Grundbegriffe der Informatik

Einheit 14: Endliche Automaten

Thomas Worsch

Karlsruher Institut für Technologie, Fakultät für Informatik

Wintersemester 2009/2010

Überblick

Erstes Beispiel: ein Getränkeautomat

Mealy-Automaten

Moore-Automaten

Spezialfall: endliche Akzeptoren

Überblick

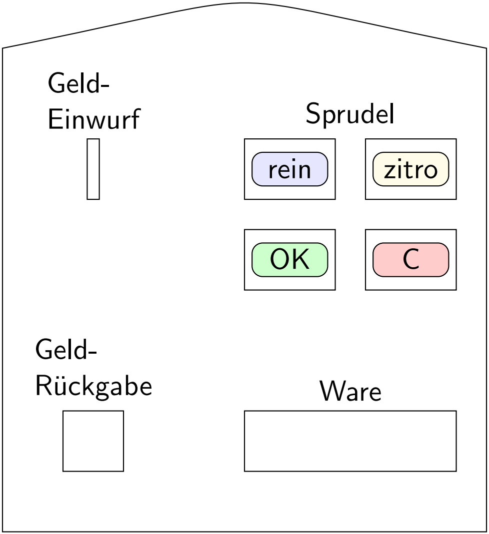
Erstes Beispiel: ein Getränkeautomat

Mealy-Automaten

Moore-Automaten

Spezialfall: endliche Akzeptoren

Ein primitiver Getränkeautomat



Getränkeautomat: umgangssprachliche Beschreibung

- nur 1-Euro-Stücke einzuwerfen
- vier Tasten
  - zwei Wahltasten für Mineralwasser rein und Zitronensprudel zitro
  - OK-Taste und Abbruch-Taste C
- Jede Flasche kostet 1 Euro.
- Guthaben von 1 Euro kann gespeichert werden
- weitere Euro-Stücke werden sofort wieder ausgegeben
- Drücken von rein/zitro: letzter wird Wunsch gespeichert
- C -Taste: bereits eingeworfener Euro wird zurückgegeben und kein Getränkewunsch mehr gespeichert.
- OK -Taste
  - ignoriert, solange noch kein Euro eingeworfen oder keine Getränkesorte ausgewählt
  - andernfalls das gewünschte Getränk ausgeworfen

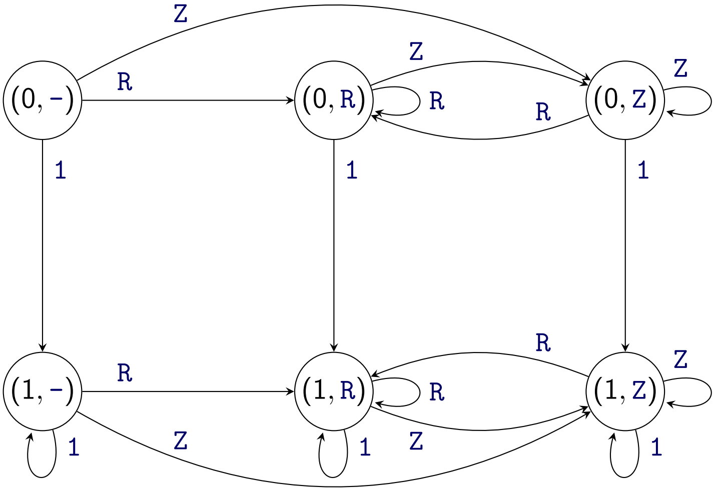
Formalisierung des Getränkeautomaten: Zustände

- Automat muss zwischen den Eingaben, die sein Verhalten beeinflussen können (Geldeinwürfe und Getränkewahl), gewisse Nachrichten speichern.
- und zwar
  - Wurde schon ein 1-Euro-Stück eingeworfen?
  - Wurde schon ein Getränk ausgewählt?
  - Wenn ja: welches?
- Modellierung: durch Paare (x, y)
  - Komponente $x \in \{0, 1\}$: schon eingeworfener Geldbetrag
  - Komponente $y \in \{\texttt{-}, \texttt{R}, \texttt{Z}\}$: Getränkewahl
  - Zustandsmenge $Z = \{0, 1\} \times \{\texttt{-}, \texttt{R}, \texttt{Z}\}$

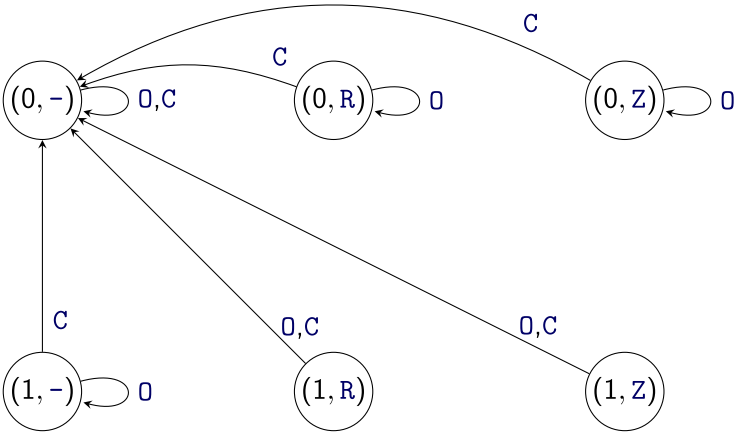
- erster wesentlicher Aspekt jedes Automaten:
  Eingaben (Einflüsse „von außen“) führen zu *Zustandsänderungen*.
- Eingaben hier:
  - Einwurf eines Euros
  - Drücken einer der vier Tasten
    - ignoriere gleichzeitige Tastendrücke („Abstraktion“)
- Modellierung der Eingaben: Symbole `1`, `R`, `Z`, `C` und `O`
- Eingabealphabet $X = \{\mathtt{1}, \mathtt{R}, \mathtt{Z}, \mathtt{C}, \mathtt{O}\}$

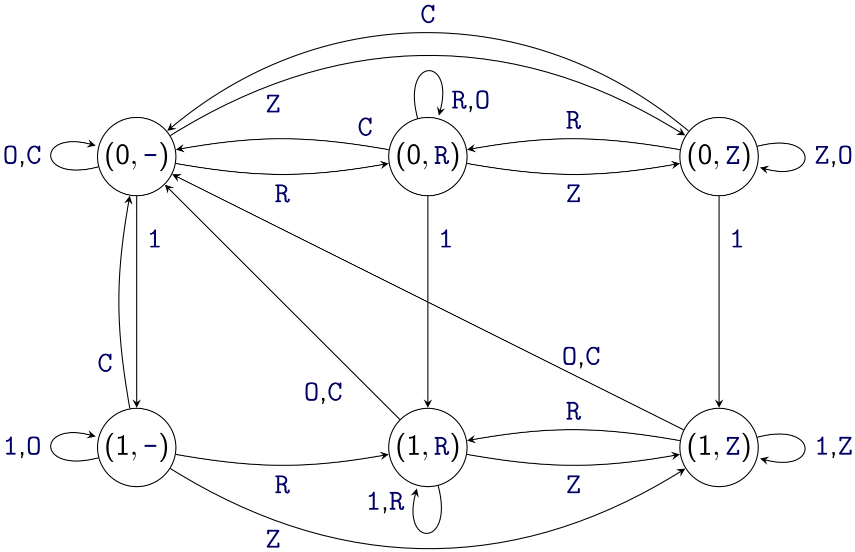
Formalisierung des Getränkeautomaten: Zustandsübergänge (1)

- Zustandsübergang
  - abhängig von aktuellem Zustand $z \in Z$ und aktuellem Eingabesymbol $x \in X$
  - z und x legen eindeutig den neuen Zustand fest.
    - also *immer* und *eindeutig*
    - jedenfalls bei dem Getränkeautomaten
  - Formalisierung: Zustandsüberführungsfunktion $f : Z \times X \to Z$
  - oft in Darstellung als Graph spezifiziert


(0, -)
(0, R)
(0, Z)
(1, -)
(1, R)
(1, Z)
R
Z
1

Formalisierung des Getränkeautomaten: Zustandsübergänge (2b)

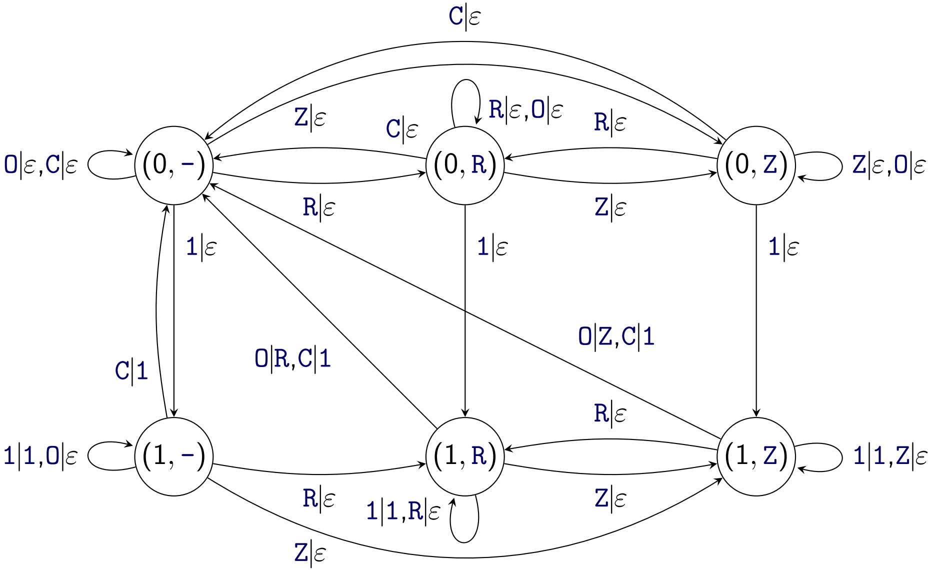



C
Z
R,O
C
R
O,C
(0, -)
(0, R)
(0, Z)
Z,O
R
Z
1
1
1
C
O,C
O,C
R
1,O
(1, -)
(1, R)
(1, Z)
1,Z
R
1,R
Z
Z

Formalisierung des Getränkeautomaten: Ausgaben (1)

- zweiter wesentlicher Aspekt jedes Automaten: Ausgaben
  - wozu sollte man ihn sonst laufen lassen
  - jedenfalls ab und zu
- Ausgaben hier:
  - Euro Rückgeld
  - gewählte Flasche
- Ausgabealphabet $Y = \{\mathtt{1}, \mathtt{R}, \mathtt{Z}\}$
- Formalisierung der Ausgaben
  - abhängig von aktuellem Zustand $z \in Z$ und aktuellem Eingabesymbol $x \in X$
  - z und x legen eindeutig die Ausgabe fest.
    - also *immer* und *eindeutig*
    - jedenfalls bei dem Getränkeautomaten
    - im allgemeinen *Wörter* über Y
  - Formalisierung: Ausgabefunktion $g : Z \times X \to Y^*$
    - Funktionswert ε: „keine Ausgabe"
- Auch g üblicherweise in den Zustandsübergangsdiagrammen mit angegeben in der Form $x|g(z,x)$

Formalisierung des Getränkeautomaten: Ausgaben (2)



Was ist wichtig

Das sollten Sie mitnehmen:

- alles endlich
- alles endlich beschreibbar
- im Beispiel Getränkeautomat:
  - aktueller Zustand und aktuelle Eingabe legen immer und eindeutig fest:
    - nächsten Zustand
    - Ausgabe

Das sollten Sie üben:

- Zustandsdiagramm lesen

Überblick

Ein *(endlicher) Mealy-Automat* ist festgelegt durch

- eine endliche *Zustandsmenge* Z,
- einen *Anfangszustand* $z_0 \in Z$,
- ein *Eingabealphabet* X,
- eine *Zustandsüberführungsfunktion* $f : Z \times X \to Z$,
- ein *Ausgabealphabet* Y,
- eine *Ausgabefunktion* $g : Z \times X \to Y^*$

Darstellung als Graph:

- Knoten: Zustände
- Kanten: $z \xrightarrow{x|g(z,x)} f(z,x)$
- Anfangszustand: in der Darstellung durch kleinen Pfeil gekennzeichnet: $\longrightarrow z$

- nach Eingabe eines ganzen Wortes $w \in X^*$:
  - Was ist der erreichte Zustand?
  - Welches sind alle durchlaufenen Zustände?
- definiere passende Funktionen f^* und f^{**}
  1. Stern: zweite Argument ist ganzes Wort von Eingabesymbolen
  2. Stern: interessieren uns für alle durchlaufenen Zuständen Zuständen
- definiere $f^* : Z \times X^* \to Z$:

$$f^*(z, \varepsilon) = z$$

$$\forall w \in X^* : \forall x \in X : \quad f^*(z, wx) = f(f^*(z, w), x)$$

- alternativ:

$$\bar{f}^*(z, \varepsilon) = z$$

$$\forall w \in X^* : \forall x \in X : \quad \bar{f}^*(z, xw) = \bar{f}^*(f(z, x), w)$$

- beide Definitionen liefern das Gleiche: $f^* = \bar{f}^*$.
  Man nimmt die bequemere.

Verallgemeinerte Zustandsübergangsfunktionen (1)

- nach Eingabe eines ganzen Wortes $w \in X^*$:
  - Was ist der erreichte Zustand?
  - Welches sind alle durchlaufenen Zustände?
- definiere passende Funktionen f^* und f^{**}
  1. Stern: zweite Argument ist ganzes Wort von Eingabesymbolen
  2. Stern: interessieren uns für alle durchlaufenen Zuständen Zuständen
- definiere $f^* : Z \times X^* \to Z$:

$$\begin{aligned} f^*(z, \varepsilon) &= z \\ \forall w \in X^* : \forall x \in X : \quad f^*(z, wx) &= f(f^*(z, w), x) \end{aligned}$$

- alternativ:

$$\begin{aligned} \bar{f}^*(z, \varepsilon) &= z \\ \forall w \in X^* : \forall x \in X : \quad \bar{f}^*(z, xw) &= \bar{f}^*(f(z, x), w) \end{aligned}$$

- beide Definitionen liefern das Gleiche: $f^* = \bar{f}^*$.
  Man nimmt die bequemere.

Verallgemeinerte Zustandsübergangsfunktionen (1)

- nach Eingabe eines ganzen Wortes $w \in X^*$:
  - Was ist der erreichte Zustand?
  - Welches sind alle durchlaufenen Zustände?
- definiere passende Funktionen f^* und f^{**}
  1. Stern: zweite Argument ist ganzes Wort von Eingabesymbolen
  2. Stern: interessieren uns für alle durchlaufenen Zuständen Zuständen
- definiere $f^* : Z \times X^* \to Z$:

$$\begin{aligned} f^*(z, \varepsilon) &= z \\ \forall w \in X^* : \forall x \in X : \quad f^*(z, wx) &= f(f^*(z, w), x) \end{aligned}$$

- alternativ:

$$\begin{aligned} \bar{f}^*(z, \varepsilon) &= z \\ \forall w \in X^* : \forall x \in X : \quad \bar{f}^*(z, xw) &= \bar{f}^*(f(z, x), w) \end{aligned}$$

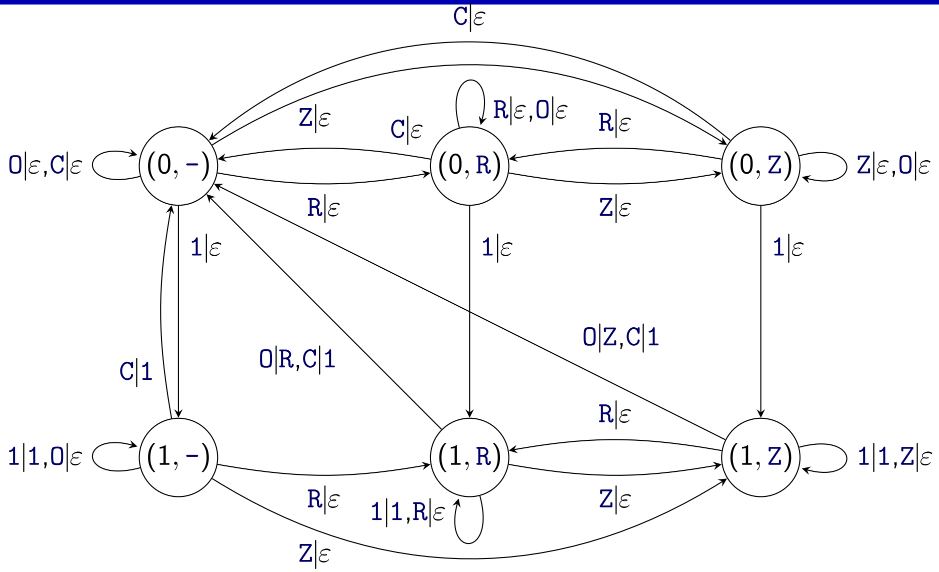
- beide Definitionen liefern das Gleiche: $f^* = \bar{f}^*$.
  Man nimmt die bequemere.

alle durchlaufenen Zustände:

- definiere $f^{**} : Z \times X^* \to Z^*$:

$$\begin{aligned} & f^{**}(z, \varepsilon) = z \\ \forall w \in X^* : x \in X : \qquad & f^{**}(z, wx) = f^{**}(z, w) \cdot f(f^*(z, w), x) \end{aligned}$$

- auch hier wieder eine alternative Definitionsmöglichkeit



$$f^{**}(\,(0,\texttt{-}),\texttt{R1RZO}) = (0,\texttt{-})\,(0,\texttt{R})\,(1,\texttt{R})\,(1,\texttt{R})\,(1,\texttt{Z})\,(0,\texttt{-})$$

Verallgemeinerte Ausgabefunktionen

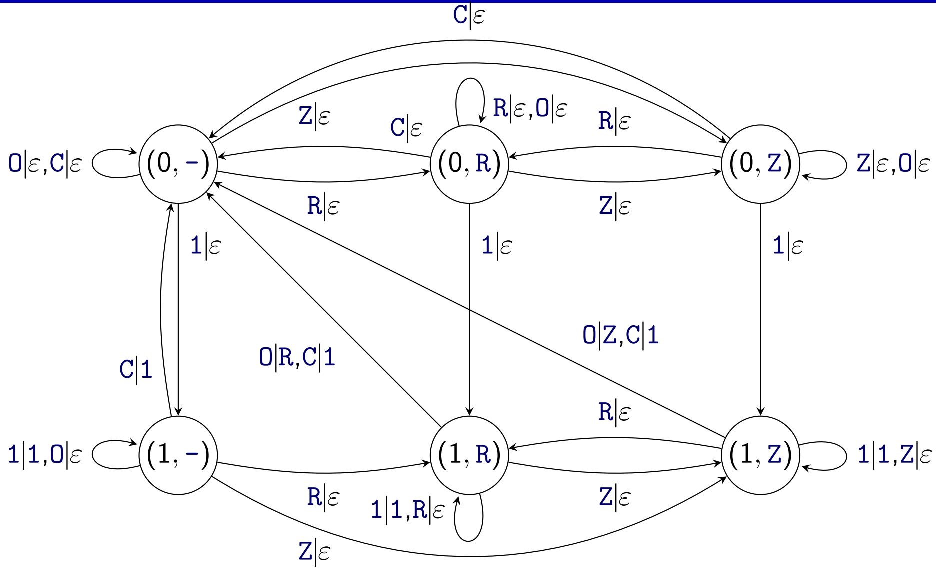
- für „die letzte“ Ausgabe: $g^* : Z \times X^* \to Y^*$

$$
\begin{aligned}
g^*(z, \varepsilon) &= \varepsilon \\
g^*(z, wx) &= g(f^*(z, w), x)
\end{aligned}
$$

- Für „alle Ausgaben konkateniert“: $g^{**} : Z \times X^* \to Y^*$:

$$
\begin{aligned}
g^{**}(z, \varepsilon) &= \varepsilon \\
g^{**}(z, wx) &= g^{**}(z, w) \cdot g^*(z, wx)
\end{aligned}
$$

Bespiel Getränkeautomaten



$$g^{**}(\,(0,\texttt{-}),\texttt{R1RZO}) = \varepsilon\varepsilon\varepsilon\varepsilon\texttt{Z} = \texttt{Z}$$

Was ist wichtig

Das sollten Sie mitnehmen:

- „offizielle“ Definition von Mealy-Automat:
  - Z
  - $z_0 \in Z$
  - X
  - $f : Z \times X \to Z$
  - Y
  - $g : Z \times X \to Y^*$ Ausgabe hängt von der Eingabe ab

Das sollten Sie üben:

- zu vorgegebenem „Verhalten“ Beispielautomaten konstruieren
- von vorgebenenen Automaten ihr Verhalten verstehen

Überblick

Erstes Beispiel: ein Getränkeautomat

Mealy-Automaten

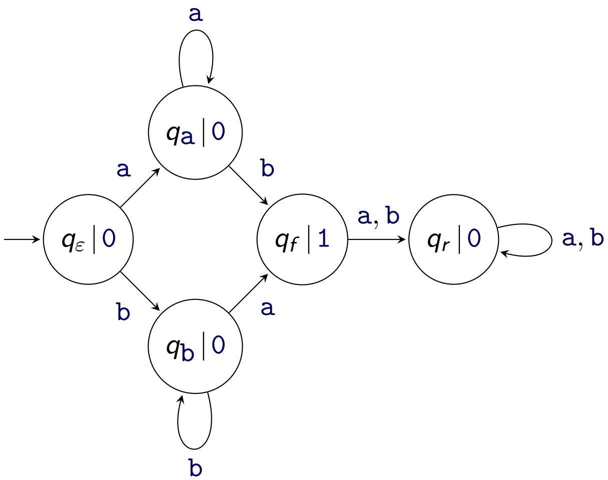
Moore-Automaten

Spezialfall: endliche Akzeptoren

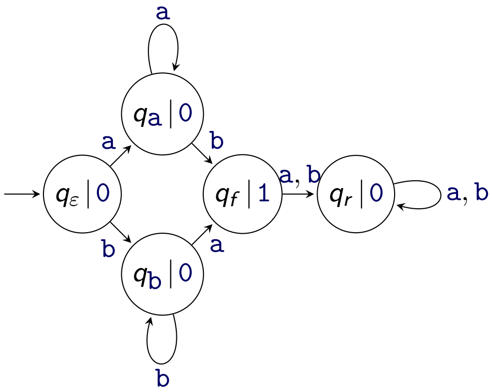
Moore-Automat

- manchmal näherliegend: Automat produziert „in jedem Zustand“ eine Ausgabe (nicht bei Zustandsübergang)
- (endlicher) Moore-Automat festgelegt durch
  - eine endliche *Zustandsmenge* Z,
  - einen *Anfangszustand* $z_0 \in Z$,
  - ein *Eingabealphabet* X,
  - eine *Zustandsüberführungsfunktion* $f : Z \times X \to Z$,
  - ein *Ausgabealphabet* Y,
  - eine *Ausgabefunktion* $h : Z \to Y^*$

graphische Darstellung analog zu Mealy-Automaten,
nur die Ausgaben in den Zuständen:



Definition von f^* und f^{**} wie bei Mealy-Automaten



Beispiel:

- $f^*(q_\varepsilon, \mathtt{aaaba}) = q_r$
- $f^{**}(q_\varepsilon, \mathtt{aaaba}) = q_\varepsilon q_{\mathtt{a}} q_{\mathtt{a}} q_{\mathtt{a}} q_f q_r$

Kleine Änderung bei der Notation für Homomorphismen

- Schreibe h^{**} statt h^* für den durch $h : A \to B^*$ induzierten Homomorphismus $A^* \to B^*$
- also $h^{**} : A^* \to B^*$ definiert vermöge

$$h^{**}(\varepsilon) = \varepsilon$$
$$\forall w \in A^* : \forall x \in A : h^{**}(wx) = h^{**}(w)h(x)$$

- Fakt

$$h^{**}(x_1 x_2 \cdots x_n) = h(x_1)h(x_2) \cdots h(x_n)$$

Verallgemeinerte Ausgabefunktionen g^* und g^{**}

- etwas einfacher als bei Mealy-Automaten
- „letzte Ausgabe“ $g^* = h \circ f^*$:

$$\forall (z,w) \in Z \times X^* : g^*(z,w) = h(f^*(z,w))$$

- „alle Ausgaben“: $g^{**} = h^{**} \circ f^{**}$

$$\forall (z,w) \in Z \times X^* : g^{**}(z,w) = h^{**}(f^{**}(z,w))$$

- Beispiel:
  - $f^*(q_\varepsilon, \mathtt{aaaba}) = q_r$
  - $f^{**}(q_\varepsilon, \mathtt{aaaba}) = q_\varepsilon q_{\mathtt{a}} q_{\mathtt{a}} q_{\mathtt{a}} q_f q_r$

  also
  - $g^*(q_\varepsilon, \mathtt{aaaba}) = h(f^*(q_\varepsilon, \mathtt{aaaba})) = h(q_r) = \mathtt{0}$
  - $$\begin{aligned} g^{**}(q_\varepsilon, \mathtt{aaaba}) &= h^{**}(f^{**}(q_\varepsilon, \mathtt{aaaba})) \\ &= h^{**}(q_\varepsilon q_{\mathtt{a}} q_{\mathtt{a}} q_{\mathtt{a}} q_f q_r) \\ &= h(q_\varepsilon) h(q_{\mathtt{a}}) h(q_{\mathtt{a}}) h(q_{\mathtt{a}}) h(q_f) h(q_r) \\ &= \mathtt{000010} \end{aligned}$$

Was ist wichtig

Das sollten Sie mitnehmen:

- „offizielle“ Definition von Moore-Automat:
  - Z
  - $z_0 \in Z$
  - X
  - $f : Z \times X \to Z$
  - Y
  - $h : Z \to Y^*$ Ausgabe hängt *nicht* von der Eingabe ab

Das sollten Sie üben:

- zu vorgegebenem Verhalten Moore-Automaten konstruieren, der es realisiert
- von vorgebenenem Moore-Automaten realisiertes Verhalten herausfinden

Überblick

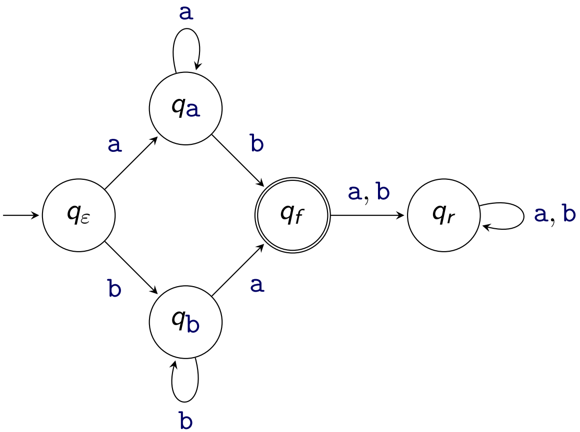
Endliche Akzeptoren

wichtiger Sonderfall von Moore-Automaten:
sogenannte endliche Akzeptoren

- Moore-Automaten mit immer genau einem Bit Ausgabe:
  - $Y = \{0, 1\}$ und
  - $\forall z : h(z) \in Y$
- Interpretation der Ausgabe:
  - Eingabe war „gut“ oder „schlecht“ bzw.
  - „syntaktisch korrekt“ oder „syntaktisch falsch“ (für eine gerade interessierende Syntax)
- bequemere Formalisierung:
  - Spezifikation der Menge F der *akzeptierenden Zustände*:
  - $F = \{z \mid h(z) = 1\}$
  - die anderen heißen *ablehnende Zustände*
- in graphischen Darstellungen

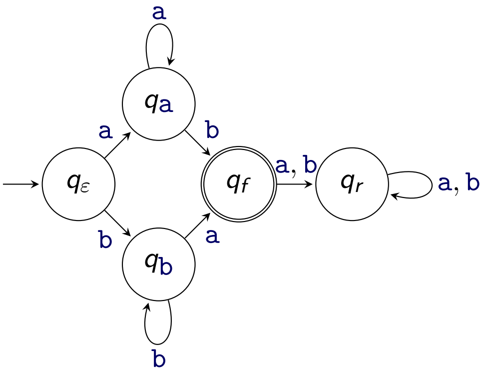
  akzeptierende Zustände mit doppeltem Kringel gemalt: 

der Moore-Automat aus dem vorangegangenen Abschnitt:



Akzeptierte und abgelehnte Wörter

- Wort $w \in X^*$ wird *akzeptiert*, falls $f^*(z_0, w) \in F$.
- Wort $w \in X^*$ wird *abgelehnt*, falls $f^*(z_0, w) \notin F$.



- aaaba wird abgelehnt, denn $f^*(z_0, \mathtt{aaaba}) = q_r \notin F$
- aaab wird akzeptiert, denn $f^*(z_0, \mathtt{aaab}) = q_f \in F$.
- allgemein:
  - Alle Wörter der Form $\mathtt{a}^k\mathtt{b}$ für ein $k \in \mathbb{N}_+$ werden akzeptiert.
  - Alle Wörter der Form $\mathtt{b}^k\mathtt{a}$ für ein $k \in \mathbb{N}_+$ werden akzeptiert.
  - Keine anderen Wörter werden akzeptiert.

Erkannte formale Sprache

- Die von einem Akzeptor $A = (Z, z_0, X, f, F)$ *akzeptierte* oder *erkannte formale Sprache* ist

$$L(A) = \{w \in X^* \mid f^*(z_0, w) \in F\}$$

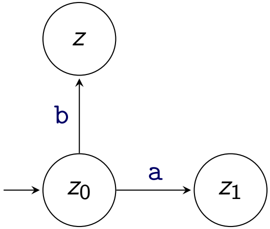
- Das ist ganz einfache „Syntaxanalyse“.
- in unserem Beispiel:

$$L(A) = \{\mathtt{a}\}^+\{\mathtt{b}\} \cup \{\mathtt{b}\}^+\{\mathtt{a}\}$$

Beispiel 2 einer erkennbaren Sprache

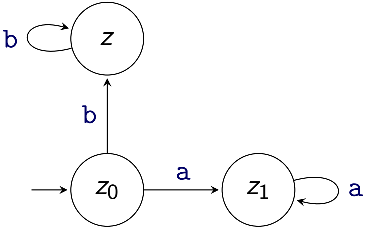
- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.

- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: erster Ansatz

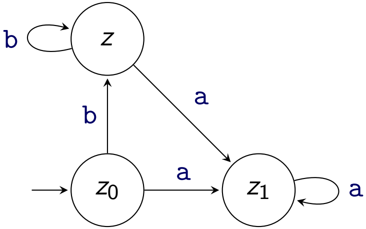


Beispiel 2 einer erkennbaren Sprache (2)

- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt

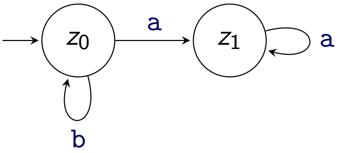


- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt



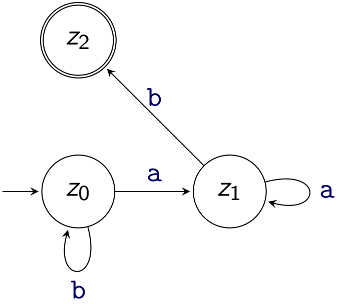
Beispiel 2 einer erkennbaren Sprache (4)

- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein b vor und
  - vor dem letzten b steht ein a
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt

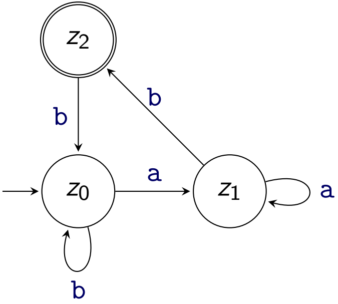


Beispiel 2 einer erkennbaren Sprache (5)

- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt

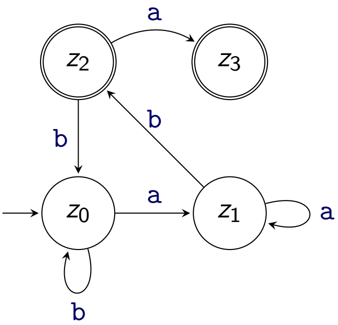


- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein b vor und
  - vor dem letzten b steht ein a
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt

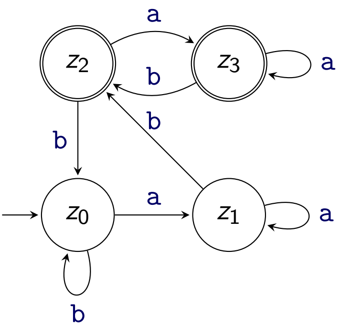


Beispiel 2 einer erkennbaren Sprache (7)

- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein $\mathtt{b}$ vor und
  - vor dem letzten $\mathtt{b}$ steht ein $\mathtt{a}$
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: nächster Schritt



- formale Sprache L aller Wörter $w \in \{\mathtt{a}, \mathtt{b}\}^*$ mit den Eigenschaften:
  - in w kommt mindestens ein b vor und
  - vor dem letzten b steht ein a
- Behauptung: Es gibt einen endlichen Akzeptor, der L erkennt.
- Konstruktion: letzter Schritt



Beispiel 3 einer erkennbaren Sprache

- eine Aufgabe aus dem Informatiker-Alltag
  - gegeben: Textdatei mit vielen Zeilen
  - gesucht: die Zeilen, in denen ein gewisses Wort m vorkommt
  - z. B. mit dem Programm `grep`
- mit anderen Worten
  - gegeben: Zeichenkette w und Textmuster m
  - gesucht: Algorithmus, der feststellt, ob in w das m vorkommt
  - das geht mit einem endlichen Akzeptor
- hier:
  - Eingabealphabet $X = \{\mathtt{a}, \mathtt{b}\}$
  - Textmuster $m = \mathtt{ababb}$
  - Ziel: endlicher Akzeptor A mit $L(A) = \{w_1\mathtt{ababb}w_2 \mid w_1, w_2 \in \{\mathtt{a}, \mathtt{b}\}^*\}$

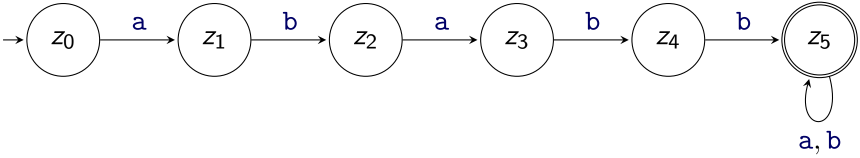
Beispiel 3 einer erkennbaren Sprache

- eine Aufgabe aus dem Informatiker-Alltag
  - gegeben: Textdatei mit vielen Zeilen
  - gesucht: die Zeilen, in denen ein gewisses Wort m vorkommt
  - z. B. mit dem Programm `grep`
- mit anderen Worten
  - gegeben: Zeichenkette w und Textmuster m
  - gesucht: Algorithmus, der feststellt, ob in w das m vorkommt
  - das geht mit einem endlichen Akzeptor
- hier:
  - Eingabealphabet $X = \{\mathtt{a}, \mathtt{b}\}$
  - Textmuster $m = \mathtt{ababb}$
  - Ziel: endlicher Akzeptor A mit
    $L(A) = \{w_1\mathtt{ababb}w_2 \mid w_1, w_2 \in \{\mathtt{a}, \mathtt{b}\}^*\}$

Beispiel 3 einer erkennbaren Sprache

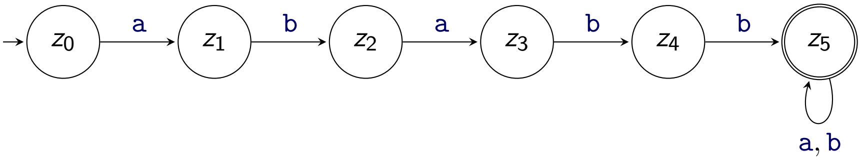
- eine Aufgabe aus dem Informatiker-Alltag
  - gegeben: Textdatei mit vielen Zeilen
  - gesucht: die Zeilen, in denen ein gewisses Wort m vorkommt
  - z. B. mit dem Programm `grep`
- mit anderen Worten
  - gegeben: Zeichenkette w und Textmuster m
  - gesucht: Algorithmus, der feststellt, ob in w das m vorkommt
  - das geht mit einem endlichen Akzeptor
- hier:
  - Eingabealphabet $X = \{\mathtt{a}, \mathtt{b}\}$
  - Textmuster $m = \mathtt{ababb}$
  - Ziel: endlicher Akzeptor A mit
    $L(A) = \{w_1\mathtt{ababb}w_2 \mid w_1, w_2 \in \{\mathtt{a}, \mathtt{b}\}^*\}$

- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

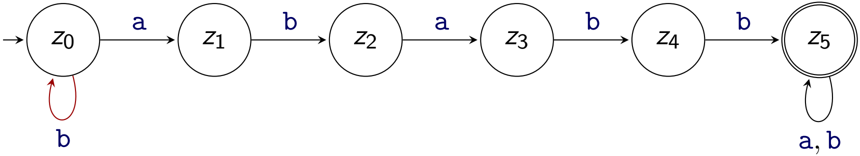
- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

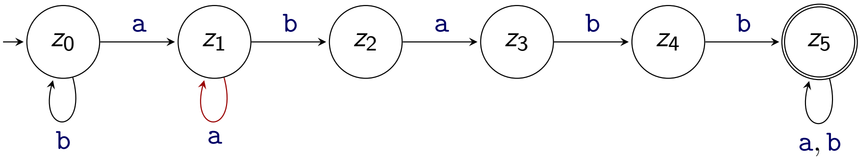
Beispiel 3 einer erkennbaren Sprache (3)

- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



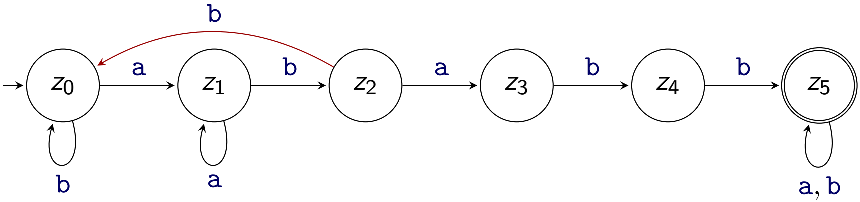
- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



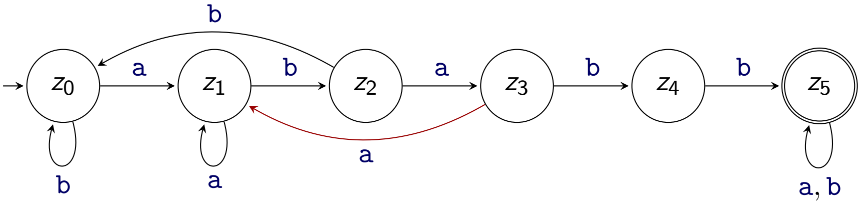
- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

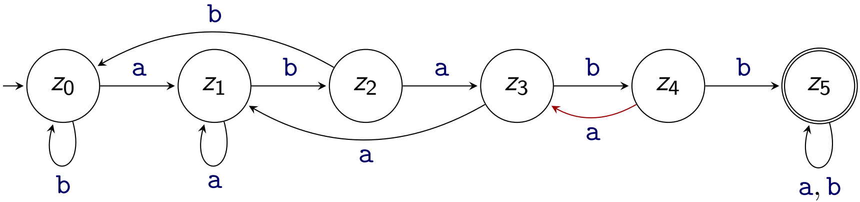
- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

Beispiel 3 einer erkennbaren Sprache (7)

- es gibt verschiedene Möglichkeiten
- hier: ein erster Ansatz



- es fehlen noch diverse Übergänge
- Was ist z. B. mit folgenden Wörtern?
  - bbbababb
  - aaaababb
  - abbababb
  - abaababb
  - abababb

Beispiel einer *nicht* erkennbaren Sprache

- Behauptung: Die formale Sprache

$$L = \{\mathtt{a}^k\mathtt{b}^k \mid k \in \mathbb{N}_0\}$$

  kann von keinem endlichen Akzeptor erkannt werden.
- Beweis?
  - „schwieriger“, als nur einen Akzeptor hinzumalen
  - man muss *alle* Akzeptoren „betrachten“
  - Wie macht man das?
    - betrachte „beliebigen“ endlichen Akzeptor A und
    - zeige: $L(A) \neq L$
  - Was bedeutet $L(A) \neq L$?
    - $L \not\subseteq L(A)$ oder
    - $L(A) \not\subseteq L$
  - folgenden Vorgehensweise ist „zielführend“: Fallunterscheidung:
    - 1. Fall: $L \not\subseteq L(A)$: dann offensichtlich $L(A) \neq L$
    - 2. Fall: $L \subseteq L(A)$: zeige: dann aber $L(A) \not\subseteq L$, d. h. der Automat akzeptiert auch ein „falsches“ Wort

- Behauptung: Die formale Sprache

$$L = \{\mathtt{a}^k \mathtt{b}^k \mid k \in \mathbb{N}_0\}$$

  kann von keinem endlichen Akzeptor erkannt werden.
- Beweis?
  - „schwieriger“, als nur einen Akzeptor hinzumalen
  - man muss *alle* Akzeptoren „betrachten“
  - Wie macht man das?
    - betrachte „beliebigen“ endlichen Akzeptor A und
    - zeige: $L(A) \neq L$
  - Was bedeutet $L(A) \neq L$?
    - $L \not\subseteq L(A)$ oder
    - $L(A) \not\subseteq L$
  - folgenden Vorgehensweise ist „zielführend“: Fallunterscheidung:
    - 1. Fall: $L \not\subseteq L(A)$: dann offensichtlich $L(A) \neq L$
    - 2. Fall: $L \subseteq L(A)$: zeige: dann aber $L(A) \not\subseteq L$, d. h. der Automat akzeptiert auch ein „falsches“ Wort

Beispiel einer *nicht* erkennbaren Sprache (2)

- $L = \{\mathtt{a}^k\mathtt{b}^k \mid k \in \mathbb{N}_0\}$
- bleibt noch zu zeigen: wenn $L \subseteq L(A)$, dann $L(A) \not\subseteq L$

- sei $m = |Z|$
- betrachte die Eingabe $w = \mathtt{a}^m\mathtt{b}^m$
- $f^{**}(z_0, \mathtt{a}^m)$ besteht aus $m+1$ Zuständen:
  - z_0
  - $z_1 = f(z_0, \mathtt{a})$
  - $z_2 = f(z_1, \mathtt{a})$
  - $\vdots$
  - $z_m = f(z_{m-1}, \mathtt{a})$
- soviele verschiedene gibt es gar nicht
- also kommt mindestens einer doppelt vor:
  A läuft in einer Schleife
- seien $i \geq 0$ und $\ell \geq 1$ die kleinsten Zahlen mit
  $z_i = z_{i+\ell}$, also $f^*(z_0, \mathtt{a}^i) = f^*(z_0, \mathtt{a}^{i+\ell})$

Beispiel einer *nicht* erkennbaren Sprache (3)

- seien $i \geq 0$ und $\ell \geq 1$ die kleinsten Zahlen mit $z_i = z_{i+\ell}$, also $f^*(z_0, \mathtt{a}^i) = f^*(z_0, \mathtt{a}^{i+\ell})$
- dann auch
  - $z_{i+1} = z_{i+\ell+1}$
  - $z_{i+2} = z_{i+\ell+2}$
  - $\vdots$
  - $z_{m-\ell} = z_m$
- also $f^*(z_0, \mathtt{a}^{m-\ell}) = f^*(z_0, \mathtt{a}^m)$
- A „unterscheidet nicht", ob m oder $m - \ell$ $\mathtt{a}$ in der Eingabe
- betrachte: $w' = \mathtt{a}^{m-\ell}\mathtt{b}^m \notin L$
- $$\begin{aligned} f^*(z_0, w') &= f^*(z_0, \mathtt{a}^{m-\ell}\mathtt{b}^m) = f^*(f^*(z_0, \mathtt{a}^{m-\ell}), \mathtt{b}^m) \\ &= f^*(f^*(z_0, \mathtt{a}^m), \mathtt{b}^m) = f^*(z_0, \mathtt{a}^m\mathtt{b}^m) \in F \end{aligned}$$
- $w' \in L(A)$ aber $w' \notin L$

Beispiel einer *nicht* erkennbaren Sprache (3)

- seien $i \geq 0$ und $\ell \geq 1$ die kleinsten Zahlen mit $z_i = z_{i+\ell}$, also $f^*(z_0, \mathtt{a}^i) = f^*(z_0, \mathtt{a}^{i+\ell})$
- dann auch
  - $z_{i+1} = z_{i+\ell+1}$
  - $z_{i+2} = z_{i+\ell+2}$
  - $\vdots$
  - $z_{m-\ell} = z_m$
- also $f^*(z_0, \mathtt{a}^{m-\ell}) = f^*(z_0, \mathtt{a}^m)$
- A „unterscheidet nicht", ob m oder $m - \ell$ $\mathtt{a}$ in der Eingabe
- betrachte: $w' = \mathtt{a}^{m-\ell}\mathtt{b}^m \notin L$
- $$\begin{aligned} f^*(z_0, w') &= f^*(z_0, \mathtt{a}^{m-\ell}\mathtt{b}^m) = f^*(f^*(z_0, \mathtt{a}^{m-\ell}), \mathtt{b}^m) \\ &= f^*(f^*(z_0, \mathtt{a}^m), \mathtt{b}^m) = f^*(z_0, \mathtt{a}^m\mathtt{b}^m) \in F \end{aligned}$$
- $w' \in L(A)$ aber $w' \notin L$

Was ist wichtig

Das sollten Sie mitnehmen:

- „offizielle“ Definition endlicher Akzeptoren:
  - Z
  - $z_0 \in Z$
  - X
  - $f : Z \times X \to Z$
  - $F \subseteq Z$
- Wenn ein „sehr langes“ Wort akzeptiert wird,
  - dann läuft der Automat in einer Schleife,
  - die beliebig oft durchlaufen werden kann ohne Akzeptanz zu ändern

Das sollten Sie üben:

- gegeben L: konstruiere A mit $L(A) = L$
- gegeben A: bestimme $L(A)$

Zusammenfassung

- Mealy-Automaten
- Moore-Automaten
  - taucht im Zusammenhang mit diversen *Protokollen* z. B. in Betriebssystemen und bei Kommunikationssystemen auf
- insbesonderen Akzeptoren
  - primitive Syntaxanalyse
  - aber oft nützlich, z. B. bei Compilerbau-Werkzeugen, Suche nach Text-Vorkommen, etc.