

Lösungsvorschlag zur Übung 7 zur Vorlesung
Theoretische Informatik für Studierende der Medieninformatik

TIMI7-1 Sprachen einordnen

Die formalen Sprachen $L_i, i \in \{0, \dots, 2\}$, seien definiert als

$$\begin{aligned} L_0 &:= \{(ab)^i \$ (ba)^i \mid i \in \mathbb{N}\} && \subseteq \{a, b, \$\}^* \\ L_1 &:= \{ab^i \mid i \in \mathbb{N}\} \cup \{c^i a \mid i \in \mathbb{N}\} && \subseteq \{a, b, c\}^* \\ L_2 &:= \{(ab)^j \$ c^i \mid i, j \in \mathbb{N}\} && \subseteq \{a, b, c, \$\}^* \end{aligned}$$

Für die i -fache Wiederholung des Worts w schreiben wir manchmal $(w)^i$ statt nur w^i , um Anfang und Ende von w zu markieren. Die Klammern sind daher *nicht* Teil des Alphabets der jeweiligen Sprachen.

Bearbeiten Sie die folgenden Arbeitsaufträge für jede der Sprachen L_i .

- Beweisen oder widerlegen Sie, dass L_i regulär ist.
- Falls L_i kontextfrei ist, beweisen Sie dies. Falls nicht, begründen Sie, warum das so ist.

Hinweis: Nutzen Sie, dass manche Aussagen direkt aus anderen Aussagen folgen. Um zu beweisen, dass L_i regulär/kontextfrei ist, genügt es, ein geeignetes Konstrukt K_i (Grammatik, Automat oder regulärer Ausdruck) anzugeben und kurz zu begründen, warum $L(K_i) = L_i$ gilt.

LÖSUNGSVORSCHLAG:

- $L_0 = \{(ab)^i \$ (ba)^i \mid i \in \mathbb{N}\}$ ist nicht regulär, aber kontextfrei.

Beobachtung: Offensichtlich haben alle Wörter in L_0 die Form $w_1 \$ w_2$, mit $w_1 = \overline{w_2}$. Der Teil vor dem $\$$ und der Teil nach dem $\$$ sind also gleich lang.

Beweis "nicht regulär": Sei $n \in \mathbb{N}$. Sei $z = (ab)^n \$ (ba)^n$. Dann gilt für alle Zerlegungen $uvw = z$ mit $|uv| \leq n$ und $|v| > 0$: liegt vollständig im vorderen Wortteil, vor $\$$. Betrachten wir jetzt uv^0w , so fällt auf, dass der Teil vor dem $\$$ jetzt um $|v|$ kürzer ist, als der Teil nach $\$$. Somit kann aber uv^0w nicht in L_0 sein, und damit erfüllt L_0 nicht das Pumpinglemma für reguläre Sprachen.

- $L_1 = \{ab^i \mid i \in \mathbb{N}\} \cup \{c^i a \mid i \in \mathbb{N}\} = L(ab^*) \cup L(c^*a)$ ist regulär (und damit auch kontextfrei), da die Sprache von einer Vereinigung regulärer Ausdrücke erzeugt wird. Jedes Wort der Sprache ist entweder aus ab^* oder aus c^*a , da beide i 's unabhängig voneinander sind. Wir nutzen hierbei, dass reguläre Sprachen unter Vereinigung abgeschlossen sind.
- $L_2 = \{(ab)^j \$ c^i \mid i, j \in \mathbb{N}\} = L((ab)^* \$ c^*)$ ist regulär (und damit auch kontextfrei), da die Sprache von einem regulären Ausdruck erzeugt wird. Jedes Wort der Sprache besteht erst aus beliebig vielen ab 's gefolgt von $\$$ und endet mit beliebig vielen c .

TIMI7-2 Turingmaschinen erstellen

Wir betrachten die Sprache $L = \{w \mid \#_b(w) > \#_c(w)\}$ über dem Alphabet $\{b, c\}$.

- a) Erstellen Sie auf <https://turingmachinesimulator.com/> eine TM, die L erkennt. Geben Sie sowohl einen Link zur Maschine an als auch den „Programmtext“ der Maschine.

LÖSUNGSVORSCHLAG:

TM: <http://turingmachinesimulator.com/shared/ndqrvoyaon>

Idee: Wir gehen von links nach rechts über das Wort. Für jedes b , das wir dabei finden, suchen wir nach rechts ein entsprechendes c , und für jedes c suchen wir ein b . Die von links abgearbeiteten b 's und c 's ersetzen wir durch X und markieren so den Teil des Wortes, den wir nicht mehr besuchen müssen. Die entsprechenden c 's und b 's ersetzen wir durch C 's und B 's, um zu markieren, dass sie bereits verwendet wurden. Wenn wir nach einem „Partner“ zu einem b suchen, aber keinen finden, also $\square$ lesen, akzeptieren wir das Wort, da es somit mindestens ein mehr b 's als c 's geben muss.

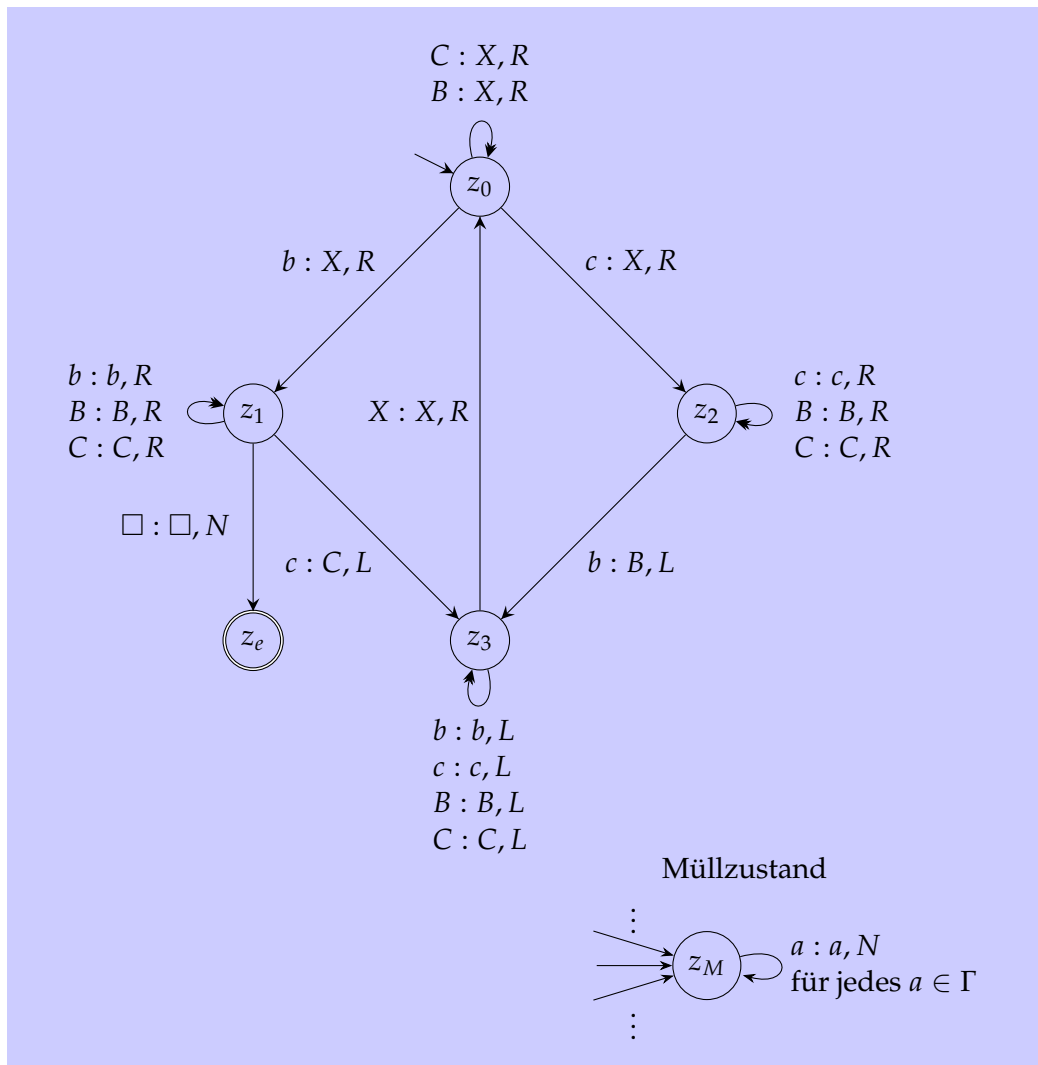
Genauer führen wir folgende Schritte in einer Schleife aus:

- i) (z_0 , Startzustand) Laufe nach rechts über das Wort und halte nach dem ersten b oder c an. Das gefundene b oder c wird dabei durch X ersetzt. Ist das erste Zeichen bereits ein b oder c , halte auf dem zweiten Zeichen.
- ii) (z_1) Ist das gefundene Zeichen ein b , laufe nach rechts und ersetze das erste c durch ein C .
- iii) (z_2) Ist das gefundene Zeichen ein c , suche entsprechend nach dem ersten b und ersetze es durch ein B .
- iv) (z_3) Laufe nach links über das Wort und halte rechts des ersten X .

Wir akzeptieren, wenn wir in z_1 $\square$ lesen, es also kein c zum bereits gelesenen b gibt. Als Optimierung ersetzen wir im ersten Schritt B und C durch X und verkürzen so den Teil des Wortes, den wir in den nächsten Durchläufen betrachten müssen. Das ist aber nicht nötig.

- b) Geben Sie für Ihre TM aus Teilaufgabe a) einen Zustandsgraphen an.

LÖSUNGSVORSCHLAG:



- c) Geben Sie die Läufe der folgenden Wörter auf Ihre TM aus Teilaufgabe a) an: $\varepsilon, c, bcc, cbcb$.

Hinweis: Wörter, die nicht in L liegen, erzeugen eventuell unendliche Läufe. Geben Sie in solchen Fällen ein Präfix an, aus dem ersichtlich wird, dass der Lauf unendlich ist.

LÖSUNGSVORSCHLAG:

Für ε :

$$z_0 \square \vdash z_M \square \vdash z_M \square \vdash \dots$$

Für c :

$$z_0 c \vdash X z_2 \square \vdash X z_M \square \vdash X z_M \square \vdash \dots$$

Für bcc :

$$\begin{aligned} z_0bcc \vdash Xz_1cc \vdash z_3XCc \vdash Xz_0Cc \vdash XXz_0c \vdash XXXz_2\Box \vdash XXXz_M\Box \\ \vdash XXXz_M\Box \vdash \dots \end{aligned}$$

Für $cbcb$:

$$\begin{aligned} z_0cbcb \vdash Xz_2cbcb \vdash z_3XBcb \vdash Xz_0Bcb \vdash XXz_0cb \vdash XXXz_2bb \\ \vdash XXz_3XBb \vdash XXXz_0Bb \vdash XXXXz_0b \vdash XXXXXz_1\Box \\ \vdash XXXXXz_e\Box \end{aligned}$$