Formale Sprachen // Alphabete

$L_1 = \{0^i \mid i \in \mathbb{N}_0\}$ und $L_2 = \{1^i \mid i \in \mathbb{N}_0\}$ seien formale Sprachen über dem Alphabet $\Sigma = \{0,1\}$. Berechnen Sie:

(a) $L_1 \cup L_2 = \{\epsilon, 0, 00, \dots, 1, 11, \dots\} = \{w \mid w = 0^i \vee w = 1^i \text{ mit } i \in \mathbb{N}_0\}$

(b) $L_1 \cap L_2 = \{\epsilon\}$

(c) $L_1 \setminus L_2 = \{0, 00, 000, \dots\} = \{0^i \mid i \in \mathbb{N}\}$

(d) $L_1 \cap \Sigma^* = L_1$

(e) $(L_1 \cup L_2) \cap \Sigma^3 = \{000, 111\}$

Anzahl Worte

Sei Σ ein Alphabet aus n Zeichen, $n \in \mathbb{N}$.

(a) Wie viele Wörter enthält Σ^m, $m \in \mathbb{N}_0$? — (a) n^m

(b) Wie viele Wörter enthält $\bigcup_{i=0}^{m} \Sigma^i$, $m \in \mathbb{N}_0$? — (b) $\sum_{i=0}^{m} n^i = \frac{n^{m+1}-1}{n-1}$

(c) Wie viele Wörter enthält Σ^*? — (c) abzählbar unendlich

NEA->DEA // Teilmengenkonstruktion a la Chat

Man hält im DEA fest in welchen Zuständen sich der Automat nach Lesen (oder bei Epsilon: Nicht-Lesen) eines Zeichens befinden könnte. Das macht man für jedes Zeichen des Alphabets. Am Ende markiert man alle neuen Zustände als Endzustände die zumindest einen Endzustand des NEAs aufweisen.
ACHTUNG: Immer gucken ob ein Epsilon an dem betrachteten Zustand hängt, dann gehört nämlich der nächste Zustand zur Liste dazu!

Beispiel Teilmengenkonstruktion mit Tabellen

Gegeben sei der nichtdeterministische endliche Automat $N = (\{1,2,3\}, \{a,b\}, \delta, 1, \{2\})$ mit

δ:

		a	b	ϵ
$\rightarrow$	1	$\{3\}$	$\emptyset$	$\{2\}$
$*$	2	$\{1\}$	$\emptyset$	$\emptyset$
	3	$\{2\}$	$\{2,3\}$	$\emptyset$

I. Ohne Berücksichtigung von ϵ-Überführungen:

	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{1\}$	$\{3\}$	$\emptyset$
$\{2\}$	$\{1\}$	$\emptyset$
$\{3\}$	$\{2\}$	$\{2,3\}$
$\{1,2\}$	$\{1,3\}$	$\emptyset$
$\{1,3\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{1,2\}$	$\{2,3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

II. Unter Berücksichtigung von ϵ-Überführungen:

R	$E(R)$	$\delta'(R,x)$: a	b
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\{1\}$	$\{1,2\}$	$\{3\}$	$\emptyset$
$\{2\}$	$\{2\}$	$\{1,2\}$	$\emptyset$
$\{3\}$	$\{3\}$	$\{2\}$	$\{2,3\}$
$\{1,2\}$	$\{1,2\}$	$\{1,2,3\}$	$\emptyset$
$\{1,3\}$	$\{1,2,3\}$	$\{2,3\}$	$\{2,3\}$
$\{2,3\}$	$\{2,3\}$	$\{1,2\}$	$\{2,3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

Reguläre Sprachen // Reguläre Ausdrücke

Stellen Sie die nachfolgenden Sprachen als reguläre Ausdrücke dar:

(a) $L = \{w \mid w \in \{0,1\}^*$ und w beginnt mit 0 und endet mit 1$\}$ — (a) $0(0 \mid 1)^*1$

(b) $L = \{w \mid w \in \{0,1\}^*$ und w enthält 11 mindestens einmal$\}$ — (b) $(0 \mid 1)^*11(0 \mid 1)^*$

(c) $L = \{w \mid w \in \{0,1\}^*$ und w enthält 11 genau einmal$\}$ — (c) $(0 \mid 10)^*11(0 \mid 01)^*$

(d) $L = \{w \mid w \in \{0,1\}^*$ und w enthält 11 höchstens einmal$\}$ — (d) $(0 \mid 10)^*(11 \mid 1 \mid \epsilon)(0 \mid 01)^*$

Pumping Lemma

Betrachte die Sprache $L = \{a^i b^i \mid i \in \mathbb{N}_0\}$.

Annahme: L sei regulär.

Dann muss es eine Konstante $n \in \mathbb{N}$ geben, so dass jedes Wort $w \in L$ mit $|w| \geq n$ in drei Teilwörter $w = xyz$ zerlegt werden kann, für die die Bedingungen des Pumping Lemmas erfüllt sind.

Betrachte das Wort $w = a^m b^m$ mit $m \geq n$:

Es gilt $|w| = |a^m b^m| = 2m > m \geq n$. Wegen $|xy| \leq n \leq m$, besteht dann xy ausschließlich aus a's. Daraus ergibt sich die folgende Zerlegung von $w = xyz$:

I. $x = a^r$ mit $r < n$, da $y \neq \epsilon$, d. h. $|y| > 0$.

II. $y = a^s$ mit $s > 0$ und $r + s \leq n$.

III. $z = a^t b^m$ mit $r + s + t = m$.

Nach dem Pumping Lemma muss nun für jedes $k \in \mathbb{N}_0$ $xy^kz \in L$ sein.

Mit $k = 0$ gilt

$$xy^0z = a^r(a^s)^0 a^t b^m = a^r a^t b^m = a^{r+t} b^m.$$

Da aber $r + t < m$, ist $xy^0z \notin L$. Es ergibt sich also ein Widerspruch zum Pumping Lemma.

Die Annahme, L sei regulär, muss also falsch sein.

Kontextfreie Sprachen // Kontextfreie Grammatiken

(a) $L_1 = \{a^n b^n c^m \mid m, n \in \mathbb{N}_0\}$

(a) $G_1 = (\{S, A, B\}, \{a, b, c\}, R_1, S)$

$$R_1: \quad S \rightarrow AB, \quad A \rightarrow aAb \mid \epsilon, \quad B \rightarrow cB \mid \epsilon$$

(b) $L_2 = \{a^m b^n c^n \mid m, n \in \mathbb{N}_0\}$

(b) $G_2 = (\{S, A, B\}, \{a, b, c\}, R_2, S)$

$$R_2: \quad S \rightarrow AB, \quad A \rightarrow aA \mid \epsilon, \quad B \rightarrow bBc \mid \epsilon$$

Mehrdeutigkeit

Es sei G eine kontextfreie Grammatik.

- *G heißt eindeutig, wenn es zu jedem Wort $w \in L(G)$ genau einen Ableitungsbaum gibt.*
- *G heißt mehrdeutig, wenn es ein Wort $w \in L(G)$ mit mehreren Ableitungsbäumen gibt.*

Eine kontextfreie Sprache L heißt inhärent mehrdeutig, falls es keine eindeutige kontextfreie Grammatik für L gibt.

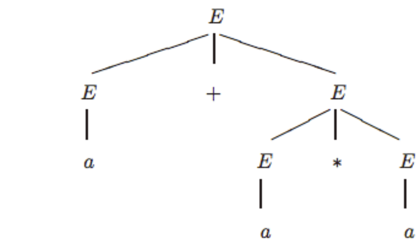


Abbildung 4.3: Ableitungsbaum für Wort $a + a * a$ (I)

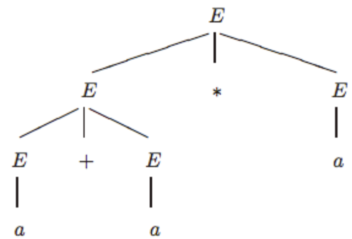


Abbildung 4.4: Ableitungsbaum für Wort $a + a * a$ (II)

Kellerautomat // Beispiel

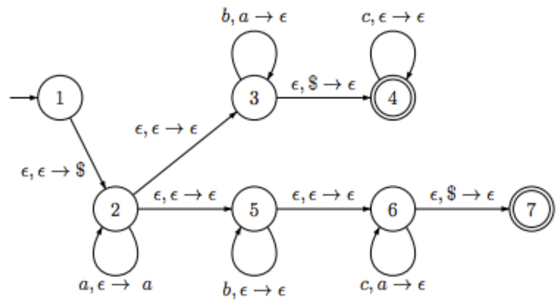
Es sei der Kellerautomat $K = (\{1,2,3,4,5,6,7\}, \{a,b,c\}, \{a,\$\}, \delta, 1, \{4,7\})$ gegeben, der die kontextfreie Sprache

$$L = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ und } i = j \text{ oder } i = k\}$$

erkennt. Die Überführungsfunktion δ ist wie folgt definiert:

Q	Σ_ϵ	Γ_ϵ	$\mathcal{P}(Q \times \Gamma_\epsilon)$
1	ϵ	ϵ	$\{(2,\$)\}$
2	a	ϵ	$\{(2,a)\}$
2	ϵ	ϵ	$\{(3,\epsilon),(5,\epsilon)\}$
3	b	a	$\{(3,\epsilon)\}$
3	ϵ	$\$$	$\{(4,\epsilon)\}$
4	c	ϵ	$\{(4,\epsilon)\}$
5	b	ϵ	$\{(5,\epsilon)\}$
5	ϵ	ϵ	$\{(6,\epsilon)\}$
6	c	a	$\{(6,\epsilon)\}$
6	ϵ	$\$$	$\{(7,\epsilon)\}$

Überführungsgraph



(c) $abbcc \notin L(K)$.

i	Q	$y_{i+1} \dots y_m \in \Sigma_\epsilon^*$	$s_i \in \Gamma_\epsilon^*$
0	1	$\epsilon a \epsilon b b \epsilon c c \epsilon$	ϵ
1	2	$a \epsilon b b \epsilon c c \epsilon$	$\$$
2	2	$\epsilon b b \epsilon c c \epsilon$	$a\$$
$\vdots$			
$\vdots$			
3	3	$b b \epsilon c c \epsilon$	$a\$$
4	3	$b \epsilon c c \epsilon$	$\$$
$\vdots$			
3	5	$b b \epsilon c c \epsilon$	$a\$$
4	5	$b \epsilon c c \epsilon$	$a\$$
5	5	$\epsilon c c \epsilon$	$a\$$
6	6	$c c \epsilon$	$a\$$
7	6	$c \epsilon$	$\$$

Kein Wort $abbcc = y_1 y_2 \dots y_m$, $y_i \in \Sigma_\epsilon$ wird nach Abarbeitung der Eingabe in einem akzeptierenden Zustand enden. Also wird $abbcc$ nicht akzeptiert.

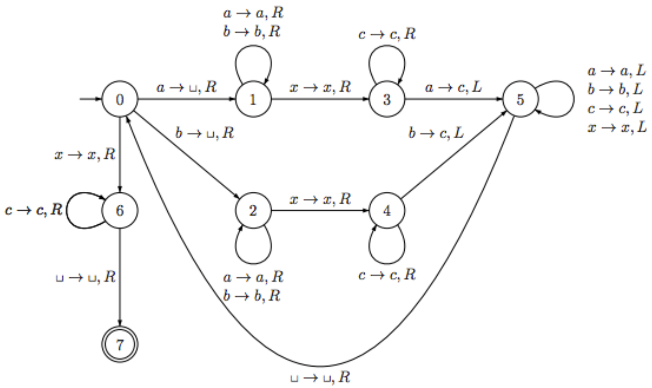
(b) Eingabewort $aabcc$:

i	Q	$y_{i+1} \dots y_m \in \Sigma_\epsilon^*$	$s_i \in \Gamma_\epsilon^*$
0	1	$\epsilon a a \epsilon b \epsilon c c \epsilon$	ϵ
1	2	$a a \epsilon b \epsilon c c \epsilon$	$\$$
2	2	$a \epsilon b \epsilon c c \epsilon$	$a\$$
3	2	$\epsilon b \epsilon c c \epsilon$	$aa\$$
4	5	$b \epsilon c c \epsilon$	$aa\$$
5	5	$\epsilon c c \epsilon$	$aa\$$
6	6	$c c \epsilon$	$aa\$$
7	6	$c \epsilon$	$a\$$
8	6	ϵ	$\$$
9	7		

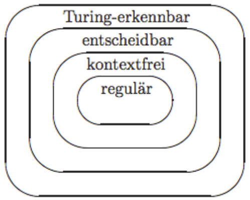
$aabcc \in L(K)$, wegen obiger Berechung und $7 \in F$.

Turingmaschinen

Beispiel



Hierarchie der Sprachfamilien



Algorithmusbegriff

Begriffsbestimmung 5.9 (Intuitiver Algorithmus)
Ein intuitiver Algorithmus ist ein allgemeines, eindeutiges Verfahren zur Lösung einer Klasse gleichartiger Probleme, gegeben durch einen aus elementaren Anweisungen bestehenden Text, der die nachfolgend aufgeführten Kriterien erfüllt:

- ***Endlichkeit:*** *Ein Algorithmus besteht aus endlich vielen elementaren Anweisungen (Schritten).*
- ***Bestimmtheit:*** *Jeder Schritt des Algorithmus muss eindeutig definiert sein (determiniert). Der jeweils nächste auszuführende Schritt muss eindeutig bestimmt sein (deterministisch). Dazu gehört auch die Bestimmung des ersten Schrittes und des Algorithmusendes.*
- ***Eingabe:*** *Ein Algorithmus hat keine oder endlich viele Eingabewerte aus einer Eingabemenge. Diese Werte werden initial an den Algorithmus gegeben bevor er mit der Ausführung des ersten Schritts beginnt.*
- ***Ausgabe:*** *Ein Algorithmus hat mindestens einen Ausgabewert aus einer Ausgabemenge. Der Algorithmus ordnet der Eingabe eine Ausgabe zu, er definiert also eine Funktion.*
- ***Effektivität:*** *Die einzelnen Anweisungen müssen überhaupt ausführbar sein.*

Turingberechnbar

Definition 5.11 (Turing-berechenbar)

- *Die Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ berechnet die Funktion $f_T : \Sigma^* \to \Gamma^*$, wenn T mit der Eingabe $w \in \Sigma^*$ in eine akzeptierende Konfiguration $uq_{accept}v$ gelangt. Die Zeichenfolge v ist der Funktionswert $f_T(w) = v$, $f_T(w)$ ist definiert. Andernfalls berechnet T keinen Funktionswert und $f_T(w)$ ist undefiniert.*
- *Eine Funktion f heißt Turing-berechenbar, falls es eine Turingmaschine T gibt, die f berechnet, also mit $f = f_T$.*
- *Ist f_T eine totale Funktion, dann berechnet T für jedes $w \in \Sigma^*$ einen Funktionswert.*
- *Ist f_T eine partielle Funktion, dann kann es $w \in \Sigma^*$ geben, für die T keinen Funktionswert berechnet.*

Folgerung 5.2 (Turing-berechenbar und intuitiv-berechenbar)
Jede Turing-berechenbare Funktion ist auch intuitv-berechenbar.

Hypothese 5.3 (Church-Turing These)
Jede intuitiv-berechenbare Funktion ist auch Turing-berechenbar.

Definition 5.12 (berechenbar)
Eine Funktion heißt berechenbar, wenn sie intuitiv-berechenbar, Turing-berechenbar oder nach irgendeiner anderen Formalisierung berechenbar ist.

Entscheidbare Probleme

Definition 6.1 (Wortproblem)
Die Frage, ob ein Wort zu einer Sprache gehört oder nicht, heißt Wortproblem.

Satz 6.2
Jede reguläre Sprache ist entscheidbar.

Satz 6.4
Jede kontextfreie Sprache ist entscheidbar.

Das Halteproblem

Definition 6.4 (Halteproblem)
Die Frage, ob eine Turingmaschine angesetzt auf eine Eingabe anhält oder nicht, heißt Halteproblem.

Satz 6.6 (Unentscheidbarkeit des Halteproblems)
Es gibt keine Turingmaschine, die das Halteproblem entscheidet.

Satz 6.7
Es gibt Sprachen, die nicht Turing-erkennbar sind.

Definition 6.8
Eine Sprache heißt co-Turing-erkennbar, wenn ihr Komplement Turing-erkennbar ist.

Satz 6.10
Eine Sprache ist genau dann entscheidbar, wenn sie Turing-erkennbar und co-Turing-erkennbar ist.