Universität Siegen
Lehrstuhl Theoretische Informatik
Markus Lohrey

Übungsblatt 2

Aufgabe 1. Sei Σ ein beliebiges Alphabet. Geben Sie eine Grammatik an, die die Sprache

$$L = \{w \in \Sigma^* \mid w = w^r\}$$

erzeugt. Dabei ist w^r das Wort w rückwärts gelesen, z.B. für $w = aabb$ ist $w^r = bbaa$. Die Sprache L ist damit die Menge aller Palindrome über dem Alphabet Σ.

Lösung zu Aufgabe 1. Wir geben zunächst ein einfaches Beispiel. Sei $\Sigma = \{a, b\}$ das Alphabet. Wir suchen eine Grammatik G mit $L = L(G)$. Eine Möglichkeit ist $G = (V, \Sigma, P, S)$ mit

- $V = \{S\}$
- $P = \{S \to aSa \mid bSb \mid a \mid b \mid \varepsilon\}$

Erklärung

Formal müsste man wieder beide Mengeninklusionen $\subseteq$ und $\supseteq$ zeigen.

Mit den Produktionen $S \to aSa$ und $S \to bSb$ verlängert man das Wort am Anfang und am Ende beliebig oft jeweils um den gleichen Buchstaben, was sicherstellt, dass Palindrome erzeugt werden. Bei Wörtern gerader Länge wird die Ableitung durch Anwendung der Produktion $S \to \varepsilon$ beendet. Bei Wörtern ungerader Länge wird am Ende durch Anwendung von $S \to a$ oder $S \to b$ noch ein Symbol in der Mitte hinzugefügt.

Anmerkung: Gesucht ist eine beliebige Grammatik, daher ist die ε-Sonderregel nicht von Bedeutung. Man könnte die Grammatik unter Verwendung der Sonderregel allerdings so umformen, dass sie kontextfrei ist.

Für ein beliebiges Alphabet Σ wählt man die Produktionen zum Beispiel wie folgt:

$$P = \{S \to \varepsilon\} \cup \bigcup_{x \in \Sigma} \{S \to xSx,\ S \to x\}$$

Die Begründung ist hierbei genau wie im Fall $\Sigma = \{a, b\}$.

Aufgabe 2. Geben Sie zu jeder der folgenden Sprachen eine Grammatik und einen endlichen Automaten an.

(a) $L_1 = \{w \in \{a, b\}^* \mid \text{Das Wort } w \text{ enthält mindestens ein } b.\}$

(b) $L_2 = \{w \in \{a, b\}^* \mid \text{Die Anzahl der } a\text{'s ist durch 3 teilbar.}\}$

(c) $L_3 = \{w \in \{a, b\}^+ \mid \text{Der erste und letzte Buchstabe in } w \text{ stimmen überein.}\}$

(d) $L_4 = \left\{a^n b^m c^\ell \mid n \geq 0,\ m \geq 1,\ \ell \geq 2\right\}$

(e) $L_5 = \{w \in \{a, b\}^* \mid |w| \leq 3\}$

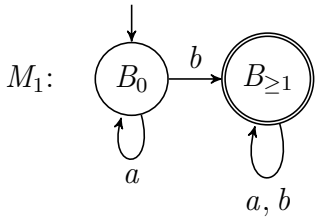
Lösung zu Aufgabe 2. Im Folgenden seien G_i die Grammatiken mit $L(G_i) = L_i$ und M_i die Automaten mit $T(M_i) = L_i$.

(a) $G_1 = (V, \Sigma, P, B_0)$ wobei

- $V = \{B_0, B_{\geq 1}\}$
- $\Sigma = \{a, b\}$
- $P = \{B_0 \to aB_0 | bB_{\geq 1},\ B_{\geq 1} \to \varepsilon | aB_{\geq 1} | bB_{\geq 1}\}$

Alternativ, als reguläre Grammatik (ε eliminiert):

- $P = \{B_0 \to b | aB_0 | bB_{\geq 1},\ B_{\geq 1} \to a | b | aB_{\geq 1} | bB_{\geq 1}\}$



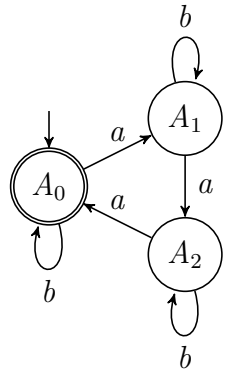
Erklärung

Wir verwenden zwei Zustände in diesem DFA um zu unterscheiden, ob bisher ausschließlich a's gelesen wurden (Zustand B_0) oder mindestens ein b gelesen wurde (Zustand $B_{\geq 1}$). Die Idee für die Grammatik ist identisch.

(b) $G_2 = (V, \Sigma, P, A_0)$ wobei

- $V = \{A_0, A_1, A_2\}$
- $\Sigma = \{a, b\}$
- $P = \{A_0 \to \varepsilon | aA_1 | bA_0,\ A_1 \to aA_2 | bA_1,\ A_2 \to aA_0 | bA_2\}$

M_2:



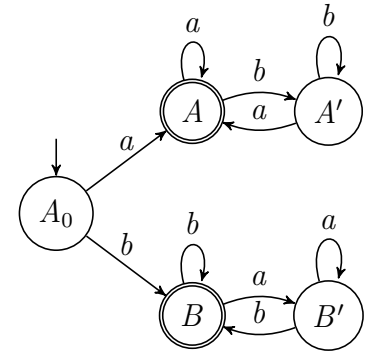
Erklärung

Wir verwenden drei Zustände in diesem DFA, um die Anzahl der a's modulo 3 zu zählen. Der Zustand A_0 sagt aus, dass die gelesene Anzahl der a's durch 3 teilbar ist. Zustand A_1 entspricht einem Rest von 1 und Zustand A_2 entspricht einem Rest von 2 bei Teilung durch 3. Die Grammatik ergibt sich wieder direkt aus dem Automaten.

(c) $G_3 = (V, \Sigma, P, S)$ wobei

- $V = \{S, A, B\}$
- $\Sigma = \{a, b\}$
- $P = \{S \to a|b|aA|bB,\ A \to a|aA|bA,\ B \to b|aB|bB\}$

M_3:

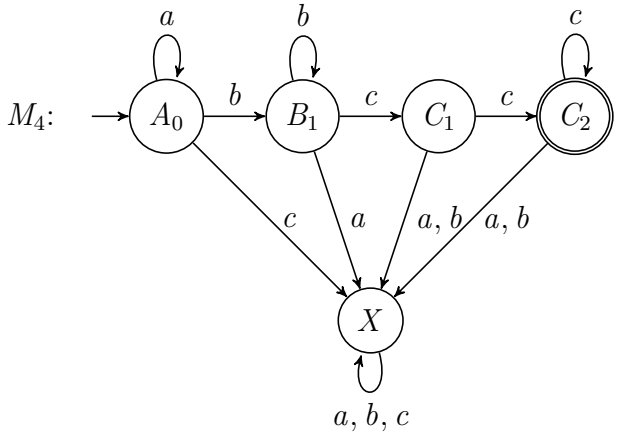


Erklärung

Die Wörter a und b beginnen und enden mit dem selben Buchstaben (daher die Regeln $S \to a$ und $S \to b$). Für längere Wörter merkt die Grammatik sich mit Hilfe von dem Nichtterminal A, dass der erste Buchstabe ein a ist und das Wort somit auch mit a enden muss. Analog werden mit Hilfe von B Wörter erzeugt, die mindestens Länge 2 haben und mit b beginnen und enden.
Der DFA arbeitet nach dem gleichen Prinzip, d.h. vom Startzustand aus erreicht man mit a einen Zustand A, so dass das Wort auch mit a enden muss und mit b erreicht man vom Startzustand einen Zustand B, so dass hier das Wort auch mit b enden muss.

(d) $G_4 = (V, \Sigma, P, A)$ wobei

- $V = \{A, B, C\}$
- $\Sigma = \{a, b\}$
- $P = \{A \to aA|bB,\ B \to bB|cC,\ C \to c|cC\}$



Erklärung

Der DFA und die Grammatik lesen bzw. erzeugen zuerst beliebig viele a's (Zustand A_0 bzw. Nichtterminal A), wobei es auch möglich ist kein a zu erzeugen und direkt in den anschließenden b-Block überzugehen. Anschließend werden beliebig viele b's gelesen bzw. erzeugt (Zustand B_1 und Nichtterminal B) und dabei wird sichergestellt, dass mindestens ein b wirklich gelesen bzw. erzeugt wird. Abschließend werden beliebig viele

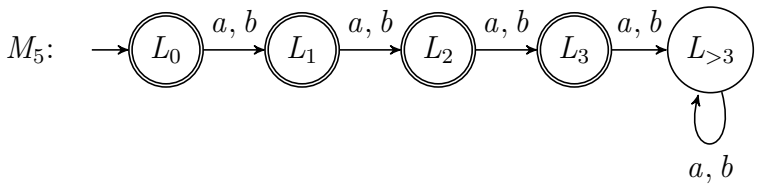
c's gelesen bzw. erzeugt (Zustände C_1, C_2 und Nichtterminal C) und dabei wird sichergestellt, dass die Anzahl der gelesenen c's mindestens zwei ist. Der Fangzustand X im DFA wird erreicht, falls ein Wort gelesen wird, welches nicht aus a's gefolgt von b's gefolgt von c's besteht.

(e) $G_5 = (V, \Sigma, P, S)$ wobei

- $V = \{S\}$
- $\Sigma = \{a, b\}$
- $P = \{S \to \varepsilon|a|b|aa|ab|ba|bb|aaa|aab|aba|abb|baa|bab|bba|bbb\}$

Alternativ, als reguläre Grammatik:

- $V = \{S, L_1, L_2\}$
- $P = \{S \to \varepsilon|a|b|aL_1|bL_1, L_1 \to a|b|aL_2|bL_2, L_2 \to a|b\}$



Erklärung

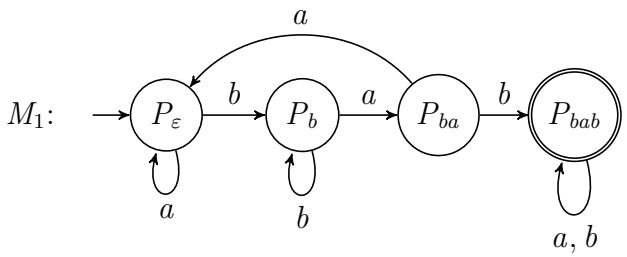
Für den DFA gilt: Der Startzustand L_0 ist gleichzeitig ein Endzustand, sodass das leere Wort akzeptiert wird, während L_1 mit Wörtern der Länge 1, L_2 mit Wörtern der Länge 2 und L_3 mit Wörtern der Länge 3 erreicht wird. Alle Wörter der Länge mindestens 4 landen im Fangzustand $L_{>3}$. Für die Grammatik können wir für jedes Wort w der Länge $|w| \leq 3$ eine Produktion $S \to w$ angeben, allerdings ist diese Grammatik nicht regulär. Für eine reguläre Grammatik muss man für verschiedene Wortlängen andere Nichtterminale verwenden.

Aufgabe 3. Geben Sie zu jeder der folgenden Sprachen einen deterministischen, endlichen Automaten an. Finden Sie einen nichtdeterministischen, endlichen Automaten, der weniger Zustände benötigt?

(a) $L_1 = \{w \in \{a, b\}^* \mid w \text{ enthält das Wort } bab.\}$

(b) $L_2 = \{w \in \{a, b, c\}^* \mid w \text{ enthält höchstens zwei verschiedene Buchstaben.}\}$

Lösung zu Aufgabe 3. (a)

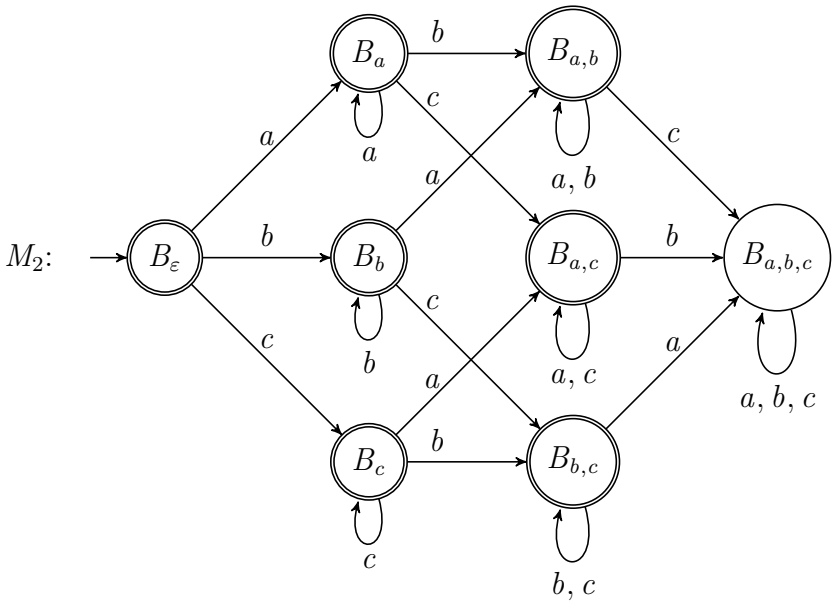


Erklärung

Die Zustände P_ε, P_b, und P_{ba} repräsentieren den maximalen Präfix von bab (ε, b, ba), mit dem das gelesene Wort zur Zeit endet falls bab bislang noch nicht vollständig gelesen wurde. Zum Beispiel landet man mit dem Wort $aaba$ im Zustand $\delta(P_\varepsilon, aaba) = P_{ba}$, weil das Wort mit ba endet. Ein anderes Beispiel wäre baa, welches in den Zustand $\delta(P_\varepsilon, abaa) = P_\varepsilon$ führt, da $abaa$ nur auf den leeren Präfix ε von bab endet. Anders ausgedrückt, wird sich mit den Zuständen P_ε, P_b, und P_{ba} gemerkt, welche Buchstaben als nächstes kommen müssen, damit das Teilwort bab vorkommt. Der Zustand P_{bab} wird erreicht, wenn bab vollständig gelesen wurde und anschließend bleibt man für beliebige Buchstaben in diesem Zustand (weil das Wort bab bereits im Wort vorkommt).

Es gibt keinen NFA mit weniger Zuständen, da auch in einem NFA (unter anderem) die Wörter ε ,b, ba und bab in unterschiedliche Zustände führen müssen, die vom Startzustand aus erreichbar sind und jeweils in einen Endzustand führen. Gäbe es einen NFA mit höchstens 3 Zustände, so müssten mindestens zwei dieser Wörter in den gleichen Zustand führen, welcher vom Startzustand aus erreichbar ist und der auch in einen Endzustand führt (Schubfachprinzip). Dadurch könnte man ein Wort konstruieren, welches bab enthält und nicht akzeptiert wird oder bab nicht enthält und trotzdem akzeptiert wird.

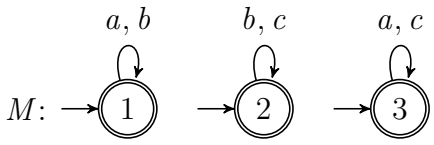
(b)



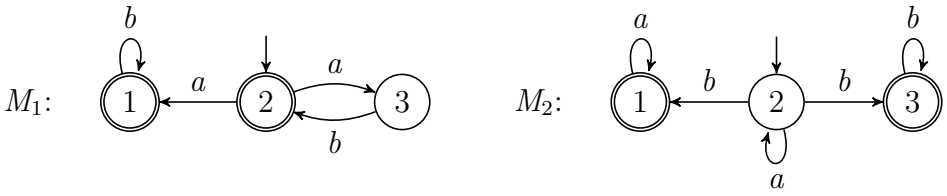
Erklärung

Wir verwenden die Zustände des Automaten um zu „speichern“, welche der Symbole a, b, c bereits im Wort enthalten sind. Zum Beispiel enthalten alle Wörter, die in den Zustand $B_{a,c}$ führen, beliebig oft die Buchstaben a und c aber nicht den Buchstaben b.

Ein deutlich kleinerer NFA M mit nur 3 Zuständen (alle sind Start- und Endzustände) sieht wie folgt aus:



Aufgabe 4. Gegeben seien die folgenden NFAs:



(a) Geben Sie die Sprachen $T(M_1)$ und $T(M_2)$ an.

(b) Geben Sie DFAs für $T(M_1)$ und $T(M_2)$ an.

Lösung zu Aufgabe 4.

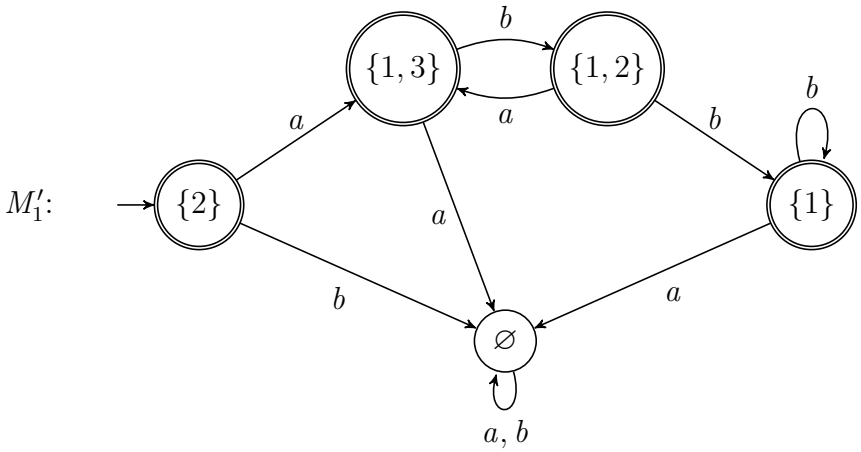
(a)

- $T(M_1) = \{(ab)^n \mid n \in \mathbb{N}\} \cup \{(ab)^n ab^m \mid n, m \in \mathbb{N}\}$
  Begründung: Vom Startzustand 2 erreicht man den Endzustand 2 ausschließlich in dem man (beliebig oft) mit a nach 3 geht und anschließend mit b wieder zurück nach 2 geht. Daher werden Wörter der Form $(ab)^n$ für beliebige $n \in \mathbb{N}$ akzeptiert. Zusätzlich kann man aber anschließend (oder sofort) vom Zustand 2 auch noch mit a in den Endzustand 1 wechseln und dort beliebig viele b's lesen. So lassen sich Wörter der Form $(ab)^n ab^m$ für $n, m \in \mathbb{N}$ erzeugen.
- $T(M_2) = \{a^n ba^m \mid n, m \in \mathbb{N}\} \cup \{a^n bb^m \mid n, m \in \mathbb{N}\}$
  Begründung: Vom Startzustand 2 erreicht man den Endzustand 1 in dem man beliebig oft mit a im Zustand 2 bleibt, anschließend mit b in den Zustand 1 wechselt und dort beliebig viele a's liest. Somit werden Wörter der Form $a^n ba^m$ für $n, m \in \mathbb{N}$ akzeptiert. Alternativ erreicht man vom Startzustand 2 den Endzustand 3 in dem man beliebig oft mit a im Zustand 2 bleibt, anschließend mit b in den Zustand 3 wechselt und dort beliebig viele b's liest. Somit werden Wörter der Form $a^n bb^m$ für $n, m \in \mathbb{N}$ akzeptiert.

(b) DFAs M_1' und M_2' für die NFAs M_1 und M_2 können wir mittels **Potenzmengenkonstruktion** konstruieren.

Hierbei betrachten wir nur die Zustände (Zustandsmengen), die vom Startzustand aus erreichbar sind.

DFA für $T(M_1)$:



DFA für $T(M_2)$: