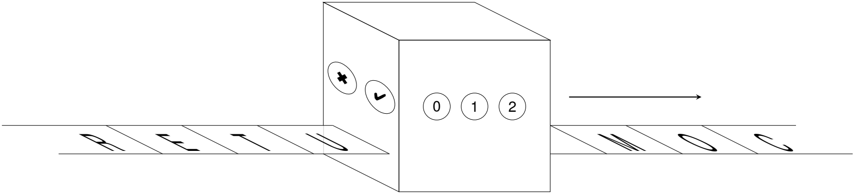
Abschnitt 4

Automaten

Überblick

- Automaten im Sinne der Vorlesung sind abstrakte, mathematische Konstrukte. Sie lassen sich jedoch als gegenständliche Maschine zur Datenverarbeitung veranschaulichen (ähnlich eines Computers).
- **Endliche Automaten** lesen ein **Eingabeband** mit Symbolen.
- Sie befinden sich stets ein einem bestimmten **Zustand**. Es gibt einen **Startzustand** und einen oder mehrere **Endzustände**.
- Das Eingabeband wird Symbol für Symbol gelesen. Abhängig vom aktuellen Zustand und dem nächsten Symbol des Eingabebands wechselt der endliche Automat in einen anderen Zustand.
- Der endliche Automat **akzeptiert eine Eingabe**, wenn er diese komplett lesen kann und sich danach in einem der Endzustände befindet. Sonst lehnt er sie ab.

Veranschaulichung



Definition

Ein nicht-deterministischer **endlicher Automat** ist ein 5-Tupel $(Q, \Sigma, \delta, q_0, F)$ mit

- einer **endlichen** Menge von Zuständen Q,
- einem **endlichen** Eingabealphabet Σ,
- einer Zustandsübergangsfunktion $\delta : Q \times \Sigma \rightarrow \wp(Q)$, die das Steuerungsverhalten des Automaten bestimmt
- **einem** Startzustand $q_0 \in Q$
- einer Menge von Endzuständen $F \subseteq Q$

Darstellung von Automaten: Tupelnotation I

$FSA = (Q, \Sigma, \delta, q_0, F)$ mit

$$
\begin{aligned}
Q =& \{0, 1, 2, 3\} \\
\Sigma =& \{a, b, f, o, r\} \\
\delta =& \{((0, f), \{1\}), \\
& ((0, b), \{1\}), \\
& ((1, o), \{2\}), \\
& ((1, a), \{2\}), \\
& ((2, o), \{3\}), \\
& ((2, r), \{3\})\} \\
q_0 =& 0 \\
F =& \{3\}
\end{aligned}
$$

Darstellung von Automaten: Tupelnotation II

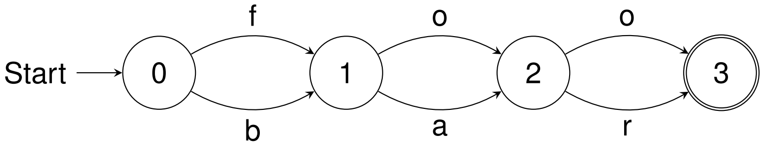
$FSA = (Q, \Sigma, \delta, q_0, F)$ mit

$$
\begin{aligned}
Q =& \{0, 1, 2, 3\} \\
\Sigma =& \{a, b, f, o, r\} \\
\delta :& \delta(0, f) = \{1\} \\
& \delta(0, b) = \{1\} \\
& \delta(1, o) = \{2\} \\
& \delta(1, a) = \{2\} \\
& \delta(2, o) = \{3\} \\
& \delta(2, r) = \{3\} \\
q_0 =& 0 \\
F =& \{3\}
\end{aligned}
$$

Tabellennotation der Übergangsfunktion

	f	o	b	a	r
0	1	-	1	-	-
1	-	2	-	2	-
2	-	3	-	-	5
3	-	-	-	-	-

Darstellung von Automaten: Graphnotation



Konfiguration

Sei $FSA = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat.

- Ein Paar $(q, \omega) \in Q \times \Sigma^*$ heißt **Konfiguration** von *FSA*. Konfigurationen sind also Paare aus Zustand und (verbliebenem) Eingabewort.
- Eine Konfiguration (q_0, ω) mit $\omega \in \Sigma^*$ heißt **initiale Konfiguration** oder **Startkonfiguration**.
- Eine Konfiguration (q, ε) mit $q \in F$ heißt **akzeptierende Konfiguration** oder **Endkonfiguration**.

Bewegung

Sei $FSA = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat, $\gamma \in \Sigma^*$ ein Wort über Σ und τ ein Symbol $\in \Sigma$.

- Der Wechsel von einer Konfiguration zur nächsten wird durch die Relation **Bewegung** $\vdash_{FSA} \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ dargestellt.
- Es gilt $(q, \tau\gamma) \vdash_{FSA} (q', \gamma)$ (sprich *"geht unter FSA nach"*) genau dann, wenn $q' \in \delta(q, \tau)$, also wenn q' ein möglicher Folgezustand von q ist.

Akzeptanz

Sei $FSA = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat, $\vdash^*_{FSA}$ die reflexive und transitive Hülle der Relation *Bewegung über FSA* und $\omega \in \Sigma^*$ ein Eingabewort.

- Das Eingabewort ω wird durch den Automaten **akzeptiert**, wenn es einen Zustand $q \in F$ gibt, sodass

$$(q_0, \omega) \vdash^*_{FSA} (q, \varepsilon)$$

Sprache eines Automaten

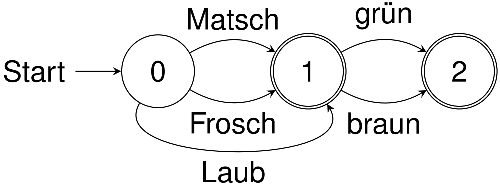
Sei $FSA = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat.

- Die Menge aller von FSA akzeptierten Wörter definiert die Sprache $\mathscr{L}_{\text{FSA}}$.

$$\mathscr{L}_{\text{FSA}} := \{\omega \in \Sigma^* \mid \text{FSA akzeptiert } \omega\}$$

Übung: Graph- zu Tupeldarstellung

Stellen Sie eine vollständige Definition für folgenden Automaten auf.
Welche Sprache definiert er?



Übung: Sprachen zu Automaten

Erstellen Sie zu folgenden Sprachen jeweils einen passenden Automaten.

- $L_1 = \{a, aa, aaa, \ldots\}$
- $L_2 = \{a, b, aa, ab, bb, aaa, aab, aba, abb, baa, bba, bbb, aaaa, ...\}$
- $L_3 = \{c, cab, cabab, cababab, ...\}$

Implementierung eines Deterministischen FSA

```
def automat_det(transition, eingabe, startzustand,
    endzustaende):
    zustand = startzustand
    for w in eingabe:
        folgezustand = transition.get((w, zustand))
        if not folgezustand: # folgezustand None?
            return False
        else:
            zustand = folgezustand
    if zustand in endzustaende:
        return True
    else:
        return False
```

```
trans = {("a",0): 1, ("b",1): 2}
automat_det(trans, "ab", 0, [2]) # True
```