Formale Sprachen und Automaten

Robin Rausch

29. Oktober 2022

Inhaltsverzeichnis

1 Grundlagen

1.1 Alphabet

Ein Alphabet Σ ist eine nicht-leere Menge von Symbolen(Zeichen, Buchstaben). Beispiel: $\Sigma_{ab} = a, b$

1.2 Wort

Ein Wort w über dem Alphabet Σ(Sigma) ist eine endliche Folge von Symbolen aus Σ. Das Wort $w = abaabab$ wurde beispielsweise aus dem Alphabet Σ_{ab} gebildet.
Die Länge eines Wortes kann durch Betragsstriche angegeben werden. Beispiel: $|w| = 7$
Ebenso kann man die Anzahl bestimmter Symbole in einem Wort bestimmen: $|w|_b = 3$
Ein einzelnes Zeichen kann durch eckige Klammern angegeben werden: $w[2] = b$
Wörter können bliebig konkateniert werden(hintereinanderschreiben ohne abstand): $w_1w_2 = abbabaab$ mit $w_1 = abba$ und $w_2 = baab$.
Wörter dürfen auch potenziert werden: $w^3 = abaababababaababababaabab = www$
Das leere Wort lautet ϵ.

1.3 Formale Sprachen

Eine formale Sprache L über einem Alphabet Σ ist eine Menge von Wörtern aus $\Sigma^* : L \subseteq \Sigma^*$. Eine Sprache kann sowohl endlich als auch unendlich sein.
Beispiel: $L_1 = \{w \in \Sigma^*_{bin}| \; |w| \geqslant 2 \wedge w[|w| - 1] = 1\}$ ist die Menge aller Binärwörter, an deren vorletzter Stelle 1 steht.
Das Produkt zweier formaler Sprachen: $L_1 \cdot L_2 = \{abac, abcb, bcac, bccb\}$ mit $L_1 = \{ab, bc\}$ und $L_2 = \{ac, cb\}$.
Sprachen können ebenfalls potenziert werden: $L^2 = \{ab, ba\} \cdot \{ab, ba\} = \{abab, abba, baab, baba\}$

1.4 Kleene Stern

Für ein Alphabet Σ und eine formale Sprache $L \subseteq \Sigma^*$ ist der Operator Kleene Stern wie folgt definiert: $L^* = \bigcup\limits_{n \in \mathbb{N}} L^n$.

Beispiel: Sei $L_1 = \{ab, ba\}$, dann $L^* = \{\epsilon, ab, ba, abab, abba, baab, baba, ababab, ...\}$.

2 Reguläre Sprachen und endliche Ausdrücke

2.1 Reguläre Ausdrücke

Ein regulärer Ausdruck über Σ beschreibt eine formale Sprache.
Die Menge aller regulären Ausdrücke über Σ ist eine formale Sprache.

Beispiel: Sprache aller Wörter über Σ_{abc}, die nur aus genau zwei Symbolen bestehen:
Ausdruck: $r_1 = (a + b + c)(a + b + c)$
Sprache: $\mathcal{L}(r_1) = \{w \in \Sigma^*_{abc} \mid |w| = 2\}$

Operatoren:

$$
\begin{array}{rcll}
r_1 + r_2 & \equiv & r_2 + r_1 & \text{Kommutativität von } + \\
(r_1 + r_2) + r_3 & \equiv & r_1 + (r_2 + r_3) & \text{Assoziativität von } + \\
(r_1 r_2) r_3 & \equiv & r_1 (r_2 r_3) & \text{Assoziativität von } \cdot \\
\emptyset r \equiv r \emptyset & \equiv & \emptyset & \text{Absorbierendes Element für } \cdot \\
\varepsilon r \equiv r \varepsilon & \equiv & r & \text{Neutrales Element für } \cdot \\
\emptyset + r & \equiv & r & \text{Neutrales Element für } + \\
(r_1 + r_2) r_3 & \equiv & r_1 r_3 + r_2 r_3 & \text{Distributivität links} \\
r_1 (r_2 + r_3) & \equiv & r_1 r_2 + r_1 r_3 & \text{Distributivität rechts} \\
r + r & \equiv & r & \text{Idempotenz von } + \\
(r^*)^* & \equiv & r^* & \text{Idempotenz von } ^* \\
\emptyset^* & \equiv & \varepsilon & \\
\varepsilon^* & \equiv & \varepsilon & \\
\varepsilon + r^* r & \equiv & r^* & \\
(\varepsilon + r)^* & \equiv & r^* & \\
r^* r & \equiv & r r^* &
\end{array}
$$

Nicht alle Operatoren sind für alle Typen zulässig:

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

2.2 Endliche Automaten

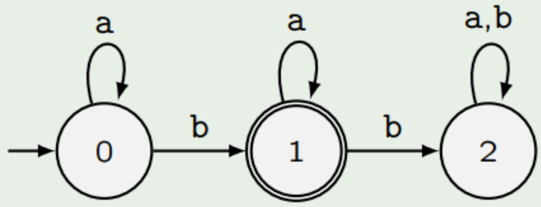
Endliche Automaten erkennen regulären Sprachen. Endliche Ausdrücke lassen sich in Reguläre Ausdrücke umformen. Genauso auch anders herum.
Endliche Automaten lassen sich sowohl deterministisch als auch nicht-deterministisch darstellen.

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 Σ_{ab} muss von jedem Zustand sowohl ein a, als auch ein b Übergang gegeben sein. Er terminiert wenn das Wort zu ende und Endzustand erreicht ist.

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
Σ ist ein endliches Alphabet
$\delta : Q \times \Sigma \to Q$ 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)

NEAs sind DEAs mit der Möglichkeit für ein Symbol mehrere Wege zu gehen.

2.2.3 Endliche Automaten und reguläre Ausdrücke

2.2.4 Minimierung

2.3 Nicht-reguläre Sprachen und das Pumping-Lemma

2.4 Eigenschaften regulärer Sprachen

3 Chomsky Grammatiken und kontextfreie Sprachen

Gramatiken erzeugen formale Sprachen dar.
G=$(N, \sum, P, S)$ mit:

N Nichtterminalsymbole. Diese können für Regeln verwendet werden, aber dürfen nicht selbst im abgeleiteten Wort stehen.

P Ableitungsregeln. Bsp.: P=$\{S \rightarrow Aa|\epsilon, A \rightarrow a\}$

S Startsymbol (ist nichtterminel)

3.1 Typ0 unbeschränkt

3.2 Typ1 Monoton

$\alpha \rightarrow \beta$ mit $|\alpha| \leq |\beta|$ und Ausnahme $S \rightarrow \epsilon$, wenn S auf keiner rechten Seite ist.

3.3 Typ2 Kontextfreie

A $\rightarrow \beta$ mit A $\in$ N und $\beta \in V$*

3.4 Typ3 rechtsregulär/-linear

A $\rightarrow$ cB mit A $\in$ N; B $\in$ N $\cup$ $\{\epsilon\}$; c $\in \sum \cup \{\epsilon\}$

3.5 Chomskynormalform CNF

Ohne ϵ und unnötige Regeln und Terminalsymbole.

4 Turing Maschine

Automaten mit endlosem Einleseband.
Terminiert wenn Endzustand erreicht und Einleseband nicht verschiebbar ist.
M=(Q, $\sum, \Gamma, \Delta, q_0$, F) mit:

Γ Vereinigung aus mindestens Blank-Symbol und Terminalsymbole: $\Gamma \supseteq \sum \cup \{\square\}$

Δ Übergangsrelationen Syntax: IST-Zustand IST-Inhalt Neuer-Inhalt Verschiebung Neuer-Zustand Bsp.:

5 Entscheidbarkeit

6 Berechenbarkeit

7 Komplexität