Prof. Dr. Jasmin Blanchette
Jannis Limperg

Ludwig-Maximilians-Universität München
Institut für Informatik
Abgabe bis 3. Mai 2023, 10:00 Uhr

Lösungsvorschlag zur Übung 2 zur Vorlesung

Formale Sprachen und Komplexität

FSK2-1 ***Kleine Automaten*** (2 Punkte)

In dieser Aufgabe geht es um DFAs mit Eingabealphabet $\Sigma = \{a, b\}$. Zeigen oder widerlegen Sie die folgenden Aussagen. Wenn Sie für Ihren Beweis konkrete Automaten verwenden, geben Sie deren Zustandsgraphen an.

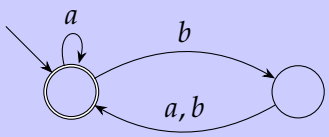
a) Für jeden DFA A mit genau einem Zustand und $aabbaba \in L(A)$ gilt auch $aa \in L(A)$.

LÖSUNGSVORSCHLAG: Wahr. Da es nur einen Zustand gibt, kommt man mit jedem Buchstaben dorthin. Da es ein akzeptiertes Wort gibt, ist der eine Zustand auch ein Endzustand. Damit werden alle Wörter akzeptiert, also auch aa.

b) Sei B ein DFA mit zwei Zuständen sodass gilt: Es gibt ein Wort $z \notin L(B)$ und $\forall i \in \mathbb{N}.\ a^i \in L(B)$. Für jeden solchen Automaten B gilt auch $bb \notin L(B)$.

LÖSUNGSVORSCHLAG:

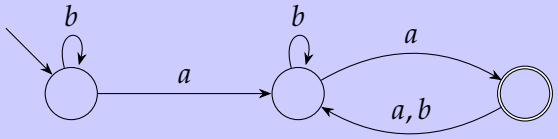
Falsch. Beispielsweise gilt es in folgendem Automaten nicht:



c) Für jeden DFA C mit genau drei Zuständen sowie $aba \in L(C)$ und $aaa, aab, abb \notin L(C)$ gilt auch $aa \notin L(C)$.

LÖSUNGSVORSCHLAG:

Falsch. Beispielsweise gilt es in folgendem Automaten nicht:



d) Für jeden DFA D mit genau zwei Zuständen sowie $\varepsilon \notin L(D)$, $a \in L(D)$, $b \in L(D)$ und $bb \in L(D)$ gilt auch $abbbabbaabbba \in L(D)$.

LÖSUNGSVORSCHLAG:

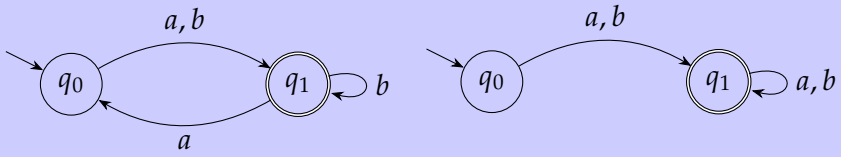
Falsch. Da es ein akzeptiertes Wort gibt, gibt es einen Endzustand. Da es ein nicht akzeptiertes Wort gibt, gibt es einen Zustand, der kein Endzustand ist. Da $\varepsilon \notin L(D)$ ist der Startzustand kein Endzustand (wir nennen ihn q_0), der andere Zustand schon (wir nennen ihn q_1).

Da $a \in L(D)$ kommt man von q_0 mit a zu q_1.

Da $b \in L(D)$ kommt man von q_0 mit b zu q_1.

Da $bb \in L(D)$ kommt man von q_1 mit b zu q_1.

Damit wissen wir nur noch nicht, wohin man von q_1 mit a kommt. Es gibt also zwei mögliche Automaten, die alle Bedingungen erfüllen:

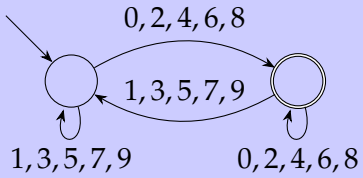


Der linke Automat akzeptiert *abbbabbaabbba* nicht, also ist die Aussage falsch.

FSK2-2 ***Automaten angeben*** (2 Punkte)

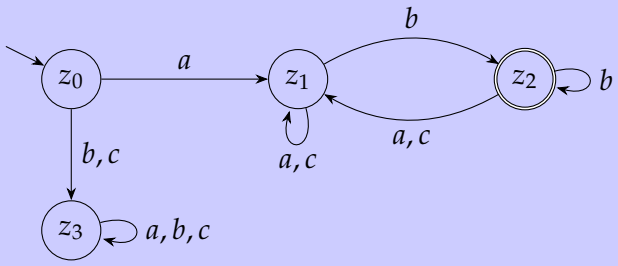
a) Geben Sie den Zustandsgraph eines DFA über dem Alphabet $\{0, \ldots, 9\}$ an, der genau die geraden natürlichen Zahlen (in Dezimalschreibweise) akzeptiert. Nullen am Anfang sind erlaubt, d.h. wir betrachten 002 auch als gerade Zahl.

LÖSUNGSVORSCHLAG:



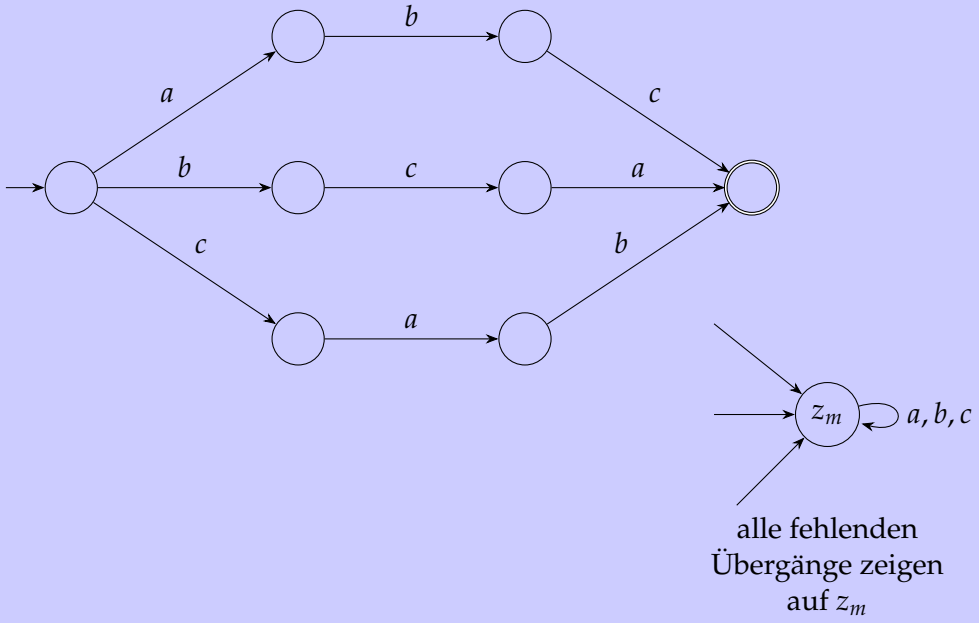
b) Geben Sie den Zustandsgraph eines DFA über dem Alphabet $\{a, b, c\}$ an, der genau die Wörter akzeptiert, die mit a anfangen und mit b enden. Frei nach dem Motto: „Wer a sagt, muss auch b sagen."

LÖSUNGSVORSCHLAG:



c) Geben Sie den Zustandsgraph eines DFA über dem Alphabet $\{a, b, c\}$ an, der genau die Wörter *abc*, *bca* und *cab* akzeptiert.

LÖSUNGSVORSCHLAG:



Hinweis: Ein Zustand mit der Rolle von z_m nennt man auch Mülleimerzustand oder Sink State.

FSK2-3 ***Reguläre Palindrome*** (0 Punkte)

Für alle Teilaufgaben in dieser Aufgabe ist das Alphabet $\Sigma = \{a, b\}$.

Hinweis: Die Sprache aller Palindrome ist nicht regulär, kann also nicht durch einen DFA beschrieben werden. (Wie man das zeigt, lernen Sie später in der Vorlesung). Sie sollten also in den Teilaufgaben b) und c) nicht versuchen, einen Automaten für die Sprache der Palindrome anzugeben.

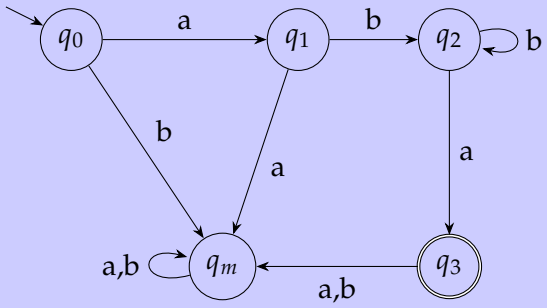
a) Zeigen Sie: Jede Sprache, die nur endlich viele Wörter enthält, ist regulär.

LÖSUNGSVORSCHLAG: Sei L eine Sprache über Σ, die nur endlich viele Wörter enthält. Wir konstruieren die Grammatik $G = (\{S\}, \Sigma, P, S)$. Als Produktionen P wählen wir für jedes Wort $w \in L$ mit $w = a_1 \dots a_n$ die Produktionen $S \to a_1 A_2$, $A_2 \to a_2 A_3$, $\dots$, $A_{n-1} \to a_{n-1} A_n$, $A_n \to a_n$, wobei die A_i neue Variablen sind. Offensichtlich ist $L(G) = L$. Wir haben außerdem nur endlich viele Produktionen eingeführt, die alle Typ 3-Format haben, somit ist G eine Typ 3-Grammatik.

b) Geben Sie den Zustandsgraph eines DFA A an, der nur Wörter akzeptiert, die Palindrome sind und jeweils mindestens ein a und ein b enthalten. Zudem soll A mindestens 8 verschiedene Wörter akzeptieren, also $|L(A)| \geq 8$.

LÖSUNGSVORSCHLAG:

Viele Lösungen sind möglich. Der folgende Automat erkennt beispielsweise die Sprache abb^*a.



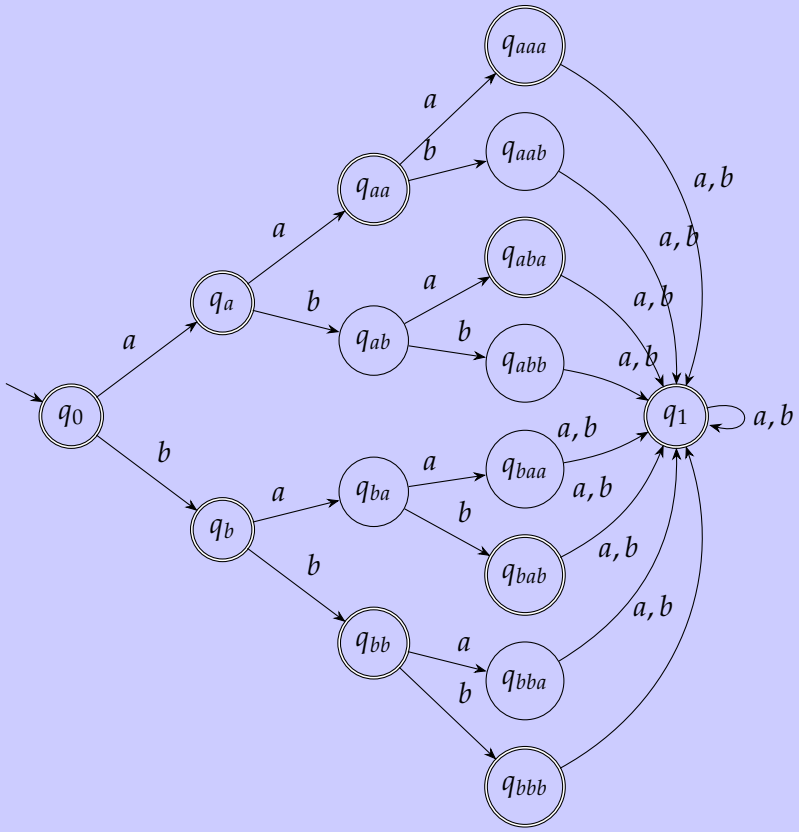
c) Geben Sie den Zustandsgraph eines DFA B an, der (1) alle Palindrome akzeptiert und (2) nur Nicht-Palindrome akzeptiert, die mindestens 4 Buchstaben lang sind.

LÖSUNGSVORSCHLAG:

Wir müssen sicherstellen, dass für $L(B)$ gilt:

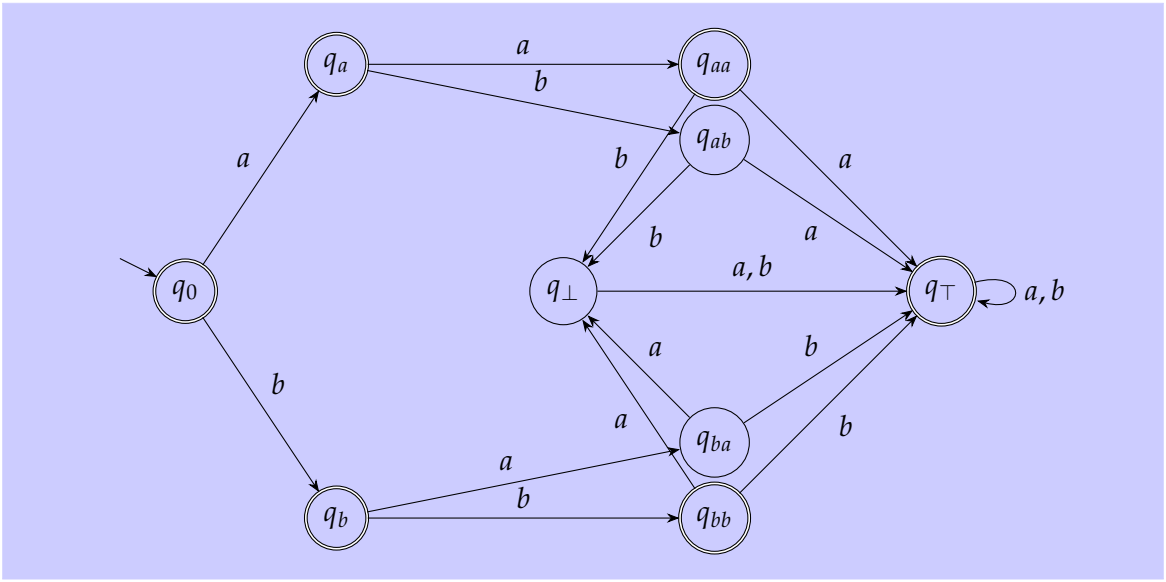
$$
\begin{aligned}
&\{w \in \{a,b\}^* \mid w \text{ ist Palindrom}\} \\
&\subseteq L(B) \\
&\subseteq \{w \in \{a,b\}^* \mid |w| < 4 \text{ und } w \text{ ist Palindrom}\} \cup \{w \in \{a,b\}^* \mid |w| \geq 4\}
\end{aligned}
$$

Aus dem linken $\subseteq$ kann kein $=$ werden, da diese Sprache nicht regulär ist, also machen wir aus dem rechten $\subseteq$ ein $=$. Wir konstruieren also B als DFA, der die Sprache $\{w \in \{a,b\}^* \mid |w| < 4 \text{ und } w \text{ ist Palindrom}\} \cup \{w \in \{a,b\}^* \mid |w| \geq 4\}$ erkennt.



(Dieser Automat ist nicht minimal, aber dafür noch halbwegs übersichtlich.)

Es gibt aber auch andere Sprachen, die unsere Spezifikation erfüllen. Hier ist eine mit einem deutlich kleineren Automaten:



FSK2-4 ***Grammatik-Konkatenation*** (0 Punkte)

Seien G und G' Typ i-Grammatiken (für $i \in \{0, \ldots, 3\}$) sodass $\varepsilon \notin L(G)$ und $\varepsilon \notin L(G')$.

Zeigen oder widerlegen Sie für alle i: Es gibt eine Grammatik G'' vom Typ i, sodass $L(G'') = L(G)L(G')$.

LÖSUNGSVORSCHLAG:

Ja, dies gilt.

Vorbereitungen: Sei $G = (V, \Sigma, P, S)$ und $G' = (V', \Sigma', P', S')$.
Um zu erreichen, dass V disjunkt zu V' und Σ' ist, führe für jedes $A \in V \cap (V' \cup \Sigma')$ eine neue Variable A' ein und ersetze A in G durch A' (sowohl in V als auch in P und eventuell in S). Führe analog für jedes $A \in V' \cap \Sigma$ eine neue Variable A' ein und ersetze A in G' durch A'.
Falls Σ und Σ' nicht disjunkt sind, führe für jedes $a \in \Sigma \cap \Sigma'$, welches auf der linken Seite einer Produktionsregel von P vorkommt, eine Variable V_a ein, ersetze jedes a in P durch V_a und füge die Produktionsregel $V_a \to a$ zu P hinzu. Führe analog für jedes $a \in \Sigma \cap \Sigma'$, welches auf der linken Seite einer Produktionsregel von P' vorkommt, eine Variable V'_a ein. Dies ändert Typ 2- und Typ 3-Grammatiken nicht, da dort keine Terminale auf linken Seiten vorkommen.
Beide Vorbereitungen ändern die erzeugten Sprachen $L(G)$ und $L(G')$ nicht.

Konstruktion von $G'' = (V'', \Sigma'', P'', S'')$:

- Für Typ 0, 1, 2:

- Sei S'' eine neue Variable (d.h. $S'' \notin V \cup V'$)
- $V'' = V \cup V' \cup \{S''\}$
- $\Sigma'' = \Sigma \cup \Sigma'$
- $P'' = P \cup P' \cup \{S'' \to SS'\}$

- Für Typ 3 verwende entweder die Abschlusseigenschaften regulärer Sprachen oder:
  - $S'' = S$
  - $V'' = V \cup V'$
  - $\Sigma'' = \Sigma \cup \Sigma'$
  - $P'' = P_n \cup P_c \cup P'$
    wobei $P_n = \{A \to aB \mid (A \to aB) \in P\}$
    und $P_c = \{A \to aS' \mid (A \to a) \in P\}$.