangsfunktion ist vom Typ $Q \times \Sigma \mapsto Q$, aber $\varepsilon \notin \Sigma$!

Indeterminismus: Der momentane Zustand und das Eingabesymbol legen den nächsten Zustand bzw. die nächste Aktion nicht eindeutig fest.

- In Zustand 1 sind bei Eingabe `0` die Folgezustände 1 und 2 möglich.
- In Zustand 3 sind bei Eingabe `E` die Folgezustände 4 und 5 möglich.
- In Zustand 4 ist Zustand 5 mit und ohne Eingabe erreichbar.

Ob die richtige Entscheidung getroffen wurde, wird erst später klar.

Nichtdeterministische endliche Automaten

Nichtdeterministischer endlicher Automat (NEA)

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$, wobei

- Q, Σ, q_0, F ... siehe DEAs
- $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ ... Übergangsrelation

DEA: $\delta\colon Q \times \Sigma \mapsto Q$ ... totale Übergangsfunktion

Erweiterte Übergangsrelation $\delta^* \subseteq Q \times \Sigma^* \times Q$

δ^* ist die kleinste Menge mit folgenden Eigenschaften:

- $(q, \varepsilon, q) \in \delta^*$ für alle $q \in Q$
- Wenn $(q_1, w, q_2) \in \delta^*$ und $(q_2, s, q_3) \in \delta$, dann $(q_1, ws, q_3) \in \delta^*$.

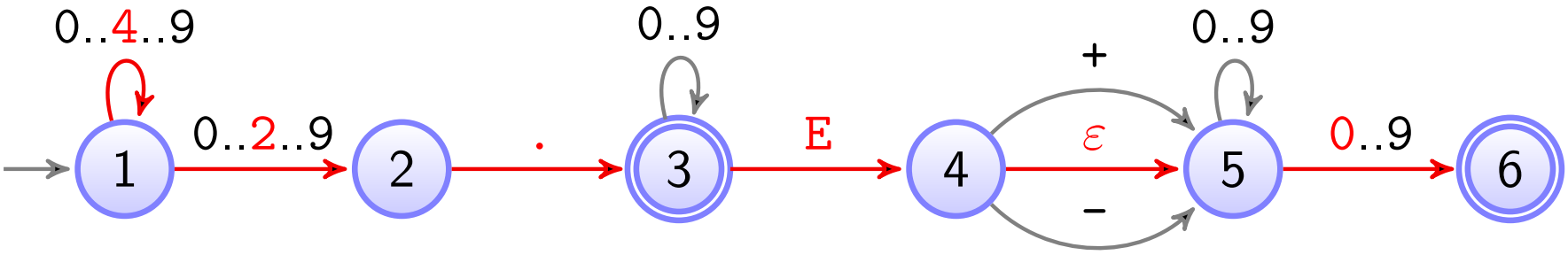
DEA: $\delta^*(q, \varepsilon) = q, \quad \delta^*(q, sw) = \delta^*(\delta(q, s), w)$

Akzeptierte/Generierte Sprache

$\mathcal{L}(\mathcal{A}) = \{ w \in \Sigma^* \mid (q_0, w, q_f) \in \delta^* \text{ für ein } q_f \in F \}$

DEA: $\mathcal{L}(\mathcal{A}) = \{ w \in \Sigma^* \mid \delta^*(q_0, w) \in F \}$

Beispiel: 42.E0 ist ein reelles Numeral



$\mathcal{L}(\mathcal{A}) = \{\, w \in \Sigma^* \mid (1, w, 3) \in \delta^* \text{ oder } (1, w, 6) \in \delta^* \,\}$

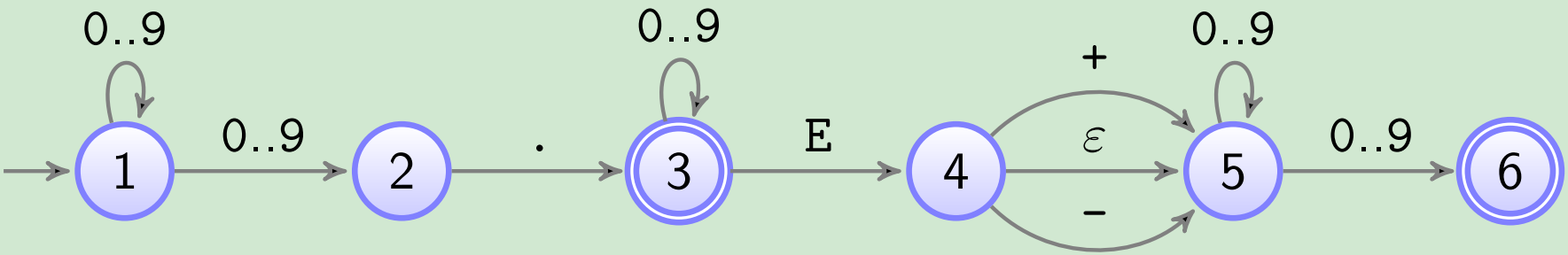
Zu zeigen: $\texttt{42.E0} \in \mathcal{L}(\mathcal{A})$

Wenn $(q_1, w, q_2) \in \delta^*$ und $(q_2, s, q_3) \in \delta$, dann $(q_1, ws, q_3) \in \delta^*$.		
$(1, \varepsilon, 1)$	$(1, 4, 1)$	$(1, 4, 1)$
$(1, 4, 1)$	$(1, 2, 2)$	$(1, 42, 2)$
$(1, 42, 2)$	$(2, ., 3)$	$(1, 42., 3)$
$(1, 42., 3)$	$(3, \texttt{E}, 4)$	$(1, 42.\texttt{E}, 4)$
$(1, 42.\texttt{E}, 4)$	$(4, \varepsilon, 5)$	$(1, 42.\texttt{E}, 5)$
$(1, 42.\texttt{E}, 5)$	$(5, 0, 6)$	$(1, 42.\texttt{E0}, 6)$

$(1, 42.\texttt{E0}, 6) \in \delta^*$, 1 ist Anfangs- und 6 Endzustand $\Longrightarrow \texttt{42.E0} \in \mathcal{L}(\mathcal{A})$

Tabellarische Darstellung der Übergangsrelation $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$

Für jeden Zustand $q \in Q$ und jede Eingabe $s \in \Sigma \cup \{\varepsilon\}$:
Tabelleneintrag mit der Menge $\{\, q' \in Q \mid (q, s, q') \in \delta \,\}$ der Folgezustände



	δ	0	$\cdots$	9	.	E	+	–	ε
AZ	1	$\{1,2\}$	$\cdots$	$\{1,2\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
	2	$\{\}$	$\cdots$	$\{\}$	$\{3\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
EZ	3	$\{3\}$	$\cdots$	$\{3\}$	$\{\}$	$\{4\}$	$\{\}$	$\{\}$	$\{\}$
	4	$\{\}$	$\cdots$	$\{\}$	$\{\}$	$\{\}$	$\{5\}$	$\{5\}$	$\{5\}$
	5	$\{5,6\}$	$\cdots$	$\{5,6\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
EZ	6	$\{\}$	$\cdots$	$\{\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$

Alternative Definition von NEAs:
Übergangsfunktion $\delta\colon Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$ (ist total!) an Stelle von
Übergangsrelation $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$

Vergleich DEA – NEA

NEAs sind flexibler:

- Mehrere Folgezustände pro Zustand und Eingabe möglich;
- Kein Folgezustand zu einem Zustand und einer Eingabe erlaubt;
- Zustandswechsel ohne Eingabe möglich (ε-Übergang).

DEAs und NEAs besitzen dieselbe Ausdrucksstärke.

- Jeder DEA ist per Definition auch ein NEA.
- Zu jedem NEA gibt es einen DEA, der dieselbe Sprache akzeptiert.
  (Lässt sich automatisch finden, siehe später.)

Vorteile von NEAs:

- Benötigen teilweise erheblich weniger Zustände und Übergänge als äquivalente DEAs.
  Die Zustandszahl im DEA kann exponentiell größer sein als im NEA.
- Bei Modellierungsaufgaben leichter zu konstruieren.

Vorteile von DEAs:

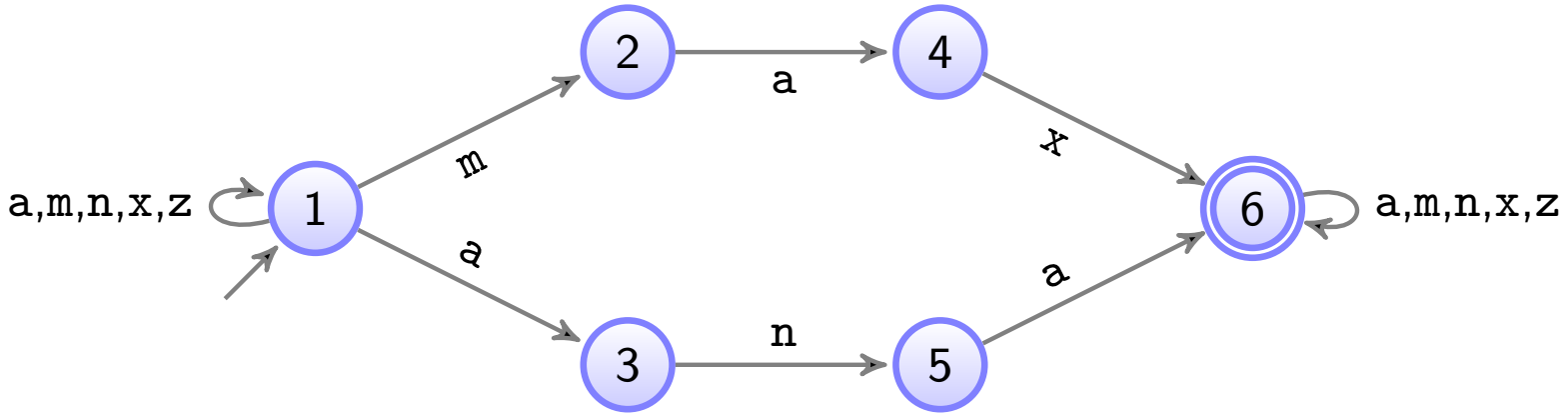
- Effiziente Abarbeitung, kein Backtracking.

Beispiel: Suche nach Max und Ana

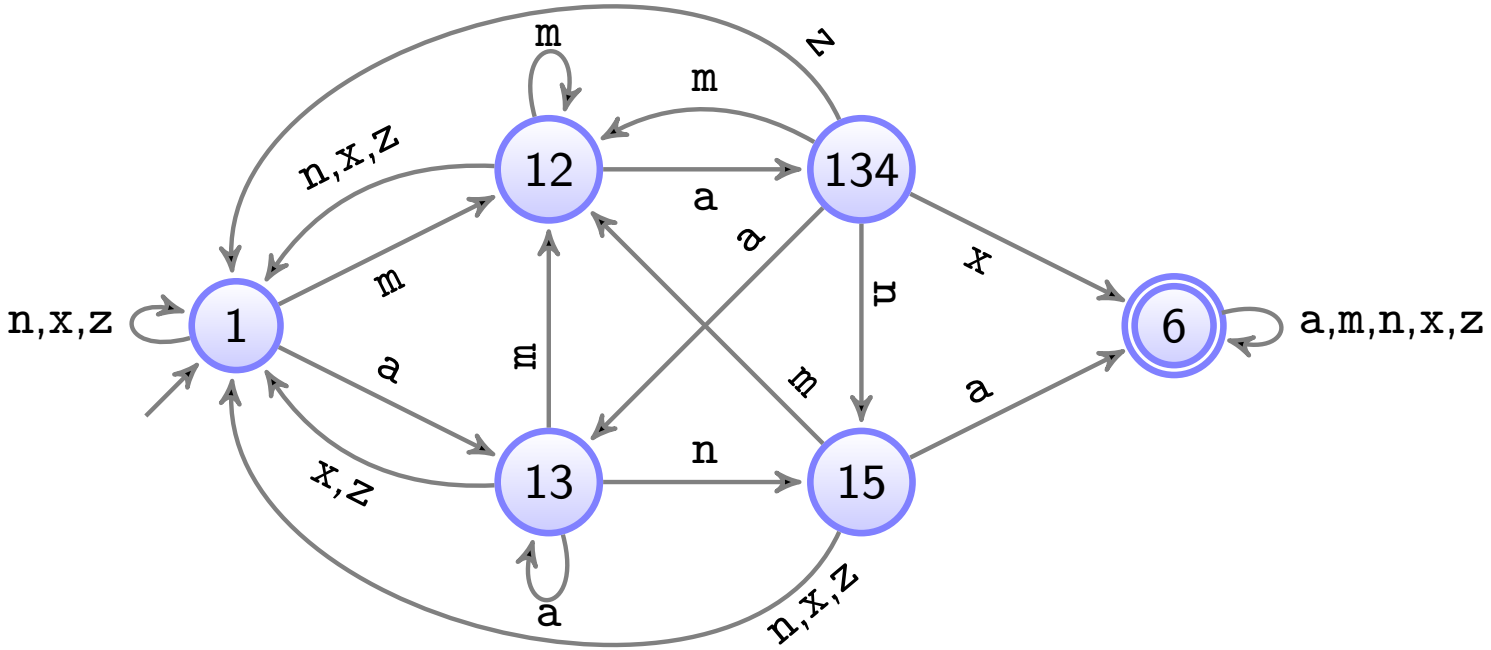
Gesucht: Automat zur Suche nach „max“ und „ana“ in einem Text

$\Sigma = \{\mathtt{a}, \mathtt{m}, \mathtt{n}, \mathtt{x}, \mathtt{z}\}$ (z ... Stellvertreter für b–l, o–w, y, z, ...)

NEA:



DEA:



Inhalt

Determinisierung

Gegeben: NEA $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$ mit $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$

Gesucht: DEA $\widehat{\mathcal{A}} = \langle \widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F} \rangle$ mit $\widehat{\delta}\colon \widehat{Q} \times \Sigma \mapsto \widehat{Q}$,
sodass $\mathcal{A}$ und $\widehat{\mathcal{A}}$ dieselbe Sprache akzeptieren.

Wir definieren den deterministischen Automaten $\widehat{\mathcal{A}}$ folgendermaßen:

- $\widehat{Q} = 2^Q$
- $\widehat{q_0} = \{q_0\}$
- $\widehat{F} = \begin{cases} \{\, \widehat{q} \in \widehat{Q} \mid \widehat{q} \cap F \neq \emptyset \,\} \cup \{\widehat{q_0}\} & \text{falls } \varepsilon \in \mathcal{L}(\mathcal{A}) \\ \{\, \widehat{q} \in \widehat{Q} \mid \widehat{q} \cap F \neq \emptyset \,\} & \text{sonst} \end{cases}$
- Für alle Zustände $\widehat{q} \in \widehat{Q}$ und alle Symbole $s \in \Sigma$:

  $\widehat{\delta}(\widehat{q}, s) = \{\, q' \in Q \mid \text{es gibt } q \in \widehat{q}, \text{ sodass } (q, s, q') \in \delta^* \,\}$

$\mathcal{A}$ und $\widehat{\mathcal{A}}$ sind äquivalent, d.h., sie akzeptieren dieselbe Sprache:
$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\widehat{\mathcal{A}})$.

Anmerkungen

Neue Endzustände: Ein neuer Zustand $\widehat{q}$ ist Endzustand, wenn

- ... seine Bezeichnung einen der alten Endzustände enthält, d.h., wenn $\widehat{q} \cap F \neq \emptyset$.
- ... es sich um den neuen Startzustand handelt und der alte Automat das Leerwort akzeptiert,
  d.h., wenn $\widehat{q} = \widehat{q_0}$ und $\varepsilon \in \mathcal{L}(\mathcal{A})$,
  d.h., wenn $\widehat{q} = \widehat{q_0}$ und $(q_0, \varepsilon, f) \in \delta^*$ für einen Endzustand $f \in F$.

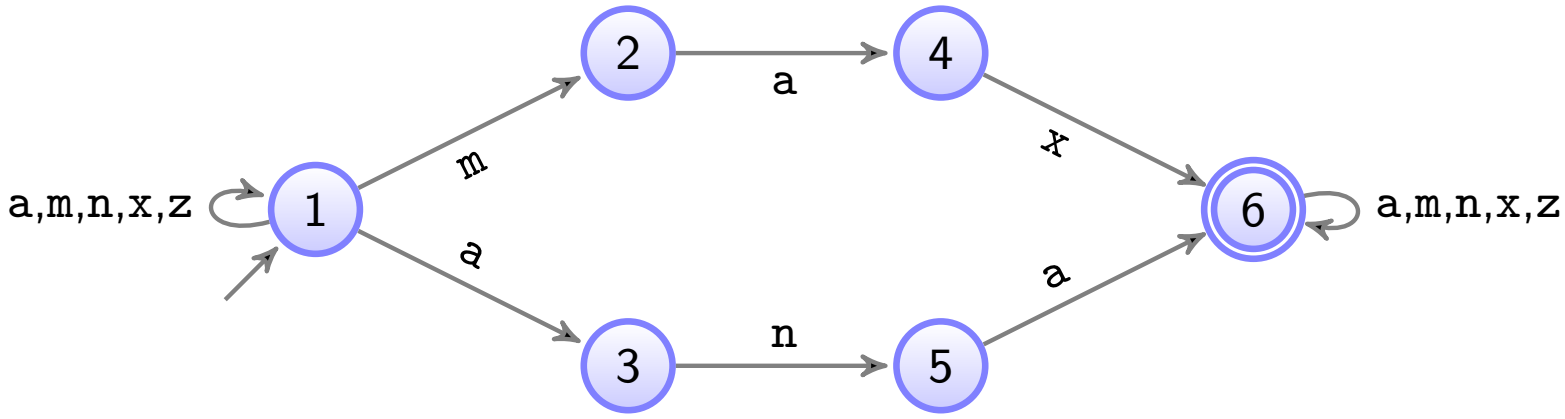
Neue Übergangsfunktion: Wegen

$$\widehat{\delta}(\widehat{q}, s) = \bigcup_{q \in \widehat{q}} \{\, q' \in Q \mid (q, s, q') \in \delta^* \,\} = \bigcup_{q \in \widehat{q}} \delta^*(q, s)$$

spart es Arbeit, zuerst $\delta^*(q, s)$ für alle $q \in Q$ und alle $s \in \Sigma$ zu berechnen. Danach müssen nur mehr die Zeilen, die $\widehat{q}$ entsprechen, vereinigt werden.

Neue Zustände: Betrachte nur jene Zustände aus 2^Q, die von $\widehat{q_0}$ aus erreichbar sind.

Beispiel: Suche nach Max und Ana



	δ^*	a	m	n	x	z
SZ	1	$\{1,3\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$
	2	$\{4\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
	3	$\{\}$	$\{\}$	$\{5\}$	$\{\}$	$\{\}$
	4	$\{\}$	$\{\}$	$\{\}$	$\{6\}$	$\{\}$
	5	$\{6\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
EZ	6	$\{6\}$	$\{6\}$	$\{6\}$	$\{6\}$	$\{6\}$

Identisch mit der Tabelle für δ, wenn es keine ε-Kanten gibt.

	δ^*	a	m	n	x	z
SZ	1	$\{1,3\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$
	2	$\{4\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
	3	$\{\}$	$\{\}$	$\{5\}$	$\{\}$	$\{\}$
	4	$\{\}$	$\{\}$	$\{\}$	$\{6\}$	$\{\}$
	5	$\{6\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
EZ	6	$\{6\}$	$\{6\}$	$\{6\}$	$\{6\}$	$\{6\}$

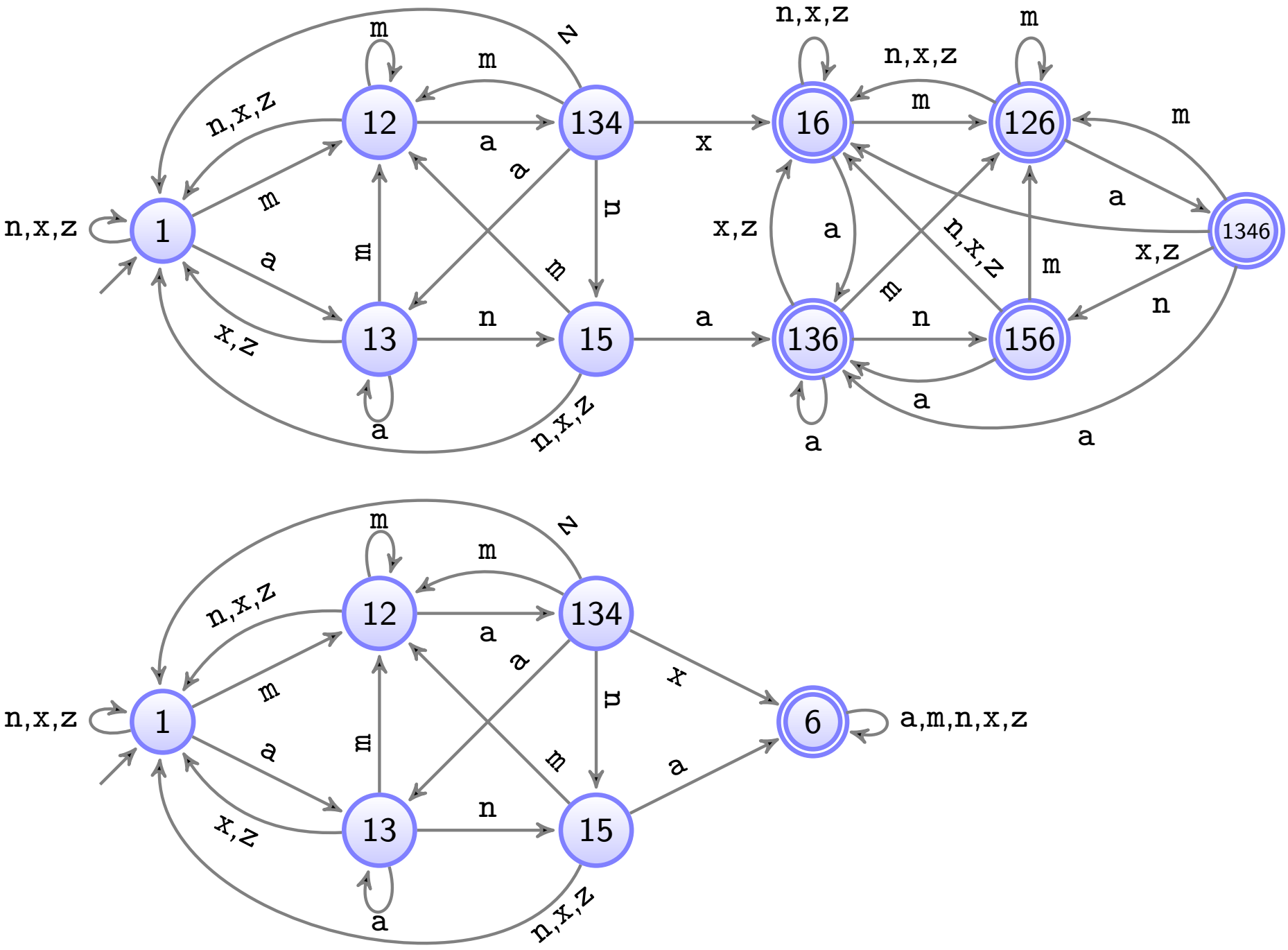
$$\widehat{Q} = 2^Q$$

$$\widehat{q_0} = \{q_0\}$$

$$\widehat{F} = \{\, \widehat{q} \in \widehat{Q} \mid \widehat{q} \cap F \neq \emptyset \,\}$$

$$\widehat{\delta}(\widehat{q}, s) = \{q' \mid q \in \widehat{q}, \; (q, s, q') \in \delta^*\}$$

	$\widehat{\delta}$	a	m	n	x	z
SZ	$\{1\}$	$\{1,3\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$
	$\{1,2\}$	$\{1,3,4\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$
	$\{1,3\}$	$\{1,3\}$	$\{1,2\}$	$\{1,5\}$	$\{1\}$	$\{1\}$
	$\{1,3,4\}$	$\{1,3\}$	$\{1,2\}$	$\{1,5\}$	$\{1,6\}$	$\{1\}$
	$\{1,5\}$	$\{1,3,6\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$
EZ	$\{1,6\}$	$\{1,3,6\}$	$\{1,2,6\}$	$\{1,6\}$	$\{1,6\}$	$\{1,6\}$
EZ	$\{1,3,6\}$	$\{1,3,6\}$	$\{1,2,6\}$	$\{1,5,6\}$	$\{1,6\}$	$\{1,6\}$
EZ	$\{1,2,6\}$	$\{1,3,4,6\}$	$\{1,2,6\}$	$\{1,6\}$	$\{1,6\}$	$\{1,6\}$
EZ	$\{1,5,6\}$	$\{1,3,6\}$	$\{1,2,6\}$	$\{1,6\}$	$\{1,6\}$	$\{1,6\}$
EZ	$\{1,3,4,6\}$	$\{1,3,6\}$	$\{1,2,6\}$	$\{1,5,6\}$	$\{1,6\}$	$\{1,6\}$


n,x,z
1
12
134
13
15
16
126
1346
136
156
6
m
a
n
x
z
x,z
n,x,z
a,m,n,x,z

Inhalt

Transducer

Endlicher Transducer

... wird beschrieben durch ein 6-Tupel $\mathcal{A} = \langle Q, \Sigma, \Gamma, \delta, I, F \rangle$, wobei

- Q, Σ, F ... siehe DEAs
- Γ ... Ausgabealphabet (*output alphabet*)
- $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$ ... Übergangsrelation
- $I \subseteq Q$ ... Anfangszustände

Erweiterte Übergangsrelation $\delta^* \subseteq Q \times \Sigma^* \times \Gamma^* \times Q$

δ^* ist die kleinste Menge mit folgenden Eigenschaften:

- $(q, \varepsilon, \varepsilon, q) \in \delta^*$ für alle $q \in Q$
- $(q_1, w, w', q_2) \in \delta^*,\ (q_2, s, s', q_3) \in \delta \quad \Longrightarrow \quad (q_1, ws, w's', q_3) \in \delta^*$

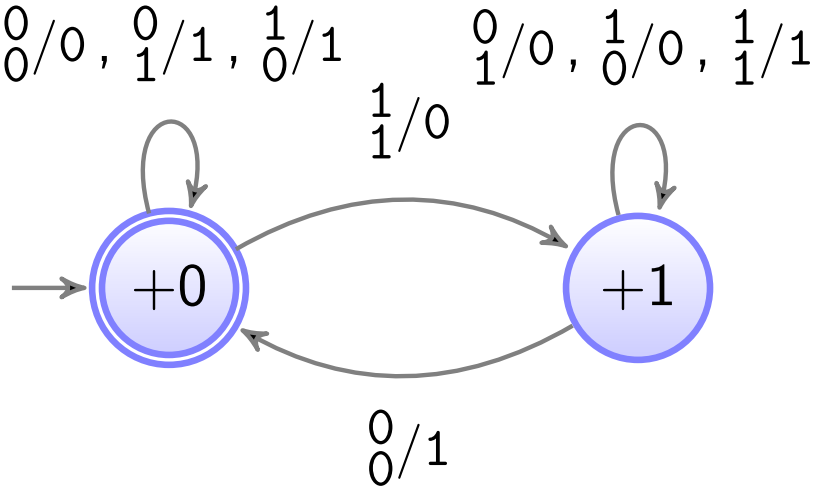
Übersetzungsrelation $[\mathcal{A}] \subseteq \Sigma^* \times \Gamma^*$

$[\mathcal{A}] = \{\, (w, w') \in \Sigma^* \times \Gamma^* \mid (i, w, w', f) \in \delta^* \text{ für ein } i \in I \text{ und ein } f \in F \,\}$

Beispiel: Binäraddition von rechts nach links

$$\begin{array}{cccccc|l} 0 & 0 & 1 & 0 & 1 & 1 & = 11_{10} \\ 0 & 0 & 0 & 1 & 0 & 1 & = 5_{10} \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & = 16_{10} \end{array}$$

$\longleftarrow$



$\mathcal{A} = \langle \{+0, +1\}, \{\binom{0}{0}, \binom{0}{1}, \binom{1}{0}, \binom{1}{1}\}, \{0, 1\}, \delta, \{+0\}, \{+0\} \rangle$, wobei

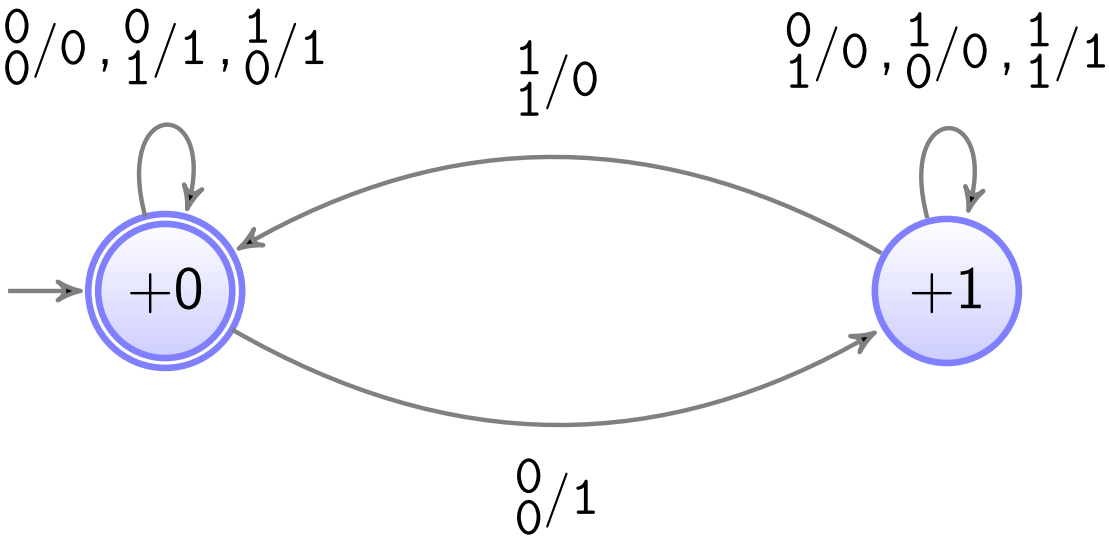
δ	$\binom{0}{0}$	$\binom{0}{1}$	$\binom{1}{0}$	$\binom{1}{1}$
$+0$	$\{(0, +0)\}$	$\{(1, +0)\}$	$\{(1, +0)\}$	$\{(0, +1)\}$
$+1$	$\{(1, +0)\}$	$\{(0, +1)\}$	$\{(0, +1)\}$	$\{(1, +1)\}$

$$[\mathcal{A}] = \{ (\varepsilon, \varepsilon), (\tbinom{0}{0}, 0), (\tbinom{0}{1}, 1), (\tbinom{1}{0}, 1), (\tbinom{0\,0}{0\,0}, 00), (\tbinom{0\,0}{0\,1}, 01), \ldots, (\tbinom{1\,0}{1\,0}, 01), \ldots, (\tbinom{1\,0\,1\,0}{1\,0\,1\,0}, 0101), \ldots \}$$

Binäraddition von links nach rechts

$$\begin{array}{cccccc|l} 0 & 0 & 1 & 0 & 1 & 1 & = 11_{10} \\ 0 & 0 & 0 & 1 & 0 & 1 & = 5_{10} \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & = 16_{10} \end{array}$$

$\longrightarrow$



+0 ... kein Übertrag
+1 ... Übertrag

Achtung, Indeterminismus! Zustand „+0“ besitzt bei Eingabe $\frac{0}{0}$ zwei Folgezustände, ebenso Zustand „+1“ bei Eingabe $\frac{1}{1}$.

Inhalt

Mealy-Automat

... wird beschrieben durch ein 6-Tupel $\mathcal{A} = \langle Q, \Sigma, \Gamma, \delta, \gamma, q_0 \rangle$, wobei

- Q, Σ, δ, q_0 ... siehe DEAs
- Γ ... Ausgabealphabet (*output alphabet*)
- $\gamma\colon Q \times \Sigma \mapsto \Gamma$ ... Ausgabefunktion (*output function*)

Erweiterte Übergangsfunktion $\delta^*\colon Q \times \Sigma^* \mapsto Q$ siehe DEA.

Erweiterte Ausgabefunktion $\gamma^*\colon Q \times \Sigma^* \mapsto \Gamma^*$

$\gamma^*(q, \varepsilon) = \varepsilon$ für alle $q \in Q$, $s \in \Sigma$, $w \in \Sigma^*$

$\gamma^*(q, sw) = \gamma(q, s) \cdot \gamma^*(\delta(q, s), w)$

Übersetzungsfunktion $[\mathcal{A}]\colon \Sigma^* \mapsto \Gamma^*$

$[\mathcal{A}](w) = \gamma^*(q_0, w)$

Mealy-Automaten sind ein Spezialfall von Transducern:

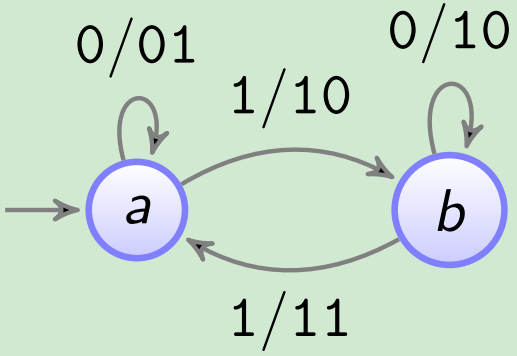
- Nur ein Anfangszustand: $I = \{q_0\}$
- Die Übergangsrelation ist deterministisch:
  - Der Folgezustand $\delta(q, s)$ ist eindeutig durch q und s bestimmt.
  - Keine ε-Übergänge
- Relationstupel: $(q, s, \gamma(q, s), \delta(q, s))$
- Alle Zustände sind Endzustände: $F = Q$

$(1 : 2)$-$(0, 1)$-RLL-Encoder als Mealy-Automat

$\mathcal{A} = \langle \{a, b\}, \{0, 1\}, \{01, 10, 11\}, \delta, \gamma, a \rangle$

δ	0	1
a	a	b
b	b	a

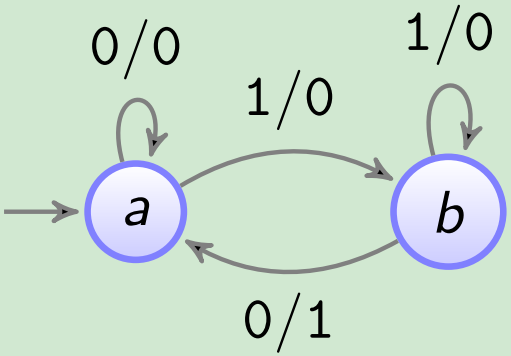
γ	0	1
a	01	10
b	10	11



w:	ε	0	1	00	10	01	11	000	100	$\cdots$
$[\mathcal{A}](w)$:	ε	01	10	0101	1010	0110	1011	010101	101010	$\cdots$

Detektor für fallende Flanken

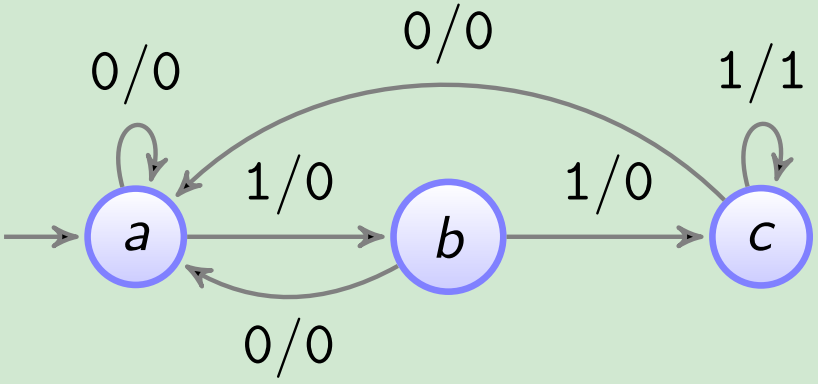
Ausgabe 1, wenn in der Eingabe ein Wechsel von 1 auf 0 stattfindet.



w:	$00110001010\cdots$
$[\mathcal{A}](w)$:	$00001000101\cdots0$

Detektor für 111-Blöcke

Ausgabe 1, wenn in der Eingabe drei 1er aufeinander folgen.



w:	$00110111110101110\cdots$
$[\mathcal{A}](w)$:	$00000001110000010\cdots$

Inhalt

Moore-Automat

... wird beschrieben durch ein 6-Tupel $\mathcal{A} = \langle Q, \Sigma, \Gamma, \delta, \gamma, q_0 \rangle$, wobei

- Q, Σ, δ, q_0 ... siehe DEAs
- Γ ... Ausgabealphabet (*output alphabet*)
- $\gamma\colon Q \mapsto \Gamma$ ... Ausgabefunktion (*output function*)

(Mealy: $\gamma\colon Q \times \Sigma \mapsto \Gamma$)

Erweiterte Übergangsfunktion $\delta^*\colon Q \times \Sigma^* \mapsto Q$ siehe DEA.

Erweiterte Ausgabefunktion $\gamma^*\colon Q \times \Sigma^* \mapsto \Gamma^*$

$\gamma^*(q, \varepsilon) = \varepsilon$ für alle $q \in Q$, $s \in \Sigma$, $w \in \Sigma^*$

$\gamma^*(q, sw) = \gamma(\delta(q, s)) \cdot \gamma^*(\delta(q, s), w)$

(Mealy: $\gamma^*(q, sw) = \gamma(q, s) \cdot \gamma^*(\delta(q, s), w)$)

Übersetzungsfunktion $[\mathcal{A}]\colon \Sigma^* \mapsto \Gamma^*$

$[\mathcal{A}](w) = \gamma^*(q_0, w)$

Moore-Automaten sind ein Spezialfall von Transducern:

- Nur ein Anfangszustand: $I = \{q_0\}$
- Alle Übergange in einen Zustand geben dasselbe Symbol aus.
- Die Übergangsrelation ist deterministisch:
  - Der Folgezustand $\delta(q, s)$ ist eindeutig durch q und s bestimmt.
  - Keine ε-Übergänge
- Relationstupel: $(q, s, \gamma(\delta(q, s)), \delta(q, s))$
- Alle Zustände sind Endzustände: $F = Q$

Vergleich von Moore- und Mealy-Automaten

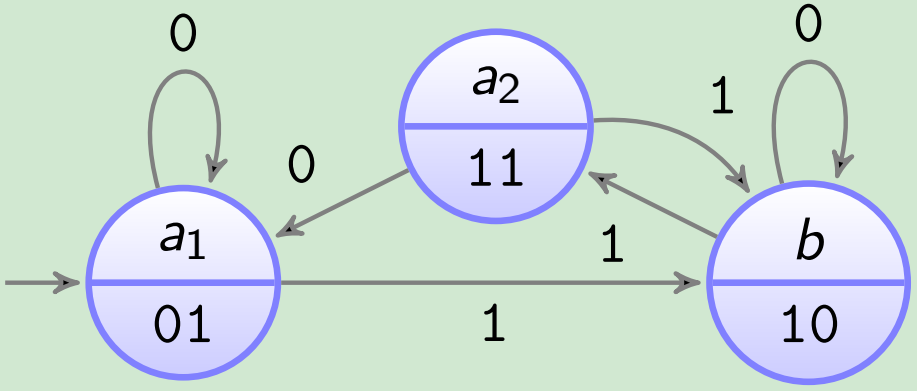
- Die Ausgabe erfolgt
  - ... bei Moore-Automaten beim „Betreten“ des Zustandes, unabhängig von der Herkunft. Die Ausgabe hängt nur vom Zustand ab.
  - ... bei Mealy-Automaten beim Zustandswechsel, der durch Ursprungszustand und Eingabe festgelegt ist.
- Moore- und Mealy-Automaten besitzen dieselbe Ausdrucksstärke, sind aber schwächer als Transducer.
- Moore-Automaten haben in der Regel mehr Zustände als Mealy-Automaten.

$(1:2)$-$(0,1)$-RLL-Encoder als Moore-Automat

$\mathcal{A} = \langle \{a_1, a_2, b\}, \{0,1\}, \{01, 10, 11\}, \delta, \gamma, a \rangle$

δ	0	1
a_1	a_1	b
a_2	a_1	b
b	b	a_2

γ	
a_1	01
a_2	11
b	10

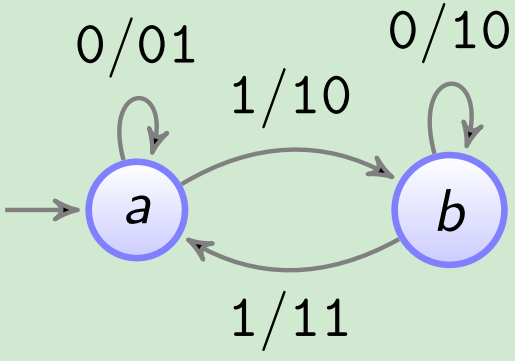


w:	ε	0	1	00	10	01	11	000	100	$\cdots$
$[\mathcal{A}](w)$:	ε	01	10	0101	1010	0110	1011	010101	101010	$\cdots$

Zum Vergleich: Mealy-Automat

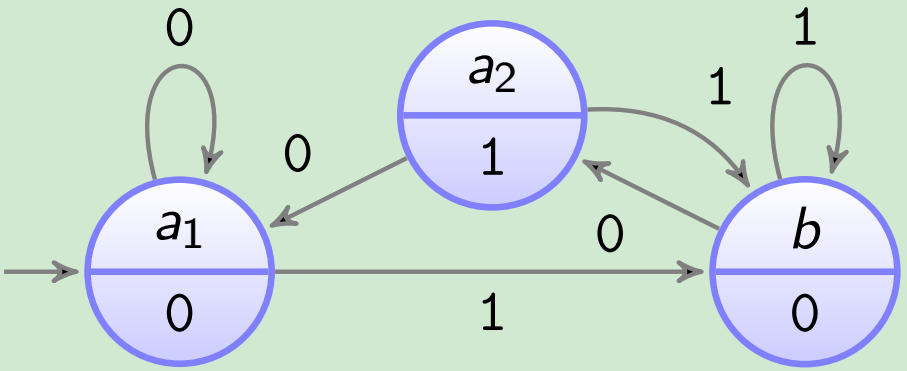
δ	0	1
a	a	b
b	b	a

γ	0	1
a	01	10
b	10	11



Detektor für fallende Flanken

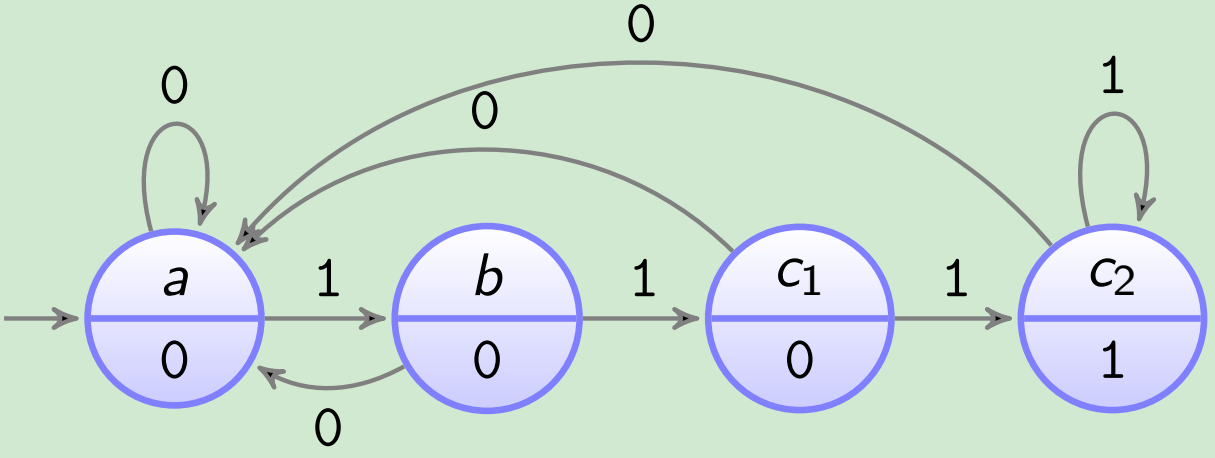
Ausgabe 1, wenn in der Eingabe ein Wechsel von 1 auf 0 stattfindet.



w:	$00110001010\cdots$
$[\mathcal{A}](w)$:	$00001000101\cdots0$

Detektor für 111-Blöcke

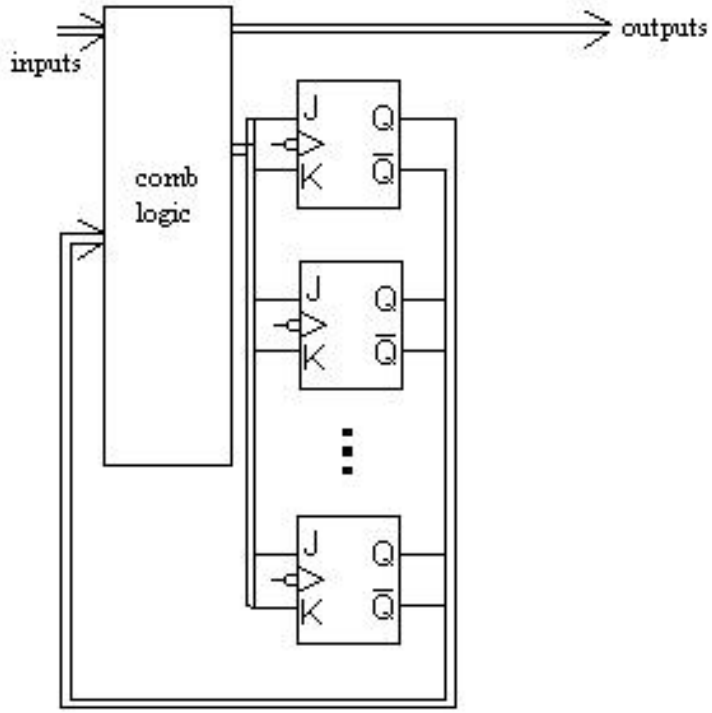
Ausgabe 1, wenn in der Eingabe drei 1er aufeinander folgen.



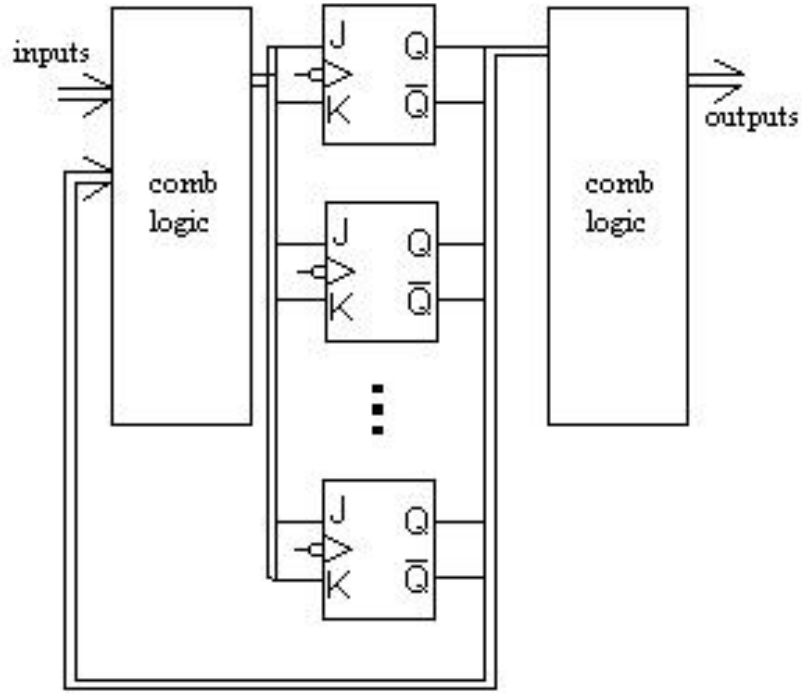
w:	$00110111110101110\cdots$
$[\mathcal{A}](w)$:	$00000001110000010\cdots$

Relevanz von Mealy- und Moore-Automaten

Mealy-Schaltwerk



Moore-Schaltwerk

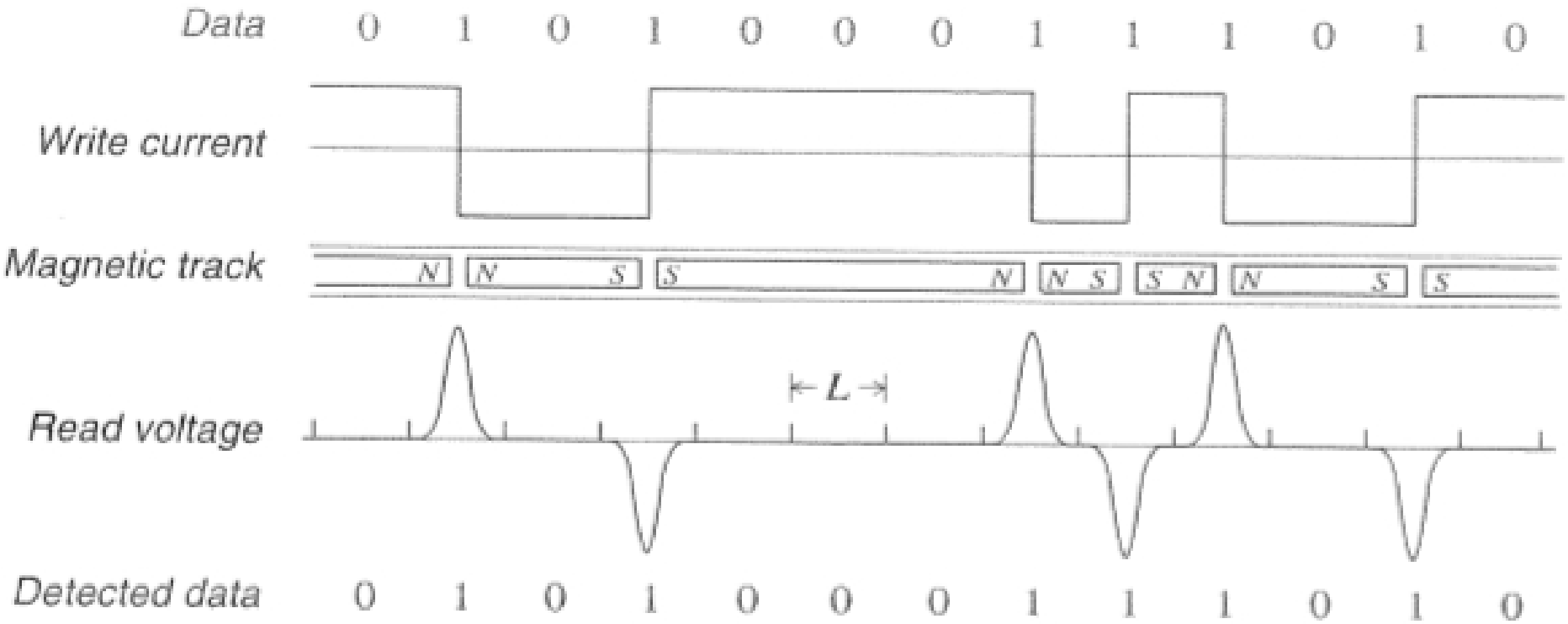


- „inputs“ stammen aus dem Eingabealphabet Σ
- „outputs“ stammen aus dem Ausgabealphabet Γ
- Flip-Flops speichern den Zustand aus Q; Reset: Anfangszustand q_0
- „combination logic“: realisiert die Übergangsfunktionen δ und γ.

Graphiken: Ken Reid, Indiana University – Purdue University Indianapolis

RLL (run-length limited) Codes

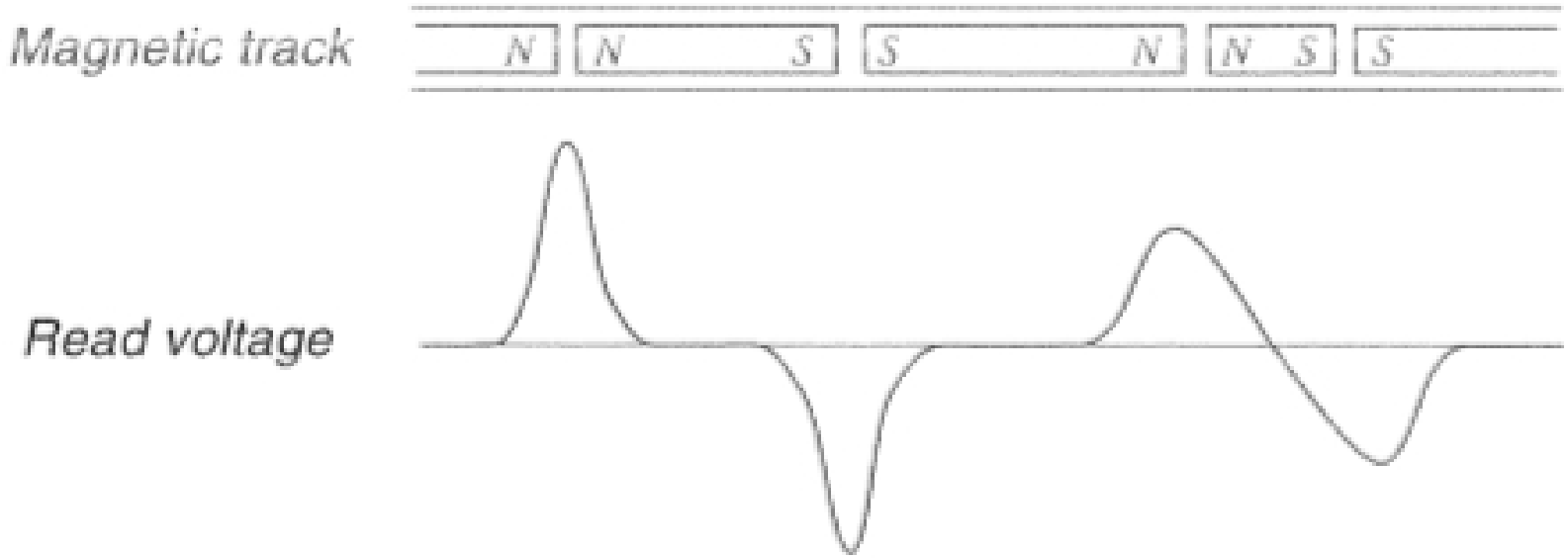
- Daten werden auf Festplatten, CDs und DVDs binär kodiert.
- Binär-1: Magnetisierungswechsel bzw. Vertiefung („pit“)



Graphik aus: D.Lind, B.Marcus: An Introduction to Symbolic Dynamics and Coding, Cambridge Unversity Press 1995.

Probleme:

- 1er zu weit auseinander: Synchronisation geht verloren.
- 1er zu dicht: werden ununterscheidbar bzw. schwächen sich ab.



Graphik aus: D.Lind, B.Marcus: An Introduction to Symbolic Dynamics and Coding, Cambridge Unversity Press 1995.

Einfache Lösung:

- Füge nach jedem Datenbit ein zusätzliches Synchronisationsbit ein.
- Wähle Bitabstand so groß, dass 1er-Folgen richtig erkannt werden.

$(m : n)$-(d, k)-RLL-Codes

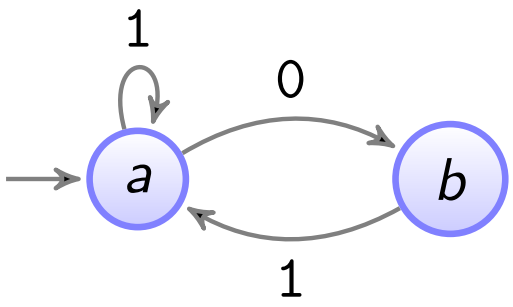
Nachteil der einfachen Lösung: Benötigt mehr Platz als notwendig.

Idee: Nütze die „natürlichen“ 0er-Strecken und 1er im Datenstrom.
m Datenbits werden so in n Codebits übersetzt, dass zwischen zwei 1ern mindestens d und höchstens k 0er auftreten.

- Existiert nur für bestimmte Werte von m, n, d, k
- Ziel: minimiere n/m, maximiere d

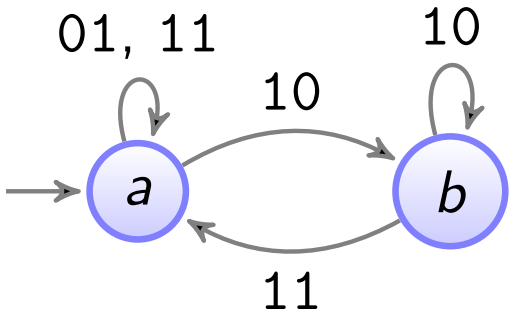
DVD: $(8 : 16)$-$(2, 10)$-RLL-Code

$(1:2)$-$(0,1)$-RLL-Code: 2 Codebits/Datenbit, max. ein 0er zwischen 1ern.



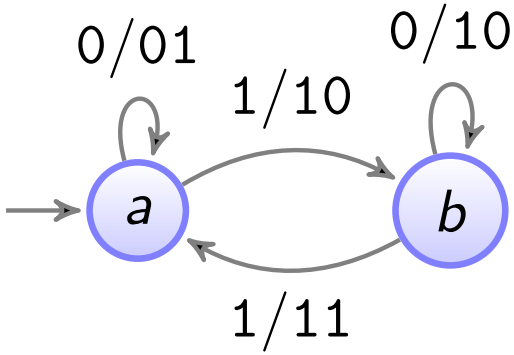
Zustand a: zwei Codeworte für zwei Datenbits – reicht.
Zustand b: ein Codewort für zwei Datenbits – reicht nicht.

„Potenzieren“ des Automaten:

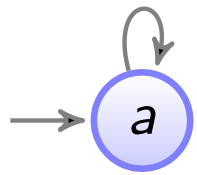


Zustand a: drei Codeworte für zwei Datenbits – reicht.
Zustand b: zwei Codeworte für zwei Datenbits – reicht.

Codierungsautomat (Mealy-Automat):



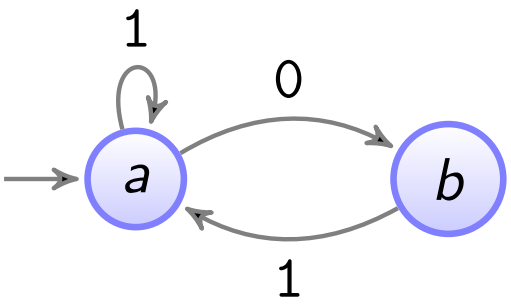
oder



oder . . .

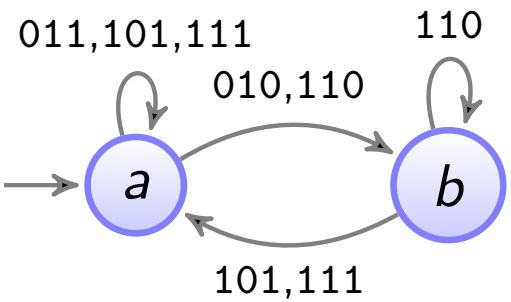
10010 $\Rightarrow$ 10 10 10 11 01 oder 11 01 01 11 01 oder . . .

$(2:3)$-$(0,1)$-RLL-Code: 3 Code-/2 Datenbits, max. ein 0er zwischen 1ern.



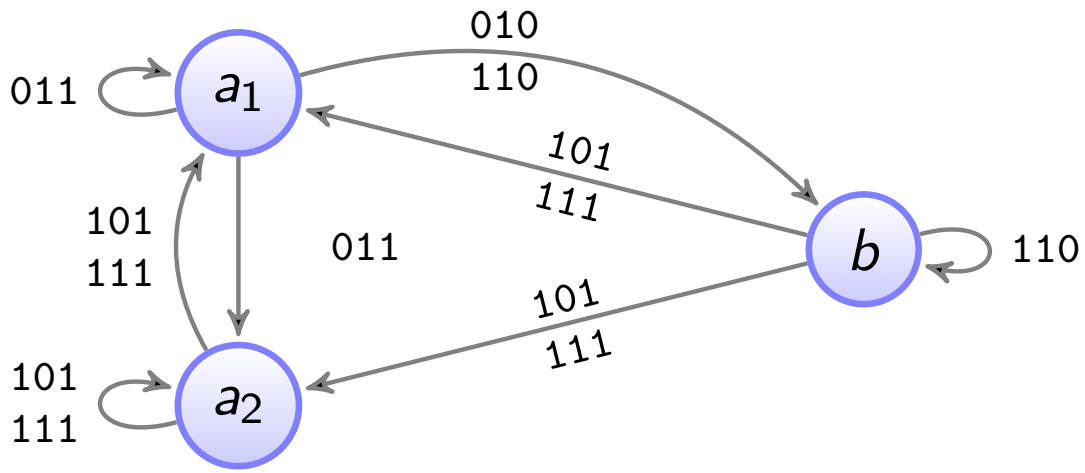
Zustand a: zwei Code- für vier Datenworte – reicht nicht.
Zustand b: ein Codewort für vier Datenworte – reicht nicht.

Dritte Potenz des Automaten:

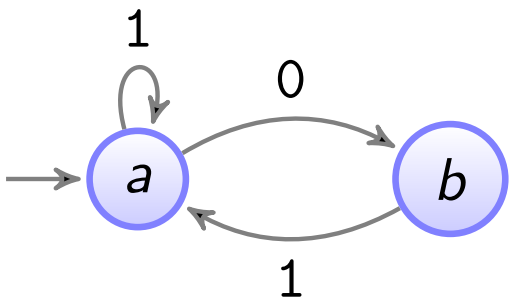


Zustand a: fünf Code- für vier Datenworte – reicht.
Zustand b: drei Code- für vier Datenworte – reicht nicht.

Teilung von Zustand a:

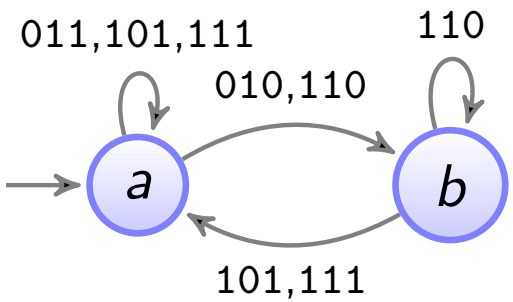


$(2:3)$-$(0,1)$-RLL-Code: 3 Code-/2 Datenbits, max. ein 0er zwischen 1ern.



Zustand a: zwei Code- für vier Datenworte – reicht nicht.
Zustand b: ein Codewort für vier Datenworte – reicht nicht.

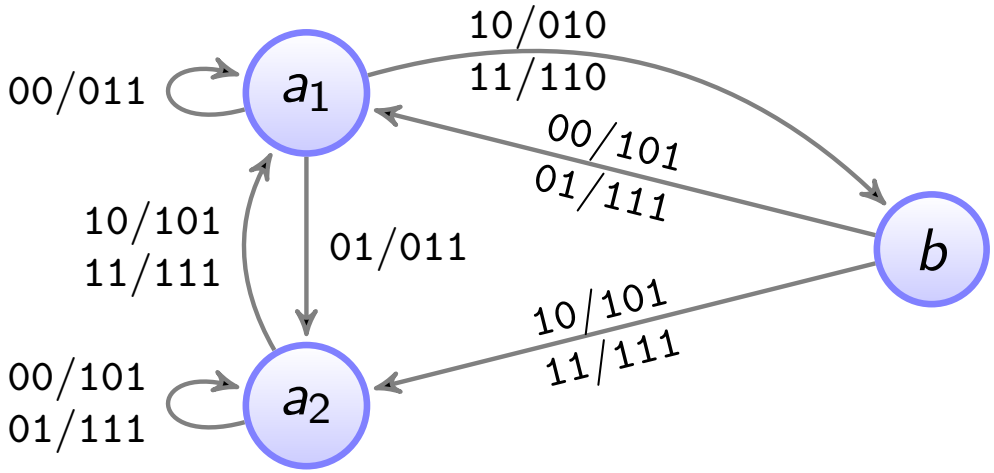
Dritte Potenz des Automaten:



Zustand a: fünf Code- für vier Datenworte – reicht.
Zustand b: drei Code- für vier Datenworte – reicht nicht.

Teilung von Zustand a:

Mealy-Automat



Inhalt

Büchi-Automaten

Deterministischer Büchi-Automat

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$, wobei

- Q, Σ, δ, q_0, F wie bei DEAs definiert sind.

Σ^ω ... Menge aller unendlichen Wörter ($= \omega$-Wörter) über Σ

Akzeptanz von Wörtern

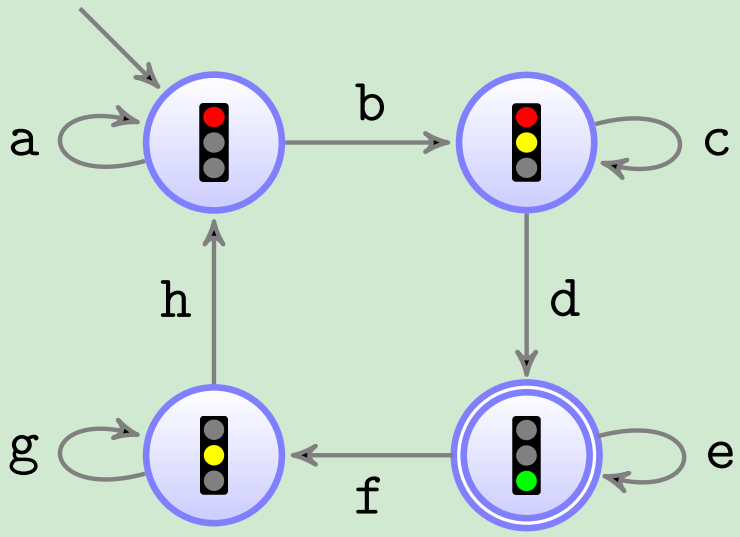
Ein deterministischer Büchi-Automat $\mathcal{A}$ akzeptiert ein Wort $s_1 s_2 s_3 \cdots \in \Sigma^\omega$, wenn es Zustände $q_0, q_1, q_2, q_3, \cdots \in Q$ gibt, sodass

- $q_0 \in Q$ der Startzustand ist,
- $\delta(q_{i-1}, s_i) = q_i$ für alle $i \in \mathbb{N}$ gilt und
- es unendlich viele i gibt, sodass q_i ein Endzustand ist ($q_i \in F$).

Akzeptierte/Generierte Sprache

$\mathcal{L}(\mathcal{A}) = \{ w \in \Sigma^\omega \mid w \text{ wird von } \mathcal{A} \text{ akzeptiert} \}$

Faire Verkehrsampel



Der Automat akzeptiert genau jene Wörter aus $\{\mathtt{a}, \ldots, \mathtt{h}\}^\omega$, bei denen es immer wieder grün wird.

Nichtdeterministischer Büchi-Automat

... wird beschrieben durch ein 5-Tupel $\mathcal{A} = \langle Q, \Sigma, \delta, I, F \rangle$, wobei

- Q, Σ, F ... siehe DEA
- $I \subseteq Q$ ... Anfangszustände
- $\delta \subseteq Q \times \Sigma \times Q$ ... Übergangsrelation

Akzeptanz von Wörtern

Ein nichtdeterministischer Büchi-Automat $\mathcal{A}$ akzeptiert ein Wort $s_1 s_2 s_3 \cdots \in \Sigma^\omega$, wenn es Zustände $q_0, q_1, q_2, q_3, \cdots \in Q$ gibt, sodass

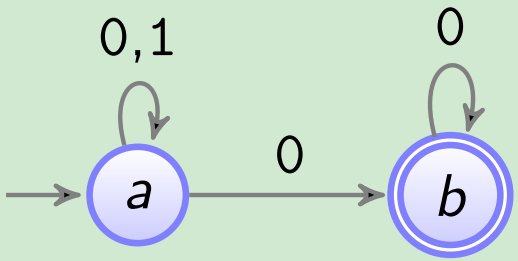
- $q_0 \in Q$ ein Startzustand ist ($q_0 \in I$),
- $(q_{i-1}, s_i, q_i) \in \delta$ für alle $i \in \mathbb{N}$ gilt und
- es unendlich viele i gibt, sodass q_i ein Endzustand ist ($q_i \in F$).

Akzeptierte/Generierte Sprache

$\mathcal{L}(\mathcal{A}) = \{ w \in \Sigma^\omega \mid w \text{ wird von } \mathcal{A} \text{ akzeptiert} \}$

Nur endlich viele 1er

Gesucht: Ein Büchi-Automat, der genau jene ω-Wörter über $\{0, 1\}$ akzeptiert, die nur eine endliche Anzahl an 1ern enthalten.



Kein deterministischer Büchi-Automaten akzeptiert diese Sprache.
$\Longrightarrow$ Nichtdeterministische Büchi-Automaten sind ausdrucksstärker als deterministische!

Modellierung mit endlichen Automaten

Vorgangsweise:

1. Was sind die Zustände des Systems? Wieviele sind notwendig? Zustandsbezeichnungen?
2. Startzustand? Endzustände?
3. Was sind die Aktionen/Eingaben, die zu Zustandsübergängen führen? Bezeichnung?
4. Was sind die Aktionen/Ausgaben, die bei Zustandsübergängen stattfinden? Bezeichnung?
5. Lege für jeden Zustand und jede Eingabe die Folgezustände und die Ausgaben fest.