

# Formale Sprachen und Automaten

*Robin Rausch*

*29. Oktober 2022*

## Inhaltsverzeichnis

<b>1 Grundlagen</b>	<b>1</b>
1.1 Alphabet . . . . .	1
1.2 Wort . . . . .	1
1.3 Formale Sprachen . . . . .	1
1.4 Kleene Stern . . . . .	1
<b>2 Reguläre Sprachen und endliche Ausdrücke</b>	<b>1</b>
2.1 Reguläre Ausdrücke . . . . .	1
2.2 Endliche Automaten . . . . .	2
2.2.1 Deterministische endliche Automaten(DEA) . . . . .	2
2.2.2 Nicht-deterministische endliche Automaten(NEA) . . . . .	3
2.2.3 Endliche Automaten und reguläre Ausdrücke . . . . .	4
2.2.4 Minimierung . . . . .	4
2.3 Nicht-reguläre Sprachen und das Pumping-Lemma . . . . .	4
2.4 Eigenschaften regulärer Sprachen . . . . .	4
<b>3 Chomsky Grammatiken und kontextfreie Sprachen</b>	<b>4</b>
<b>4 Turing Maschine</b>	<b>4</b>
<b>5 Entscheidbarkeit</b>	<b>4</b>
<b>6 Berechenbarkeit</b>	<b>4</b>
<b>7 Komplexität</b>	<b>4</b>



Operatoren:

$r_1 + r_2 \equiv r_2 + r_1$	Kommutativität von $+$
$(r_1 + r_2) + r_3 \equiv r_1 + (r_2 + r_3)$	Assoziativität von $+$
$(r_1 r_2) r_3 \equiv r_1 (r_2 r_3)$	Assoziativität von $\cdot$
$\emptyset r \equiv r \emptyset \equiv \emptyset$	Absorbierendes Element für $\cdot$
$\varepsilon r \equiv r \varepsilon \equiv r$	Neutrales Element für $\cdot$
$\emptyset + r \equiv r$	Neutrales Element für $+$
$(r_1 + r_2) r_3 \equiv r_1 r_3 + r_2 r_3$	Distributivität links
$r_1 (r_2 + r_3) \equiv r_1 r_2 + r_1 r_3$	Distributivität rechts
$r + r \equiv r$	Idempotenz von $+$
$(r^*)^* \equiv r^*$	Idempotenz von $*$
$\emptyset^* \equiv \varepsilon$	
$\varepsilon^* \equiv \varepsilon$	
$\varepsilon + r^* r \equiv r^*$	
$(\varepsilon + r)^* \equiv r^*$	
$r^* r \equiv r r^*$	

Nicht alle Operatoren sind für alle Typen zulässig:

	Vereinigung	Konkatenation	Potenz	Kleene-Stern
Wörter	<b>X</b>	$w_1 \cdot w_2$	$w^n$	<b>X</b>
Sprachen	$L_1 \cup L_2$	$L_1 \cdot L_2$	$L^n$	$L^*$
Reguläre Ausdrücke	$r_1 + r_2$	$r_1 \cdot r_2$	<b>X</b>	$r^*$

## 2.2 Endliche Automaten

Endliche Automaten sind eine andere Darstellung einer regulären Sprache. Endliche Ausdrücke lassen sich in Reguläre Ausdrücke umformen. Genauso auch anders herum.

Endliche Automaten lassen sich sowohl deterministisch als auch nicht-deterministisch darstellen.

### 2.2.1 Deterministische endliche Automaten(DEA)

Ein DEA hat endlich viele Zustände. Jeder mögliche Übergang muss hierbei behandelt werden können. D.h. für das Alphabet  $\Sigma_{ab}$  muss von jedem Zustand sowohl ein  $a$ , als auch ein  $b$  Übergang gegeben sein. Der Automat beginnt im Startzustand und muss im Endzustand enden. Wenn der Automat sich in einem Nicht-Endzustand befindet, befindet sich das Wort nicht in der Sprache, welche vom Automaten abgebildet wird.

Der DEA lässt sich durch folgendes 5-Tupel darstellen:

$\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$  mit den Komponenten:

$Q$  ist eine endliche Menge von Zuständen

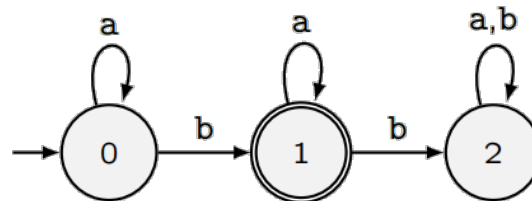
$\Sigma$  ist ein endliches Alphabet

$\delta : Q \times \Sigma \rightarrow Q$  ist die Übergangsfunktion

$q_0 \in Q$  ist der Startzustand

$F \subseteq Q$  ist die Menge der Endzustände

Beispiel:



$\mathcal{A}_b = (Q, \Sigma, \delta, q_0, F)$  mit

■  $Q = \{0, 1, 2\}$

■  $\Sigma = \Sigma_{ab}$

■  $\delta(0, a) = 0; \delta(0, b) = 1; \delta(1, a) = 1; \delta(1, b) = \delta(2, a) = \delta(2, b) = 2$

■  $q_0 = 0$

■  $F = \{1\}$

Zustand 2: „Mülleimerzustand“ (junk state), d.h. kein Wort wird mehr akzeptiert

## 2.2.2 Nicht-deterministische endliche Automaten(NEA)

Ein NEA hat endlich viele Zustände. Nicht jeder mögliche Übergang muss hierbei behandelt werden. D.h. für das Alphabet  $\Sigma_{ab}$  reicht es, nur den  $a$ -Übergang, bzw. nur den  $b$ -Übergang zu besitzen. Der Automat beginnt im Startzustand und muss im Endzustand enden. Wenn der Automat sich in einem Nicht-Endzustand befindet, befindet sich das Wort nicht in der Sprache, welche vom Automaten abgebildet wird. Zudem gibt es  $\epsilon$ -Übergänge, diese können jederzeit verwendet werden ohne ein Eingabesymbol.

Der NEA lässt sich durch folgendes 5-Tupel darstellen:

$\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$  mit den Komponenten:

$Q$  ist eine endliche Menge von Zuständen

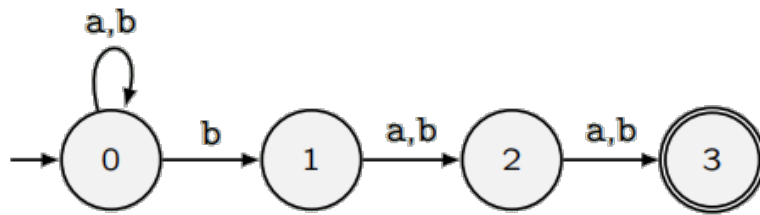
$\Sigma$  ist ein endliches Alphabet

$\Delta$  ist eine Relation über  $Q \times (\Sigma \cup \{\epsilon\}) \times Q$

$q_0 \in Q$  ist der Startzustand

$F \subseteq Q$  ist die Menge der Endzustände

Beispiel:



$\mathcal{A}_n = (Q, \Sigma, \Delta, q_0, F)$  mit

$Q = \{0, 1, 2, 3\}$

$\Sigma = \Sigma_{ab}$

$\Delta = \{(0, a, 0), (0, b, 0), (0, b, 1),$   
 $(1, a, 2), (1, b, 2),$   
 $(2, a, 3), (2, b, 3)\}$

$q_0 = 0$

$F = \{3\}$

$\mathcal{A}_n$		a	b	$\epsilon$
$\rightarrow$	0	{0}	{0, 1}	{ }
	1	{2}	{2}	{ }
	2	{3}	{3}	{ }
*	3	{ }	{ }	{ }

### 2.2.3 Endliche Automaten und reguläre Ausdrücke

TOP TEXT TOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXT-  
 TOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXT-  
 TOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP TEXTTOP  
 TEXTTOP TEXTTOP TEXTTOP TEXT.

### 2.2.4 Minimierung

## 2.3 Nicht-reguläre Sprachen und das Pumping-Lemma

## 2.4 Eigenschaften regulärer Sprachen

# 3 Chomsky Grammatiken und kontextfreie Sprachen

# 4 Turing Maschine

# 5 Entscheidbarkeit

# 6 Berechenbarkeit

# 7 Komplexität