Prof. Javier Esparza
Philipp Czerner, Kevin Kappelmann, Valentin Krasotin

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2024 – Hausaufgabenblatt 1

- Beachten Sie die ergänzten Abgaberichtlinien für die Hausaufgaben auf der Vorlesungswebsite!
- Bitte beachten Sie, dass in dieser Vorlesung generell Antworten mit Begründung gefordert werden, solange die Aufgabe nicht explizit das Gegenteil sagt.

AT-Aufgabe H1.1. (*Wer a sagt, muss auch b sagen*) keine Korrektur

Hinweis: Diese Aufgabe wird mit Automata Tutor bearbeitet. AT ist ein Online-Tool, mit dem Sie zu vielen Themen aus dem THEO-Kurs Aufgaben bearbeiten können. Dabei bekommen Sie sogar automatisch individuelles Feedback! Dazu müssen Sie einen Account erstellen, und dann den Link verwenden, um sich in den Kurs einzuschreiben. Es gibt keine Verknüpfung zwischen AT und TUM-internen Systemen, sie können (sollten) also insbesondere ein anderes Passwort verwenden. AT wurde in den letzten Jahren signifikant überarbeitet — wir hoffen, dass es bereits in einem Zustand ist, Ihnen mit der Wiederholung des Materials zu helfen. Wenn Sie Bugs finden, melden Sie diese bitte auf Zulip.

Der kleine Theo hat gerade das Sprichwort „Wer A sagt, muss auch B sagen“ zum ersten Mal gehört und sofort entschieden, es zu seinem Lebensmotto zu machen. Leider weiß Theo nicht genau, was dieses Sprichwort bedeutet. Er ist sich lediglich sicher, dass es sich darum handelt, nur bestimmte Wörter zu sagen, kann aber nicht genau feststellen, welche Wörter das sind. Nun möchte er einen Automaten bauen, um überprüfen zu können, welche Wörter das Sprichwort befolgen. Sei $\Sigma := \{a, b, x\}$. Helfen Sie Theo, indem Sie einen DFA über dem Alphabet Σ angeben, der die Sprache aller Wörter erkennt,

(a) in denen auf jedes a sofort ein b folgt.

(b) in denen auf jedes a irgendwann ein b folgt.

(c) für die gilt: Wenn ein a vorkommt, dann kommt auch ein b vor.

Aufgabe H1.2. (*Fragwürdige Wahrheiten*) Korrektur-Prio. 3

Beim Chat mit einem textbasierten Onlinedialogsystem™ erhält Dora die folgenden, angeblich gültigen, Aussagen. Sie ist sich jedoch nicht sicher, ob sie dem System vertrauen kann... Sie bittet euch daher um Hilfe, die Behauptungen zu überprüfen.

Sei $\Sigma \neq \emptyset$ ein Alphabet, und $A, B, C \subseteq \Sigma^*$ Sprachen über Σ. Beweisen oder widerlegen Sie folgende Aussagen:

(a) $AA^* = A^* \implies \varepsilon \in A$

(b) $A \subseteq B \Leftrightarrow A^+ \subseteq B^+$

(c) $|A \times A| = |AA|$

(d) $A(BC) = (AB)C$

(e) $AB \setminus AC = A(B \setminus C)$

(f) $|AB| \geq |A|$

Hinweis: Für alle ungültigen Aussagen gibt es ein Gegenbeispiel, in dem jede Sprache höchstens zwei Wörter mit höchstens zwei Buchstaben hat.

Lösungsskizze. Es gibt 5 Punkte pro Teilaufgabe. In allen Gegenbeispielen verwenden wir $\Sigma := \{a, b\}$.

(a) Wahr. Sei $A^* = AA^*$. Da $\varepsilon \in A^*$ gilt, folgt auch $\varepsilon \in AA^*$. Deshalb folgt $\varepsilon = uv$ mit $u \in A, v \in A^*$. Dann muss $u = \varepsilon$ sein und daher auch $\varepsilon \in A$.

(b) Falsch. $A := \{aa\}, B := \{a\}$, womit $A^+ \subseteq B^+$ gilt, $A \subseteq B$ aber nicht.

(c) Falsch. Sei $A := \{\varepsilon, a\}$. Dann ist $|A \times A| = |\{(\varepsilon, \varepsilon), (\varepsilon, a), (a, \varepsilon), (a, a)\}| = 4$ aber $|AA| = |\{\varepsilon, a, aa\}| = 3$.

(d) Wahr. Nutze die Assoziativität von Konkatenation auf Wörtern:

$$\begin{aligned} u \in A(BC) &\iff u = v(xy) \text{ für } v \in A, x \in B, y \in C \\ &\iff u = (vx)y \text{ für } v \in A, x \in B, y \in C \\ &\iff u \in (AB)C \end{aligned}$$

(e) Falsch. $A := \{a, aa\}, B := \{ab\}, C := \{b\}$, also gelten $AB \setminus AC = \{aaab\}$ und $A(B \setminus C) = \{aab, aaab\}$.

(f) Falsch. $A := \{a\}, B := \emptyset$

Aufgabe H1.3. (*Kodierungen*) Korrektur-Prio. 2

Der kleine Theodor lernt gerade in der Schule Zahlen zu multiplizieren. Er will aber lieber malen als malnehmen und hat sich schon ein Programm geschrieben, das seine Hausaufgaben vollautomatisch löst. Er ist sich aber nicht sicher, ob die Ausgabe das richtige Format hat — können Sie ihm helfen?

Sei $\Sigma := \{\texttt{0}, ..., \texttt{9}\}$. Wir können eine Liste von Wörtern $w_1, ..., w_k$, mit $k \in \mathbb{N}$, über ein einziges Wort kodieren, indem wir vor jedes Wort w_i dessen Länge $n_i := |w_i|$ schreiben: $w := n_1 w_1\, n_2 w_2 ... n_k w_k$. Das funktioniert nur, wenn $w_1, ..., w_k$ höchstens Länge 9 haben. Ein so erzeugtes Wort w nennen wir *Präfix-Kodierung*. Zum Beispiel können wir $w_1 = \texttt{512}, w_2 = \varepsilon, w_3 = \texttt{2024}$ über $w = \texttt{3512042024}$ kodieren. Weitere Präfix-Kodierungen sind etwa `123456`, ε, oder `2020`. Die Wörter `1234`, `007` und `2024` sind dagegen keine Präfix-Kodierungen.

Wir betrachten nun die Sprache L aller Präfix-Kodierungen.

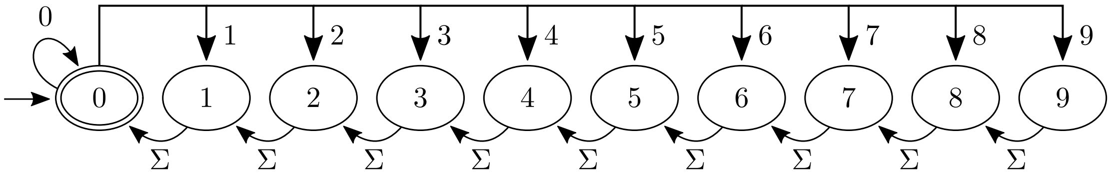
(a) Geben Sie einen deterministischen endlichen Automaten (DFA) an, der L akzeptiert. Erstellen Sie dafür sowohl eine Skizze als auch eine formale Beschreibung, in der Sie das Tupel $(Q, \Sigma, \delta, q_0, F)$ angeben. Ihr DFA darf höchstens 20 Zustände haben.

Hinweis: Beim Zeichnen von DFAs können sie eine Kante mit Σ beschriften, statt jedes Zeichen des Alphabets einzeln aufzulisten.

(b) Geben Sie einen mengentheoretischen Ausdruck für L an. Sie dürfen nur endliche Mengen verwenden und diese mit Hilfe von endlichen Vereinigungen, endlichen Schnitten, sowie den Operationen auf Sprachen aus Definition 2.3 kombinieren.

Lösungsskizze. Es gibt 5 Punkte pro Teilaufgabe.
(a) $Q := \Sigma$, $q_0 := \texttt{0}$, $F := \{\texttt{0}\}$,

$$\delta(q, c) := \begin{cases} c & \text{falls } q = 0 \\ q - 1 & \text{sonst} \end{cases}$$



(b) $L = \left(\bigcup_{i \in \Sigma} \{i\}\Sigma^i\right)^*$

Quizaufgabe H1.4. (*Verstehen Sie Sprachen?*) keine Korrektur

Sei $\Sigma \coloneqq \{a, b\}$, $A \coloneqq \{\varepsilon, a, ab\}$, $B \coloneqq \{b, ba\}$. Bestimmen Sie folgende Mengen:

(a) $AB \setminus (A \cup B)$ (b) $A^3\emptyset$ (c) $\emptyset^* A^0$ (d) $(A \times \emptyset)^* \times B$

Lösungsskizze.

(a) $\{aba, abb, abba\}$

(b) $\emptyset$

(c) $\{\varepsilon\}$

(d) $\{(\varepsilon, b), (\varepsilon, ba)\}$

Quizaufgabe H1.5. (*Buchstabenfhler*) keine Korrektur

Der Superschurke Dr. Evilsparza hat sich ins Campusnetzwerk gehackt und es ist ihm gelungen, wichtige Daten zu löschen! Sie sind Teil des Cybersecurityteams, das versucht das Ausmaß des Schadens zu bestimmen.

Sei $\Sigma \coloneqq \{a, b\}$ ein Alphabet.

(a) Sei G eine beliebige Grammatik, und G' eine Grammatik, die durch das Entfernen von einer beliebigen Produktion aus G entsteht. Beweisen oder widerlegen Sie $L(G') \subseteq L(G)$.

(b) Sei G wieder eine beliebige Grammatik, und G' eine Grammatik, die man erhält, wenn man von G ein beliebiges Nichtterminalzeichen aus der rechten Seite einer beliebigen Produktion löscht. Beweisen oder widerlegen Sie: $L(G') \subseteq L(G)$

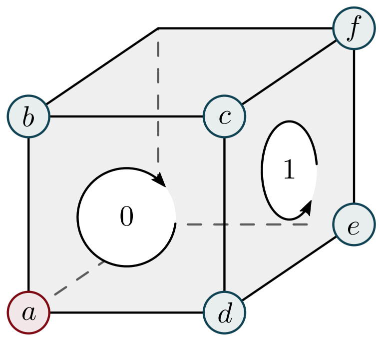
Lösungsskizze.

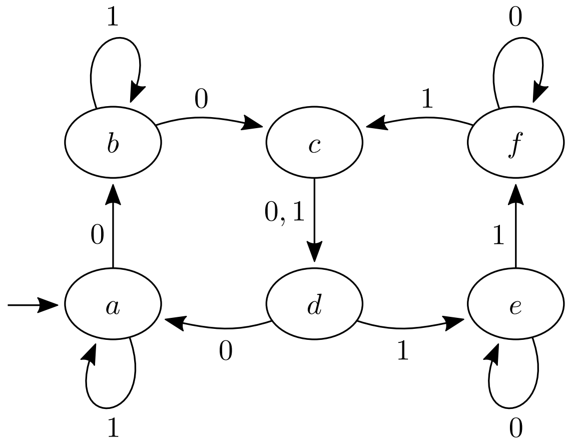
(a) Die Aussage ist wahr: Sei $w \in L(G')$ beliebig, und S das Startsymbol von G. Nach Definition von $L(G')$ gilt also $S \to_{G'}^* w$. Da wir für G' aber nur eine Produktion entfernt haben, gilt $S \to_G^* w$ ebenfalls, und somit $w \in L(G)$.

(b) Falsch! Ein Gegenbeispiel wäre $G = (\{S\}, \Sigma, P, S)$ mit $P = \{S \to aS\}$. Es ist nicht möglich, in G ein Wort zu produzieren, das nur aus Terminalzeichen besteht, also $L(G) = \emptyset$. Wenn wir allerdings die Produktionsregel zu $S \to a$ verändern, erhalten wir $L(G') = \{a\}$.

Knobelaufgabe H1.6. (*Thue-Morse-Sequenz*) siehe unten

Die Thue-Morse-Sequenz über dem Alphabet $\Sigma \coloneqq \{0, 1\}$ kann wie folgt definiert werden. Wir setzen $w_0 \coloneqq 0$, und für $n \in \mathbb{N}$ definieren wir $w_{n+1} \coloneqq w_n\overline{w_n}$, wobei $\overline{w}$ das Einerkomplement des Binärwortes $w \in \Sigma^*$ bezeichnet, z.B. $\overline{100111} = 011000$. Die ersten Folgenglieder sind also $w_0 = 0$, $w_1 = 01$, $w_2 = 0110$, und $w_3 = 01101001$.





Wir betrachten nun, wie sich die untere linke Ecke eines Zauberwürfels verhält, wenn man die Sequenz ausführt, indem man bei einer 0 die Vorderseite im Uhrzeigersinn dreht, und bei einer 1 die rechte Seite gegen den Uhrzeigersinn. Dies ist links abgebildet. Rechts sehen Sie einen entsprechenden DFA.

An welcher Stelle befindet sich a nach der Ausführung von w_{2024}?

Hinweis: Sie können einen Computer verwenden. Nach w_{100} ist die Ecke in d.

Abgabe: Lösungen für die Knobelaufgaben bitte auf Moodle separat abgeben. Sie werden höchstwahrscheinlich alle korrigiert werden. Gruppenabgabe ist nicht notwendig.

Lösungsskizze. Sei $M = (Q, \Sigma, \delta, q_0, F)$ obiger DFA. Gesucht ist $\delta(q_0, w_{2024})$, also der Zustand des Automaten nach Einlesen von w_{2024}.

Wir betrachten zwei Funktionen $f_i, g_i : Q \to Q$, für $i \in \mathbb{N}$, die definiert sind als $f_i(q) := \delta(w_i, q)$ und $g_i(q) := \delta(\overline{w_i}, q)$. Offensichtlich lassen sich f_0 und g_0 leicht anhand der Transitionsfunktion δ ausrechnen. Es gilt

$$f_{i+1}(q) = \delta(w_{i+1}, q) = \delta(w_i\overline{w_i}, q) = \delta(\overline{w_i}, \delta(w_i, q)) = g_i(f_i(q))$$

Insbesondere können wir also $f_{i+1} = g_i \circ f_i$ folgern, und $g_{i+1} = f_i \circ g_i$ gilt analog. (Hier bezeichnet $\circ$ Verknüpfung von Funktionen.) Nach diesem Muster berechnen wir nun $f_{2024}(q_0) = \delta(q_0, w_{2024})$, etwa wie folgt.

```
f = [1,2,3,0,4,5]
g = [0,1,3,4,5,2]
for it in range(2024):
    f_new = [g[f[i]] for i in range(6)]
    g_new = [f[g[i]] for i in range(6)]
    f, g = f_new, g_new
print(f[0])
```

Das Ergebnis ist d.

Anmerkung: Wir beobachten $f_1 = f_5$ und $g_1 = g_5$. Somit ist $f_{2024} = f_4$.