

Einführung in die Theoretische Informatik

Sommersemester 2024 – Übungsblatt 2

- **Tipp:** Auf dieser (und dort verlinkten) Website(n) können Sie interaktiv DFAs, NFAs, reguläre Ausdrücke,... erzeugen, simulieren, umwandeln, ...
Beachten Sie, dass auf der Website \$ statt ϵ verwendet wird.
- Das Übungsblatt ist in zwei Teile gegliedert: den Vorbereitungsteil, den Sie vor der Übung selbstständig bearbeiten sollen, und den Übungs-/Nachbereitungsteil, der Aufgaben enthält, die in der Übung besprochen werden und von Ihnen anschließend zur Nachbereitung verwendet werden können.
- Für den Rest des Semesters gilt: $\mathbb{N} := \mathbb{N}_0 := \{0, 1, 2, \dots\}$ und $\mathbb{N}_+ := \mathbb{N}_0 \setminus \{0\}$.

Vorbereitung (→ vor der Übung selbstständig zu bearbeiten)

Vorbereitungsaufgabe Ü2.1. (Wichtige Begriffe)

Überprüfen Sie, dass Sie die folgenden Begriffe oder Notationen korrekt definieren können.

- NFA und deren Akzeptanzbedingung
- Rechtslineare Grammatik
- ϵ -NFA
- Potenzmengenkonstruktion
- Regulärer Ausdruck

Vorbereitungsaufgabe Ü2.2. (Automata Tutor: NFAs und Reguläre Ausdrücke)

Lösen Sie die Aufgaben Ü2.2 (a–i) auf Automata Tutor. Beachten Sie, dass wir für einen regulären Ausdruck r das folgende Makro definieren: $r^+ := rr^*$.

Vorbereitungsaufgabe Ü2.3. (Konstruieren von NFAs mit Einschränkungen)

Entscheiden Sie, ob die folgenden Aussagen korrekt sind, und begründen Sie Ihre Behauptung, indem Sie entweder ein Gegenbeispiel oder eine passende Konstruktion angeben.

Für jeden NFA $N = (Q, \Sigma, \delta, q_0, F)$ gibt es einen NFA $N' = (Q', \Sigma, \delta', q'_0, F')$ mit $L(N) = L(N')$ und ...

- der Startzustand hat keine eingehenden Kanten.
- kein Endzustand hat eine ausgehende Kante.
- für jeden Zustand q gilt: alle eingehenden Kanten von q sind mit demselben Zeichen beschriftet.
- für jeden Zustand q gilt: alle ausgehenden Kanten von q sind mit demselben Zeichen beschriftet.

Zu dieser Aufgabe gibt es Video-Lösungen: (a) (b) (c) (d)

Übung und Nachbereitung

Übungsaufgabe Ü2.4. (*Tick Tock Boom, Blow Up*)

Mit $|w|_x$ bezeichnen wir die Anzahl der Vorkommen des Buchstabens $x \in \Sigma$ in $w \in \Sigma^*$. Sei $\Sigma = \{a, b, c\}$ und $L = \{w \in \Sigma^* \mid \exists x \in \Sigma. |w|_x = 0\}$.

- Konstruieren Sie einen NFA N mit genau 4 Zuständen und $L(N) = L$.
- Determinisieren Sie den NFA N aus (a) mittels der Potenzmengenkonstruktion um einen DFA D mit $L(D) = L(N)$ zu erhalten.
- Geben Sie eine rechtslineare Grammatik G (gemäß Satz 3.13) an, sodass $L(G) = L(D)$.
- Übersetzen Sie die Grammatik G (gemäß Satz 3.9) in einen NFA N' , sodass $L(N') = L(G)$.
- Vergleichen Sie die NFAs N' und N bezüglich Zustands- und Transitionszahl. Diskutieren Sie dann, inwiefern Sie Ihre Beobachtung verallgemeinern können.

Übungsaufgabe Ü2.5. (*Teilwörter und reguläre Sprachen*)

Ein Teilwort eines Wortes w ist ein zusammenhängendes Wort in w . Wir definieren die Menge aller Teilwörter von w als $\downarrow[w] := \{w' \in \Sigma^* \mid w \text{ enthält } w'\}$. Also gilt beispielsweise $the, he, theo, \epsilon \in \downarrow[theo]$, aber $to, theoo \notin \downarrow[theo]$. Die Menge aller Wörter, von denen w ein Teilwort ist, ist dann entsprechend definiert als $\uparrow[w] := \{w' \in \Sigma^* \mid w' \text{ enthält } w\}$; z.B. $thetheotee \in \uparrow[theo]$. Wir erweitern dies auf Sprachen: Für eine beliebige Sprache $L \subseteq \Sigma^*$ definieren wir also $\downarrow[L] := \bigcup_{w \in L} \downarrow[w]$ und $\uparrow[L] := \bigcup_{w \in L} \uparrow[w]$.

Sei nun $\Sigma := \{t, h, e, o\}$ und L eine beliebige reguläre Sprache.

- Geben Sie für $\downarrow[theo]$ und $\uparrow[theo]$ einen ϵ -NFA an.
- Zeigen Sie, dass $\downarrow[L]$ regulär ist.
- Zeigen Sie, dass $\uparrow[L]$ regulär ist.

Übungsaufgabe Ü2.6.

In dieser Aufgabe beschäftigen wir uns mit rekursiven Funktionen und Beweisen auf regulären Ausdrücken.

- Geben Sie eine rekursive Funktion $\text{empty}(r)$ an, die für einen gegebenen regulären Ausdruck r entscheidet, ob $L(r) = \emptyset$. Für Ihre Definition sollten Sie das folgende Gerüst verwenden:
 - $\text{empty}(\emptyset) =$
 - $\text{empty}(a) =$
 - $\text{empty}(\epsilon) =$
 - $\text{empty}(\alpha\beta) =$
 - $\text{empty}(\alpha \mid \beta) =$
 - $\text{empty}(\alpha^*) =$

Beweisen Sie mittels struktureller Induktion, dass Ihre Definition korrekt ist.

Erinnerung: Da reguläre Ausdrücke induktiv definiert sind, sind sie mit einem Induktionsschema ausgestattet. Um zu beweisen, dass eine Eigenschaft $P(r)$ für alle regulären Ausdrücke r gilt, können wir dieses Induktionsschema verwenden. Bei Anwendung des Schemas ergeben sich dann folgende Beweisaufgaben:

- Zeige $P(\emptyset)$.

- Zeige $P(\epsilon)$.
- Zeige $P(a)$ für alle $a \in \Sigma$.
- Unter der Annahme $P(\alpha)$ und $P(\beta)$, zeige $P(\alpha\beta)$.
- Unter der Annahme $P(\alpha)$ und $P(\beta)$, zeige $P(\alpha | \beta)$.
- Unter der Annahme $P(\alpha)$, zeige $P(\alpha^*)$.

Dieses Prinzip, welches für alle induktiv definierte Strukturen gilt, wird *strukturelle Induktion* genannt.

(b) **(optionale Teilaufgabe)** Gegeben sein nun folgende Funktion, die bestimmen soll, ob die von einem regulärem Ausdruck erzeugte Sprache unendlich viele Wörter enthält. In anderen Worten soll gelten, dass $\text{infinite}(r) \iff |L(r)| = |\mathbb{N}|$.

- | | |
|---|---|
| • $\text{infinite}(\emptyset) = \text{false}$ | • $\text{infinite}(\alpha\beta) = \text{infinite}(\alpha) \vee \text{infinite}(\beta)$ |
| • $\text{infinite}(a) = \text{false}$ | • $\text{infinite}(\alpha \beta) = \text{infinite}(\alpha) \vee \text{infinite}(\beta)$ |
| • $\text{infinite}(\epsilon) = \text{false}$ | • $\text{infinite}(\alpha^*) = \text{true}$ |

Beweisen Sie, dass die obige Definition korrekt ist oder widerlegen Sie die Korrektheit mit einem Gegenbeispiel.