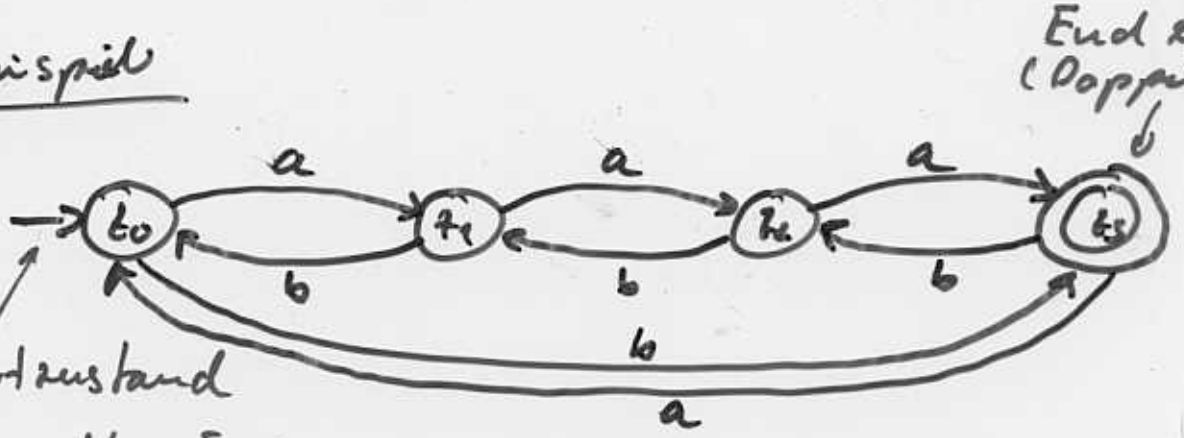
Beispiel



(ohne Vorgänger mit hineingehendem Pfeil)

Automat akzeptiert Wörter über dem Arbeitsalphabet $A = \{a, b\}$ mit beliebiger Aufeinanderfolge der a's und b's, wobei gilt

$$[(\text{Anzahl a's in } w) - (\text{Anzahl b's in } w)] \bmod 4 = 3$$

Definition: DFA M wird angegeben durch ein 5-Tupel $M = (Z, A, \delta, S, E)$ wo

Z: Zustandsmenge
A: Eingabealphabet
δ: Zustandsübergangsfunktion $\delta: Z \times A \to Z$
E: Menge der Endzustände $E \subseteq Z$
S: Startzustand $S \in Z$

und $Z \cap A = \emptyset$

Für den Beispielautomaten ist:

$$Z = \{z_0, z_1, z_2, z_3\}, \quad A = \{a, b\}, \quad E = \{z_3\}$$

$$\delta(z_0, a) = z_1, \quad \delta(z_0, b) = z_3$$

$$\delta(z_1, a) = z_2, \quad \delta(z_1, b) = z_0$$

$$\delta(z_2, a) = z_3, \quad \delta(z_3, b) = z_2$$

Sei also $w = c_1 c_2 \dots c_n$ ein Wort über dem Alphabet A. Mit Hilfe der Übergangsfunktion δ läßt sich sofort angeben, ob ein Wort akzeptiert wird

$$\delta(\dots \delta(\delta(z_0, c_1), c_2), \dots, c_n)$$

ist ein Endzustand

also: $L(M) = \{ c_1 c_2 \dots c_n \in A^* \mid$

$$\delta(\dots \delta(\delta(z_0, c_1), c_2), \dots c_n) \in E \}$$

oder über die Definition einer Übergangsfunktion für ganze Wörter

$$\hat{\delta}(z_0, c) = \hat{\delta}(z_0, c_1 c_2 \dots c_n) = \delta(\dots \delta(z_0, c_1), c_2 \dots)$$

Schreiben wir:

$$L(M) = \{ c \in A^* \mid \hat{\delta}(z_0, c) \in E \}$$

<u>Satz</u>: Jede durch einen EA erkannte Sprache ist vom Typ 3

<u>Beweis</u>: Zunächst eine Richtung EA → Typ 3 Grammatik durch Konstruktion der Grammatikregeln aus dem Automaten

1. Zustände des Automaten werden zu Nicht-Terminalen (Variablen) der Grammatik

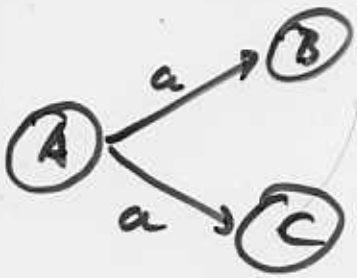
2. Jeder Zustandsübergang $\delta(z_i, c_n) = z_j$ wird in eine Produktion der Form $z_i \to c_n z_j$ umgewandelt.

3. Falls z_j Endzustand, kommt noch eine Regel der Form $z_i \to c_n$ hinzu.

Für den gegebenen Beispielautomaten ergibt sich

$$z_0 \to a z_1 \mid b z_3 \mid b$$
$$z_1 \to a z_2 \mid b z_0$$
$$z_2 \to a z_3 \mid b z_1 \mid a$$
$$z_3 \to a z_0 \mid b z_2$$

Bemerkung Typ 3 → EA ist kompliziert, weil Regeln der Form $A \to aB \mid aC$ im Automaten eine Verzweigung wie folgt aussehen würde

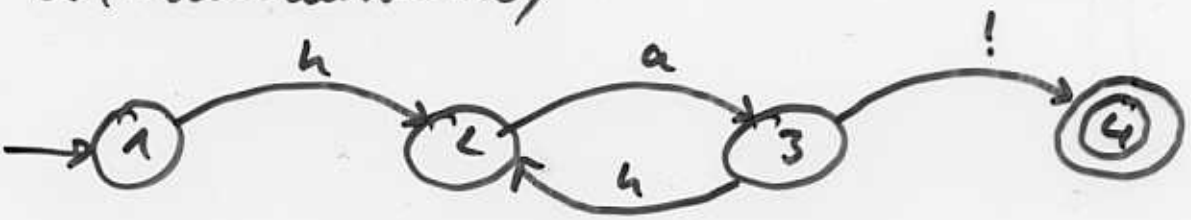


Dies somit nicht deterministische Automaten (siehe später)

Beispiel für EA als Akzeptor: "Lachautomat", akzeptiert reguläre Ausdrücke der Form:

$$(ha)+! = ha(ha)*! = h(ah)*a! =$$

L (Laufautomat)



Zwischenstufe wird Buchstabe für Buchstabe abgearbeitet; jeder Buchstabe führt zum Zustandswechsel, falls entsprechend gekennzeichnete Kante vorhanden.

Bemerkungen

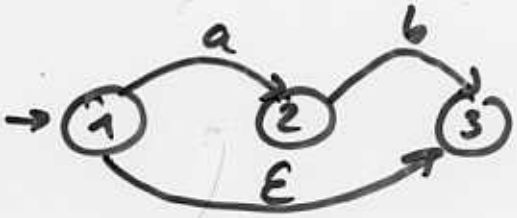
a) Automaten-Strukturen für unterschiedliche Ausdrücke:

Ein Terminalzeichen a

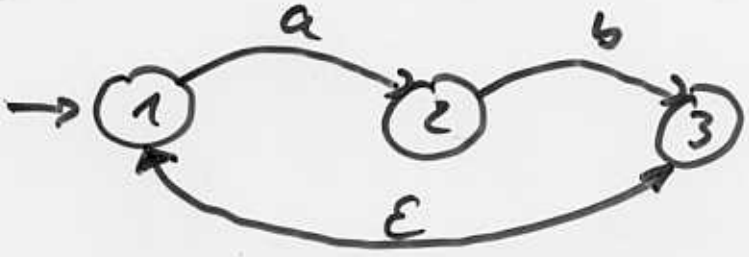
Alternative (a | b)

Verkettung a b

Wiederholung (a b)+

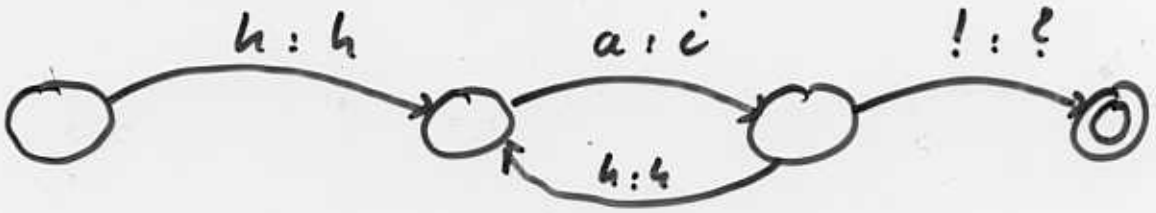


Optionalität (a b)?



Optionale Wiederholung (a b)*

b) Transducer sind EA, die beim Lesen auch eine Ausgabe erzeugen (nützlich für Wortumformungen, oder z.B. um Zustände auch außen mit zu teilen):



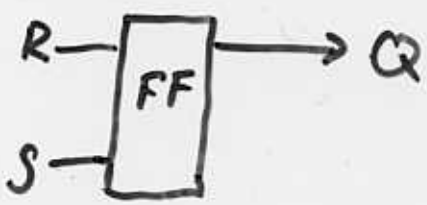
haha! → hihi?

Deterministische Automaten in der Technik

Beispiele

a) Modellierung von Hardware, wie etwa Flip-Flop, Dekodierer, Rechenwerke

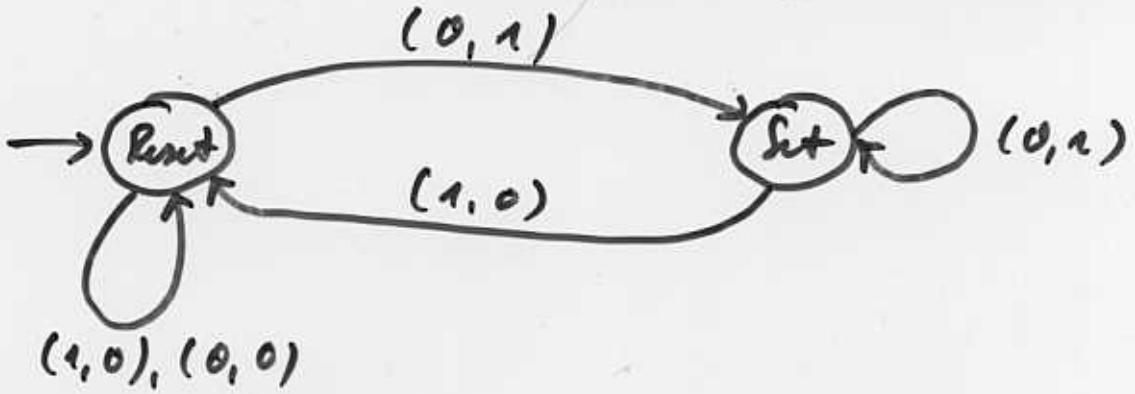
Flip-Flop als Black-Box wie folgt



Eingangswort = $(R,S) \in \mathbb{B}^2$

Ausgangswort = $Q \in \mathbb{B}$

Zustände = {Reset, Set}



→ Benutzung als elementarer Speicher

b) Modellierung von Protokollen / Bussystemen in der Netzwerktechnik, z.B. TCP/IP Protokollmaschinen

c) Komplexe technische Systeme, wie z.B. Ampelsteuerungen, Avionik

Probleme bei der Verwendung einfacher EA zur Modellierung von technischen Systemen mit einer Vielzahl von Parametern ⇒ Zustände müssen eingeführt werden für jede Kombination (Variation) der Variablenbelegungen
⇒ Führt zur Explosion der Zustandsmenge.
Ausweg: Zusammenfassung von "Basiszuständen" zu "Modulen" und ausschließliche Betrachtung der interessanten Veränderungen: Hierarchisierung, siehe "state charts", Teil von UML.