

Grundlagen der Theorie formaler Sprachen

Wir beginnen damit, dass wir in diesem Kapitel zunächst einige grundlegende Begriffe und Methoden aus der Theorie formaler Sprachen, insbesondere der regulären Sprachen, wiederholen.

1.1 Reguläre Sprachen und endliche Automaten

1.1.1 Reguläre Sprachen

Sei $\Sigma = \{a, b, \dots\}$ ein endliches *Alphabet*. Ein *endliches Wort* über Σ ist eine Folge $w = a_0 \dots a_{n-1}$, wobei $a_i \in \Sigma$ für $i = 0, \dots, n-1$. Wir schreiben $|w|$ für die *Länge* von w , also n in diesem Fall, und $w(i)$ für das i -te Symbol a_i von w . Das *leere Wort*, also die Folge der Länge 0, wird mit ε bezeichnet. Σ^* bezeichnet die Menge aller Wörter über Σ und Σ^+ die Menge aller nichtleeren Wörter über Σ . Die *Konkatenation* zweier Wörter w und v entsteht durch Anhängen der Folge v an die Folge w und wird mit wv bezeichnet.

Eine *Sprache* ist eine Menge von Wörtern, also eine Teilmenge $L \subseteq \Sigma^*$. Somit sind alle üblichen, mengentheoretischen Operationen wie Vereinigung, Schnitt, Komplement, Differenz, etc. auch Operationen auf Sprachen. Das *Komplement* einer Sprache L bezeichnen wir üblicherweise mit $\bar{L}$, und es ist definiert als $\Sigma^* \setminus L$.

Andere wichtige Operationen auf Sprachen sind die Konkatenation und der Kleene-Abschluss (Kleene-Stern). Sind L_1 und L_2 zwei Sprachen, so ist deren *Konkatenation* die Sprache $L_1 L_2 := \{w_1 w_2 \mid w_1 \in L_1, w_2 \in L_2\}$, die also durch Konkatenation beliebiger Wörter aus L_1 mit beliebigen Wörtern aus L_2 entsteht.

Ist L eine Sprache, dann ist ihr Kleene-Abschluss die Sprache $L^* := \{w_1 \dots w_n \mid n \in \mathbb{N}, w_i \in L \text{ für alle } i = 1, \dots, n\}$. Diese entsteht also durch beliebig, aber nur endlich oft wiederholte Konkatenation der Sprache mit sich selbst. Beachte, dass $\varepsilon \in L^*$ für jedes $L \subseteq \Sigma^*$ gilt, denn auch $n = 0$ ist in

der Definition des Kleene-Abschlusses zugelassen. Anders gesagt gilt für alle $L \subseteq \Sigma^*$:

$$L^0 := \{\varepsilon\} , \quad L^{i+1} := LL^i , \quad L^* := \bigcup_{i \in \mathbb{N}} L^i$$

Eine Klasse von Sprachen, für die wir uns hier insbesondere interessieren, ist die der *regulären Sprachen* über dem Alphabet Σ , bezeichnet mit REG_Σ . Kann Σ aus dem Kontext erkannt werden, so schreiben wir auch einfach nur REG .

Definition 1.1. Die Klasse REG_Σ ist die kleinste Klasse von Sprachen über Σ , für die gilt:

- a) $\emptyset \in \text{REG}_\Sigma$ und $\{a\} \in \text{REG}_\Sigma$ für jedes $a \in \Sigma$,
- b) wenn $L_1, L_2 \in \text{REG}$, dann $L_1 \cup L_2 \in \text{REG}$, $L_1 L_2 \in \text{REG}$ und $L_1^* \in \text{REG}$.

Hier bedeutet *kleinste Klasse*, dass keine Sprache, die nicht durch (1) oder (2) abgedeckt ist, zu REG gehört. Dies ist gleichbedeutend damit, dass eine Sprache nur dann zu REG gehört, wenn sie entweder von einer in (1) genannten Form ist, oder aber sich laut (2) aus einer oder zweien Sprachen zusammensetzt, von denen bereits bekannt ist, dass sie zu REG gehören. Noch anders gesagt: Jede reguläre Sprache lässt sich durch Anwenden von nur *endlich vielen* Operationen, die in (2) genannt sind, aus den Sprachen, die in (1) genannt sind, erzeugen.

Beispiel 1.2. Die Menge L aller Wörter über dem Alphabet $\{a, b\}$, in denen nach jedem a irgendwann später noch ein b folgt, ist regulär. Man überlege sich zunächst, dass die informelle Beschreibung äquivalent dazu ist, dass ein Wort in der Sprache nicht auf a enden darf. Da es nur zwei verschiedene Buchstaben gibt, muss jedes Wort in der Sprache also auf b enden oder leer sein. Somit ist

$$L = \{\varepsilon\} \cup ((\{a\} \cup \{b\})^* \{b\}) .$$

Es bleibt nur noch zu sehen, dass auch $\{\varepsilon\}$ eine reguläre Sprache ist. Dies ist der Fall, denn es gilt $\{\varepsilon\} = \emptyset^*$.

Bei der formalen Beschreibung einer Sprache erlauben wir es uns auch, möglichst viele Klammern wegzulassen. Dazu lassen wir den Kleene-Stern stärker als die Konkatenation und diese wiederum stärker als die Vereinigung binden. Außerdem schreiben wir Einermengen nicht explizit als solche auf. Die Sprache aus dem vorigen Beispiel könnte man also auch als $\varepsilon \cup (a \cup b)^* b$ schreiben.

Man sieht z.B. leicht, dass jede endliche Menge von Wörtern eine reguläre Sprache bildet, da sich jede Sprache, die nur ein Wort enthält, durch endlich viele Konkatenationen von Sprachen der Form $\{a\}$ bilden lässt. Eine Sprache mit endlich vielen Wörtern lässt sich dann leicht als endliche Vereinigung beschreiben. Beachte, dass die Klasse der regulären Sprachen nicht unter unendlichen Vereinigungen abgeschlossen ist, denn dann wäre ja jede Sprache

regulär, da sich jede Sprache als unendliche Vereinigung der Einermengen ihrer Wörter schreiben lässt.

Der Punkt (2) in der Definition von REG bedeutet, dass REG *abgeschlossen* ist unter den Operationen Vereinigung, Konkatenation und Kleene-Iteration. REG ist darüberhinaus noch abgeschlossen unter einer ganzen Menge anderer Operationen, u.a. unter all den üblichen mengentheoretischen Operationen. Dies ist jedoch für den Schnitt und das Komplement nicht unbedingt offensichtlich aus der Definition. Um solche Abschlusseigenschaften zu beweisen, braucht man in der Regel andere Charakterisierungen der regulären Sprachen. Diese sind auch nötig, um reguläre Sprachen repräsentieren und verarbeiten zu können, da es sich bei Sprachen im Allgemeinen um unendlich große Mengen handelt.

1.1.2 Endliche Automaten

Eine der wichtigsten Repräsentationsformalismen für reguläre Sprachen sind endliche Automaten, die wir im folgenden wiederholen werden.

Definition 1.3. Ein *nichtdeterministischer, endlicher Automat* (NFA) ist ein Tupel $\mathcal{A} = (Q, \Sigma, q_I, \delta, F)$ mit

- endlicher Zustandsmenge Q ,
- Eingabealphabet Σ ,
- Anfangszustand $q_I \in Q$,
- Transitionsfunktion $\delta : Q \times \Sigma \rightarrow 2^Q$,
- Endzustandsmenge $F \subseteq Q$.

Ein *Lauf* eines NFA $\mathcal{A}$ auf einem Wort $w = a_0 \dots a_{n-1}$ ist eine Folge $q_0 \dots q_n$, so dass $q_0 = q_I$ und für alle $i = 0, \dots, n-1$ gilt: $q_{i+1} \in \delta(q_i, a_i)$. Der Lauf heißt *akzeptierend*, falls $q_n \in F$. Sei $L(\mathcal{A}) := \{w \in \Sigma^* \mid \text{es gibt einen akzeptierenden Lauf von } \mathcal{A} \text{ auf } w\}$ die von $\mathcal{A}$ *erkannte Sprache*.

Zwei Automaten sind *äquivalent*, wenn sie dieselbe Sprache erkennen.

Als *Größe* eines NFA $\mathcal{A}$ bezeichnen wir normalerweise die Anzahl seiner Zustände und schreiben dies auch als $|\mathcal{A}|$. Beachte, dass dies kein exaktes Maß für den Platz ist, den man braucht, um einen NFA niederzuschreiben. Sei $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$. Dann gilt i.A. $|\delta| = \mathcal{O}(|Q|^2 \cdot |\Sigma|)$. D.h. bei festgelegtem Alphabet kann die Transitionstabelle quadratisch in der Zustandszahl wachsen. Dennoch ist die Anzahl der Zustände eines Automaten ein sinnvolles Maß für seine Komplexität.

NFAs lassen sich am einfachsten als kanten-beschrifteter Graph darstellen. Die Zustände bilden die Knoten, die Zustandsübergangsfunktion δ wird durch Kanten in dem Graph repräsentiert. Ist $q' \in \delta(q, a)$ für einen Zustand q und ein Alphabetsymbol a , so wird eine Kante von q nach q' mit Beschriftung a gezogen. Normalerweise wird der Anfangszustand durch eine von nirgendwoher eingehende und unbeschriftete Kante gekennzeichnet und Endzustände durch einen Doppelkreis ausgezeichnet.

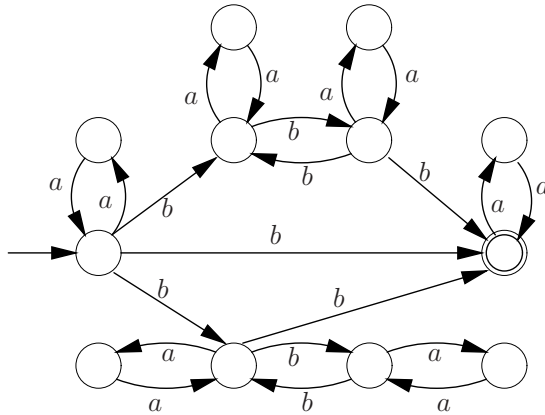


Abb. 1.1. Beispiel eines NFA.

Ein Beispiel ist in Abb. 1.1 gezeigt. Zur Übung bestimme man die Sprache, die von diesem NFA erkannt wird und versuche, einen NFA mit nur 4 Zuständen für dieselbe Sprache zu konstruieren.

Viele der Operationen, unter denen die Klasse der regulären Sprachen abgeschlossen ist, lassen sich ebenfalls auf NFAs ausführen, welches die folgenden Resultate besagen. Wir zeigen exemplarisch die Konstruktion für den Kleene-Stern, die anderen sind Übungen.

Satz 1.4. Seien $\mathcal{A}_i$ zwei NFAs für $i = \{1, 2\}$. Dann existieren NFAs $\mathcal{A}^\cup$ und $\mathcal{A}^i$, so dass $L(\mathcal{A}^\cup) = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$ und $L(\mathcal{A}^i) = L(\mathcal{A}_1)L(\mathcal{A}_2)$.

Beweis. Übung.

Satz 1.5. Sei $\mathcal{A}$ ein NFA. Dann existiert ein NFA $\mathcal{A}^*$, so dass $L(\mathcal{A}^*) = L(\mathcal{A})^*$.

Beweis. Sei $\mathcal{A} = (Q, \Sigma, q_I, \delta, F)$ und q'_I ein neuer Zustand, der nicht in Q enthalten ist. Wir geben zuerst die Konstruktion von $\mathcal{A}^*$ an und beweisen dann, dass dieser NFA die gewünschte Sprache erkennt.

$$\mathcal{A}^* := (Q \cup \{q'_I\}, \Sigma, q'_I, \delta^*, F \cup \{q'_I\})$$

wobei für alle $a \in \Sigma$ und alle $q \in Q \cup \{q'_I\}$:

$$\delta^*(q, a) := \begin{cases} \delta(q_I, a) & , \text{ falls } q = q'_I \\ \delta(q, a) \cup \delta(q_I, a) & , \text{ falls } q \in F \\ \delta(q, a) & , \text{ falls } q \in Q \setminus F \end{cases}$$

Wir beobachten, dass ein Zustand in $\mathcal{A}^*$ also dieselben Transitionen wie derselbe Zustand in $\mathcal{A}$ hat mit dem Unterschied, dass Endzustände zusätzlich

noch die Transitionen des ursprünglichen Anfangszustand haben. Der neue Anfangszustand ist eine Kopie des alten Anfangszustands, der aber von keinem anderen Zustand aus erreichbar ist.

Es bleibt zu zeigen, dass $L(\mathcal{A}^*) = L(\mathcal{A})^*$ gilt. Für den $\subseteq$ -Teil nehmen wir an, dass $w \in L(\mathcal{A}^*)$ für ein $w = a_1 \dots a_n \in \Sigma^*$ gilt. Also existiert ein akzeptierender Lauf $q'_I, q_1, \dots, q_n$ auf w . Aus obiger Beobachtung folgt sofort, dass es $i_0, i_1, \dots, i_m$ gibt, so dass

- $i_0 = 0$,
- $i_0 < i_1 < \dots < i_m$,
- $i_m = n$,
- für alle $j = 1, \dots, m$ ist $q_{i_{j-1}}, \dots, q_{i_j}$ ein akzeptierender Lauf von $\mathcal{A}$ auf $a_{i_{j-1}+1} \dots a_{i_j}$.

Somit ist $w \in L(\mathcal{A})^*$.

Für den $\supseteq$ -Teil sei $w \in L(\mathcal{A})^*$. Falls $w = \varepsilon$, so gilt $w \in L(\mathcal{A}^*)$, da der Anfangszustand von $\mathcal{A}^*$ auch Endzustand ist. Sei $w \neq \varepsilon$. Also existiert eine Zerlegung $w = v_0 \dots v_k$, so dass $v_i \neq \varepsilon$ und $v_i \in L(\mathcal{A})$ für alle $i = 0, \dots, k$.

Wir zeigen nun durch Induktion über k , dass $v_0 \dots v_k \in L(\mathcal{A}^*)$ gilt. Im Induktionsanfang ist $k = 0$, d.h. $w = v_0$. Da $v_0 \in L(\mathcal{A})$ gibt es einen akzeptierenden Lauf $q_0, q_1, \dots, q_m$ von $\mathcal{A}$ auf v_0 mit $q_m \in F$. Da $v_0 \neq \varepsilon$, gilt $m \geq 1$. Jetzt ist aber $q'_I, q_1, \dots, q_m$ auch ein akzeptierender Lauf von $\mathcal{A}^*$ auf v_0 , da der neue Anfangszustand q'_0 dieselben Transitionen wie der alte q_0 hat und q_m auch weiterhin Endzustand in $\mathcal{A}^*$ ist.

Sei nun $k > 0$ und $v_0 \dots v_{k-1} \in L(\mathcal{A}^*)$, d.h. es gibt einen akzeptierenden Lauf $q'_I, q_1, \dots, q_m$ von $\mathcal{A}^*$ auf $v_0 \dots v_{k-1}$. Da $v_k \in L(\mathcal{A})$ gibt es auch einen akzeptierenden Lauf von $p_0, \dots, p_l$ von $\mathcal{A}$ auf v_k . Insbesondere gilt $p_0 = q_I$, $p_l \in F$ und $l \geq 1$, da $v_k \neq \varepsilon$. Dann ist aber $q'_I, q_1, \dots, q_m, p_1, \dots, p_l$ ein akzeptierender Lauf von $\mathcal{A}^*$ auf $v_0 \dots v_k$, denn es gilt $p_1 \in \delta^*(q_m, v_k(0))$, da $p_1 \in \delta(q_I, v_k(0))$. $\square$

Wir erinnern noch an das fundamentale Resultat, dass die von NFAs erkannten Sprachen *genau* die regulären Sprachen sind. D.h. zu jeder regulären Sprache kann man einen NFA angeben, und umgekehrt lässt sich zu jedem NFA auch zeigen, dass die von ihm erkannte Sprache regulär laut Def. 1.1 ist. Die erste Richtung ist recht einfach; die Hauptteile des Beweises haben wir bereits gesehen.

Satz 1.6. *Ist L regulär, so gibt es einen NFA $\mathcal{A}$ mit $L(\mathcal{A}) = L$.*

Beweis. Übung.

Für die Umkehrung dieses Satzes braucht man dazu z.B. das *Arden'sche Lemma*. Den Beweis stellen wir wieder als Übungsaufgabe.

Satz 1.7. *Seien $U, V \subseteq \Sigma^*$ Sprachen, so dass $\varepsilon \notin U$, und $L \subseteq \Sigma^*$ eine Sprache, so dass $L = UL \cup V$. Dann gilt $L = U^*V$.*

Beweis. Übung.

Dieser Satz ist das Werkzeug mit dem sich für die von einem NFA beschriebene Sprache ein regulärer Ausdruck finden lässt. Der Trick dabei ist, einen NFA mit n Zuständen als rekursives Gleichungssystem mit n Gleichungen zu betrachten, welches man sukzessive durch Anwendung des Arden'schen Lemmas lösen kann.

Satz 1.8. *Wird L von einem NFA erkannt, so ist L regulär.*

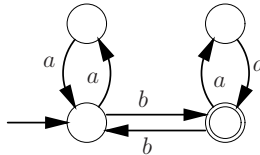
Beweis. Sei $L = L(\mathcal{A})$ für einen NFA $\mathcal{A} = (Q, \Sigma, q_I, \delta, F)$. O.B.d.A. nehmen wir an, dass $Q = \{0, \dots, n-1\}$ und $q_I = 0$ gilt.

Für jedes $i = 0, \dots, n-1$ definieren wir nun die Sprache X_i bestehend aus all den Wörtern, für die es einen in i beginnenden akzeptierenden Lauf in $\mathcal{A}$ gibt. Formell also $X_i = L(Q, \Sigma, i, \delta, F)$. Beachte, dass $L = X_0$. Diese Sprachen genügen nun dem folgenden Gleichungssystem:

$$X_i = \left(\bigcup_{a \in \Sigma} \bigcup_{j \in \delta(i, a)} \{a\} X_j \right) \cup \begin{cases} \{\varepsilon\} & , \text{ falls } i \in F \\ \emptyset & , \text{ sonst} \end{cases}$$

Mithilfe von offensichtlichen Transformationen und sukzessiver Anwendung des Arden'schen Lemmas können nun Ausdrücke für die X_i bestimmt werden, die zeigen, dass diese jeweils nach Def. 1.1 regulär sind. Da $L = X_0$, gilt dies auch für L selbst. $\square$

Beispiel 1.9. Als Beispiel betrachten wir den folgenden NFA.



Das dazugehörige Gleichungssystem—unter Auslassung einiger Klammern—lautet wie folgt.:

$$\begin{aligned} X_0 &= aX_1 \cup bX_2 & \text{(I)} \\ X_1 &= aX_0 & \text{(II)} \\ X_2 &= aX_3 \cup bX_0 \cup \{\varepsilon\} & \text{(III)} \\ X_3 &= aX_2 & \text{(IV)} \end{aligned}$$

Einsetzen von (IV) in (III) liefert

$$X_2 = aaX_2 \cup bX_0 \cup \varepsilon \quad \text{(V)}$$

also

$$X_2 = (aa)^*(bX_0 \cup \varepsilon) \quad \text{(VI)}$$

Einsetzen von (VI) und (II) in (I) liefert

$$\begin{aligned} X_0 &= aaX_0 \cup b(aa)^*(bX_0 \cup \varepsilon) \\ X_0 &= (aa \cup b(aa)^*b)X_0 \cup b(aa)^* \\ X_0 &= (aa \cup b(aa)^*b)^*b(aa)^* \end{aligned}$$

wobei sich die zweite Gleichung durch Vereinfachung und die dritte Gleichung wiederum mit dem Arden'schen Lemma ergibt. Damit ist X_0 regulär.

1.1.3 Deterministische, endliche Automaten

Definition 1.10. Ein NFA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ heißt *deterministisch* (DFA), falls für alle $q \in Q$ und alle $a \in \Sigma$ gilt: $|\delta(q, a)| = 1$. In diesem Fall schreiben wir auch einfach $\delta(q, a) = p$ statt $\delta(q, a) = \{p\}$.

In einigen Anwendungsfällen sind nichtdeterministische Automaten nicht ausreichend, sondern man braucht deterministische. Daher stellt sich die Frage, ob man zu jedem nichtdeterministischen Automaten oder—äquivalenterweise—jeder regulären Sprache auch einen deterministischen Automaten finden kann, der diese erkennt. Die Antwort ist bekanntermaßen “ja”, und sie wird typischerweise durch eine als *Potenzmengenkonstruktion* bekannt gewordene Umwandlung eines NFA in einen äquivalenten DFA gezeigt.

Satz 1.11. *Wird eine Sprache L von einem NFA mit n Zuständen erkannt, so wird L auch von einem DFA mit höchstens 2^n vielen Zuständen erkannt.*

Beweis. Sei $L = L(\mathcal{A})$ für einen NFA $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$. Dann sei $\mathcal{P} := (2^Q, \Sigma, \{q_0\}, \Delta, F')$ wobei für jedes $S \subseteq Q$ und jedes $a \in \Sigma$.

$$\Delta(S, a) := \bigcup_{q \in S} \delta(q, a)$$

und $F' := \{S \subseteq Q \mid F \cap S \neq \emptyset\}$.

Man beachte, dass $\mathcal{P}$ in der Tat ein DFA ist: Vom Zustand S erreicht man unter Lesen von a genau einen Zustand, auch wenn die Transitionsfunktion hier als Vereinigungsmenge geschrieben ist. Zustände sind allerdings Mengen von Zuständen aus $\mathcal{A}$, weswegen solch eine Vereinigungsmenge einen Zustand darstellt.

Es bleibt zu zeigen, dass $L(\mathcal{P}) = L(\mathcal{A})$ gilt. Für die $\supseteq$ -Richtung existiere ein akzeptierender Lauf $q_0, \dots, q_n$ von $\mathcal{A}$ auf einem Wort $w = a_0, \dots, a_{n-1}$. Da $\mathcal{P}$ deterministisch ist, existiert ein eindeutiger Lauf $S_0, \dots, S_n$ von $\mathcal{P}$ auf w . Es gilt offensichtlich $q_0 \in S_0$, da $S_0 = \{q_0\}$. Darüberhinaus wird die folgende Invariante für alle $i = 0, \dots, n-1$ bewahrt: Ist $q_i \in S_i$, so ist $q_{i+1} \in S_{i+1}$. Dies ist der Fall, weil $q_{i+1} \in \delta(q_i, a_i)$ für alle diese i gilt. Somit gilt letztendlich auch $q_n \in S_n$, und da $q_n \in F$ auch $F \cap S_n \neq \emptyset$, weswegen $S_0, \dots, S_n$ ein akzeptierender Lauf von $\mathcal{P}$ auf w ist.

Für die $\subseteq$ -Richtung sei $S_0, \dots, S_n$ ein akzeptierenden Lauf von $\mathcal{P}$ auf einem Wort $w = a_0 \dots a_{n-1}$. Falls $n = 0$, also $w = \varepsilon$, so muss $S_0 \cap F \neq \emptyset$ gelten, was aber $q_0 \in F$ bedeutet, weswegen $\mathcal{A}$ auch w akzeptiert.

Sei also $n > 0$. Beachte, dass jeder $\mathcal{A}$ -Zustand q in einem S_i in gewissem Sinne mit einem $q' \in S_{i-1}$ verbunden ist: Für jedes $i = 1, \dots, n$ und jedes $q \in S_i$ existiert ein $q' \in S_{i-1}$, so dass $q \in \delta(q', a_{i-1})$. Dies ist eine sofortige Konsequenz aus der Definition von Δ . Dies bedeutet aber auch, dass sich für jedes $i = 1, \dots, n$ und jedes $q \in S_i$ ein Lauf von $\mathcal{A}$ auf dem Wort $a_0 \dots a_{i-1}$ konstruieren lässt, der in q endet. Da $S_n \cap F \neq \emptyset$, gibt es ein $q \in S_n$ mit $q \in F$. Die Anwendung dieser Beobachtung auf dieses q und $i = n$ liefert dann einen akzeptierenden Lauf von $\mathcal{A}$ auf w . $\square$

Da jeder DFA auch ein NFA per Definition ist, folgt also, dass die regulären Sprachen nicht nur genau diejenigen sind, die durch NFAs erkannt werden, sondern ebenfalls genau diejenigen, die durch DFAs erkannt werden.

Eine wichtige Konsequenz aus der Tatsache, dass DFAs ausreichen, um reguläre Sprachen zu beschreiben, ist der Komplementabschluss der Klasse der regulären Sprachen. Beachte, dass zu einer gegebenen, formalen Beschreibung α nicht ohne weiteres eine Beschreibung $\bar{\alpha}$ für das Komplement der Sprache gefunden werden kann. Dasselbe gilt für NFAs, da das Komplementieren aus der existentiellen Quantifizierung über Läufe eine universelle Quantifizierung machen würde. Diese kann aber i.A. nicht wieder als existentielle Quantifizierung beschrieben und somit mit einem NFA erkannt werden. Da DFAs aber auf einem gegebenen Wort jeweils einen eindeutigen Lauf haben, sind bei diesem Automatenmodell die Aussagen “es gibt einen Lauf auf w ” und “für alle Läufe auf w ” äquivalent.

Satz 1.12. *Für alle $L \subseteq \Sigma^*$ gilt: $L \in \text{REG} \Rightarrow \bar{L} \in \text{REG}$.*

Beweis. Sei $L \in \text{REG}$. Laut Satz 1.6 gibt es dann einen NFA $\mathcal{A}$ mit $L(\mathcal{A}) = L$. Nach Satz 1.11 lässt sich dieser per Potenzmengenkonstruktion in einen äquivalenten DFA $\mathcal{B}$ umwandeln. Es gilt also auch $L(\mathcal{B}) = L$.

Sei nun $\mathcal{B} = (Q, \Sigma, q_0, \delta, F)$. Definiere $\bar{\mathcal{B}} := (Q, \Sigma, q_0, \delta, Q \setminus F)$. Es bleibt zu zeigen, dass $L(\bar{\mathcal{B}}) = \bar{L}(\mathcal{B})$ gilt.

“ $\subseteq$ ” Wir nehmen an, dass $w \in L(\bar{\mathcal{B}})$ gilt für ein beliebiges $w \in \Sigma^*$. Da $\mathcal{B}$ deterministisch ist, gibt es einen eindeutigen Lauf $q_0, \dots, q_n$ von $\mathcal{B}$ auf w , so dass $q_n \in Q \setminus F$. Beachte, dass $Q \setminus F$ die Endzustände von $\bar{\mathcal{B}}$ sind. Da $\mathcal{B}$ und $\bar{\mathcal{B}}$ aber dieselben Anfangszustände und Transitionsfunktion haben, ist $q_0, \dots, q_n$ auch ein Lauf von $\mathcal{B}$ auf w . Dieser endet aber in einem Nichtendzustand bzgl. $\mathcal{B}$. Da ein DFA höchstens einen Lauf auf einem gegebenen Wort haben kann, gibt es keinen akzeptierenden Lauf von $\mathcal{B}$ auf w , und somit gilt $w \notin L(\mathcal{B})$ bzw. $w \in \bar{L}(\mathcal{B})$.

“ $\supseteq$ ” Wir benutzen die $\subseteq$ -Richtung. Beachte, dass $\bar{\bar{\mathcal{B}}}$ wiederum ein DFA ist, für den dann auch die $\subseteq$ -Richtung, also $L(\bar{\bar{\mathcal{B}}}) \subseteq \bar{L}(\bar{\mathcal{B}})$ gelten muss. Da $F \subseteq Q$ gilt auch $Q \setminus (Q \setminus F) = F$. Das bedeutet einfach, dass zweimal Komplementieren wieder den alten Automaten herstellt, also $\bar{\bar{\mathcal{B}}} = \mathcal{B}$, woraus $L(\mathcal{B}) \subseteq \bar{L}(\bar{\mathcal{B}})$ folgt. Dies ist aber äquivalent dazu, dass $\bar{L}(\bar{\mathcal{B}}) \supseteq \bar{\bar{L}(\mathcal{B})} = L(\mathcal{B})$ gilt, was noch zu beweisen war.

Somit ist $\overline{\mathcal{B}}$ auch ein NFA und nach Satz 1.8 und ist $L(\overline{\mathcal{B}}) = \overline{L}$ auch eine reguläre Sprache. $\square$

Sei L also eine reguläre Sprache, die von einem NFA mit n Zuständen erkannt wird. Dann lässt sich zwar prinzipiell ein DFA für diese Sprache und damit auch für $\overline{L}$ konstruieren. Die Anzahl seiner Zustände ist im allgemeinen aber exponentiell in n , d.h. sie kann nur durch 2^n nach oben abgeschätzt werden. Man kann sogar zeigen, dass dies optimal ist, d.h., dass es Sprachen L_n , $n \in \mathbb{N}$ gibt, die von einem NFA mit $n+1$ Zuständen, aber nicht von einem DFA mit weniger als 2^n Zuständen erkannt werden.

Beachte, dass das Verfahren im obigen Beweis so präsentiert ist, dass der resultierende DFA immer 2^n Zustände hat. Dies lässt sich jedoch verbessern, indem man nur den Teil des DFA konstruiert, der vom Anfangszustand $\{q_0\}$ aus erreichbar ist. In den meisten Fällen erhält man so einen DFA mit wesentlich weniger als 2^n Zuständen.

Auch ist es möglich, die Definition eines DFA leicht abzuschwächen, indem man verlangt, dass es zu jedem Zustand und jedem Alphabetsymbol *höchstens* statt *genau* einen Nachfolgezustand gibt. Man überlegt sich leicht, dass jeder Automat durch Hinzunahme eines einzigen Zustands in einen umgewandelt werden, der zu jedem Zustand und Alphabetsymbol einen Nachfolger hat, ohne dass dabei die erkannte Sprache verändert wird.

1.2 Entscheidbarkeit und Komplexität

Zum Abschluss dieses Kapitels betrachten wir noch die algorithmische Handhabbarkeit der regulären Sprachen. Dazu definieren wir mehrere Entscheidungsprobleme.

- Das *Leerheitsproblem* für reguläre Sprachen ist das folgende. Gegeben ist eine reguläre Sprache, repräsentiert durch einen NFA oder DFA $\mathcal{A}$, entscheide, ob $L(\mathcal{A}) = \emptyset$ ist.
- Das *Universalitätsproblem* fragt, ob $L(\mathcal{A}) = \Sigma^*$ für einen gegebenen NFA $\mathcal{A}$ gilt.
- Das *Wortproblem* fragt zu gegebenem $\mathcal{A}$ und gegebenem Wort $w \in \Sigma^*$, ob $w \in L(\mathcal{A})$ gilt.

Ersteres ist im Grunde das wichtigste Entscheidungsproblem für reguläre Sprachen, da sich viele andere Fragen—so auch Universalitäts- und Wortproblem—auf dieses zurückführen lassen.

Bevor wir zeigen, dass diese Probleme entscheidbar sind, machen wir noch zwei Bemerkungen. Erstens kann die Komplexität solcher (oder ähnlicher) Entscheidungsprobleme natürlich von der Art und Weise, wie L repräsentiert ist, abhängen. Zweitens interessiert man sich oft für komplementäre Probleme. So will man z.B. wissen, ob ein gegebener NFA eine *nichtleere* Sprache beschreibt. Genau genommen handelt es sich dabei um das *Nichtleerheitsproblem*. Im folgenden werden wir diese beiden aber nicht weiter unterscheiden.

Der Grund dafür ist, dass wir bei solchen Entscheidungsverfahren meistens an deterministischen Verfahren interessiert sind, und bei solchen macht es prinzipiell keinen Unterschied, ob man das gegebene Problem oder sein Komplement löst.

Satz 1.13. *Das Leerheitsproblem für NFAs mit n Zuständen lässt sich in Zeit $\mathcal{O}(n^2)$ lösen.*

Beweis. Sei $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ mit $|Q| = n$. Man kann $\mathcal{A}$ als gerichteten und kantenbeschrifteten Graphen mit Knotenmenge Q und Kantenrelation $q \xrightarrow{a} q'$, genau dann wenn $q' \in \delta(q, a)$, auffassen. Mithilfe einer Breiten- oder Tiefensuche, die alle von q_0 aus erreichbaren Zustände markiert, lässt sich in Zeit $\mathcal{O}(n^2)$ feststellen, ob es einen Zustand $q \in F$ gibt, welcher von q_0 aus erreichbar ist. Beachte, dass Breiten- oder Tiefensuche im worst-case linear in der Anzahl der Kanten eines Graphen sind. Diese wiederum kann natürlich quadratisch in der Anzahl der Knoten des Graphen sein.

Ist ein Endzustand in diesem Verfahren markiert worden, so ist $L(\mathcal{A}) \neq \emptyset$, denn der Pfad von q_0 nach q beschreibt einen akzeptierenden Lauf von $\mathcal{A}$ auf dem Wort, welches durch Konkatenation der einzelnen Kantenbeschriftungen entlang dieses Pfades entsteht. Wird in der Suche kein Endzustand markiert, so muss $L(\mathcal{A}) = \emptyset$ sein, dann kein Lauf von $\mathcal{A}$ auf irgendeinem Wort kann in einem Endzustand enden. $\square$

Das Wortproblem ist ebenfalls entscheidbar. Man kann im Prinzip ein ähnliches Verfahren verwenden, in dem der NFA $\mathcal{A}$ wieder als gerichteter Graph aufgefasst wird. Allerdings wird die Tiefen- oder Breitensuche so modifiziert, dass sie in Tiefe i nur noch diejenigen Nachfolger als erreichbar markiert, die Nachfolger eines erreichbaren Zustands bzgl. des i -ten Symbols des Eingabeworts sind. Außerdem darf die Suche natürlich nicht abbrechen, wenn in einer Iteration keine neuen erreichbaren Zustände gefunden wurden, sondern sie muss solange weitersuchen, bis alle in Tiefe $|w|$ erreichbaren Zustände gefunden wurden. Insgesamt ergibt sich so eine Laufzeit von $\mathcal{O}(n^2 \cdot |w|)$, falls $\mathcal{A}$ genau n Zustände hat.

Die folgenden Probleme sind für gegebene endliche Automaten $\mathcal{A}, \mathcal{B}$ ebenfalls entscheidbar.

- a) Das *Schnittproblem*: Ist $L(\mathcal{A}) \cap L(\mathcal{B}) = \emptyset$?
- b) Das *Äquivalenzproblem*: Ist $L(\mathcal{A}) = L(\mathcal{B})$?
- c) Das *Subsumptionsproblem*: Ist $L(\mathcal{A}) \subseteq L(\mathcal{B})$?

Automatentheorie und Logik

Hofmann, M.; Lange, M.

2011, X, 238 S. 42 Abb., Softcover

ISBN: 978-3-642-18089-7