Prof. Javier Esparza
Kevin Kappelmann, Simon Roßkopf, Lukas Stevens

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2023 – Übungsblatt 2

- **Tipp:** Auf dieser (und dort verlinkten) Website(n) können Sie interaktiv DFAs, NFAs, reguläre Ausdrücke,... erzeugen, simuliereren, umwandeln,...
- 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: $\mathbb{N} \coloneqq \mathbb{N}_0 \coloneqq \{0, 1, 2, ...\}$ und $\mathbb{N}_+ \coloneqq \mathbb{N}_0 \setminus \{0\}$.

Vorbereitung ($\rightarrow$ vor der Übung selbständig zu bearbeiten)

Individualaufgabe Ü2.1. (*Wichtige Begriffe*)

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

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

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

Lösen Sie die Aufgaben Ü2.2 (a–i) auf Automata Tutor.[1] Beachten Sie, dass wir für einen regulären Ausdruck r das folgende Makro definieren: $r^+ \coloneqq rr^*$.

Individualaufgabe 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 ...

(a) der Startzustand hat keine eingehenden Kanten.

(b) kein Endzustand hat eine ausgehende Kante.

(c) für jeden Zustand q gilt: alle eingehenden Kanten von q sind mit demselben Zeichen beschriftet.

(d) 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)

[1] Wenn Sie Automata Tutor noch nicht verwendet haben, folgen Sie erst den Schritten in Ü1.2, um sich richtig zu registrieren.

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 $L = \{w \in \Sigma^* \mid \exists\, x \in \Sigma.\ |w|_x = 0\}$.

(a) Konstruieren Sie einen NFA N mit genau 4 Zuständen und $L(N) = L$.

(b) Determinisieren Sie den NFA N aus (a) mittels der Potenzmengenkonstruktion um einen DFA D mit $L(D) = L(N)$ zu erhalten.

(c) Geben Sie eine rechtslineare Grammatik G (gemäß Satz 3.13) an, sodass $L(G) = L(D)$.

(d) Übersetzen Sie die Grammatik G (gemäß Satz 3.9) in einen NFA N', sodass $L(N') = L(G)$.

(e) 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 $\downarrow[w] := \{w' \in \Sigma^* : w \text{ enthält } w'\}$. Also gilt beispielsweise $the, he, theo, \varepsilon \in \downarrow[theo]$, aber $to, theoo \notin \downarrow[theo]$. Die Menge aller Wörter, von denen w ein Teilwort ist, ist dann entsprechend definiert als $\uparrow[w] := \{w' \in \Sigma^* : w' \text{ enthält } w\}$; 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.

(a) Geben Sie für $\downarrow[theo]$ und $\uparrow[theo]$ einen ϵ-NFA an.

(b) Zeigen Sie, dass $\downarrow[L]$ regulär ist.

(c) Zeigen Sie, dass $\uparrow[L]$ regulär ist.

Übungsaufgabe Ü2.6.

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

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

- $\mathsf{empty}(\emptyset) =$
- $\mathsf{empty}(\mathsf{a}) =$
- $\mathsf{empty}(\boldsymbol{\epsilon}) =$
- $\mathsf{empty}(\alpha\beta) =$
- $\mathsf{empty}(\alpha \mid \beta) =$
- $\mathsf{empty}(\alpha^*) =$

Beweisen Sie mittels struktureller Induktion, dass Ihre Definition korrekt ist. Sie müssen nur den Fall Konkatenation (also $r = \alpha\beta$) behandeln.

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 $P(\boldsymbol{\epsilon})$.

- Zeige $P(\mathsf{a})$ für alle $\mathsf{a} \in \Sigma$.
- Unter der Annahme $P(\alpha)$ und $P(\beta)$, zeige $P(\alpha\beta)$.
- Unter der Annahme $P(\alpha)$ und $P(\beta)$, zeige $P(\alpha \mid \beta)$.
- 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 $\mathsf{infinite}(r) \iff |L(r)| = |\mathbb{N}|$.

- $\mathsf{infinite}(\emptyset) = \mathsf{false}$
- $\mathsf{infinite}(\mathsf{a}) = \mathsf{false}$
- $\mathsf{infinite}(\epsilon) = \mathsf{false}$
- $\mathsf{infinite}(\alpha\beta) = \mathsf{infinite}(\alpha) \vee \mathsf{infinite}(\beta)$
- $\mathsf{infinite}(\alpha \mid \beta) = \mathsf{infinite}(\alpha) \vee \mathsf{infinite}(\beta)$
- $\mathsf{infinite}(\alpha^*) = \mathsf{true}$

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