Lösungen der Aufgaben
Theoretische Informatik für Dummies
Teil I - Endliche Automaten

Prof. Dr. R. Schmitz*

September 2019

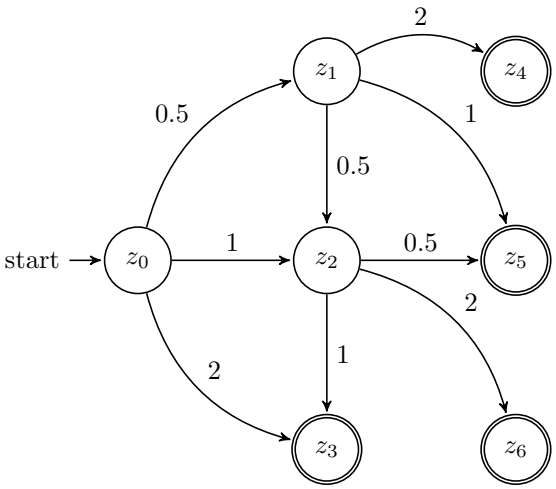
1 Aufgaben zu DFAs

1.1

Es ist nicht für jedes Paar aus Zustand und Eingabesymbol ein Nachfolgezustand erklärt. Deshalb handelt es sich genau genommen um einen NFA (s. Kapitel 2).

1.2

$\Sigma = \{0.5, 1, 2\}$. Der Übersichtlichkeit halber sind die Zustandsübergänge für die vier Endzustände z_3, z_4, z_5, z_6 weggelassen. Sie verhalten sich analog zum Startzustand.



1.3

a) Alle Strings, die höchstens eine Null enthalten: $L(M) = \{w \in \Sigma^* | N_0(w) \leq 1\}$.

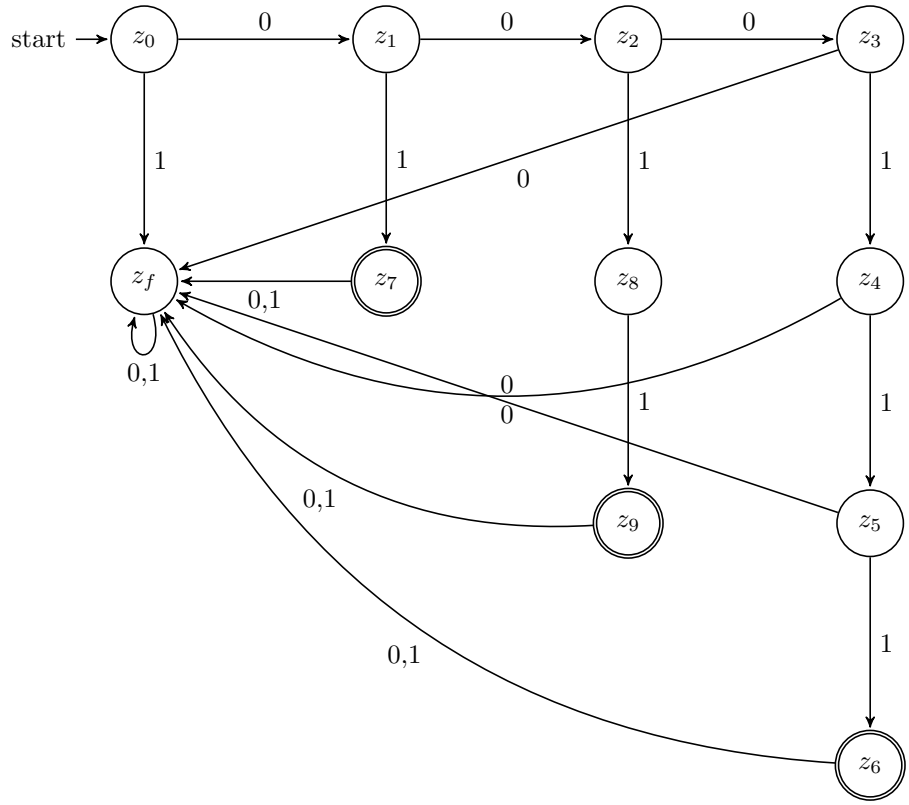
b) $L(M) = \{w \in \Sigma^* | N_0(w) \equiv 2 \bmod 3, N_1(w) \equiv 1 \bmod 3\}$.

c) $L(M) = \{w \in \Sigma^* | w = 1^n 001^m \text{ mit } n, m \geq 0\} \cup \{\epsilon\}$.

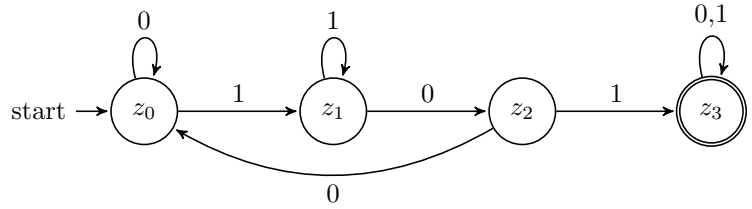
*Studiengang Medieninformatik, Hochschule der Medien Stuttgart

1.4

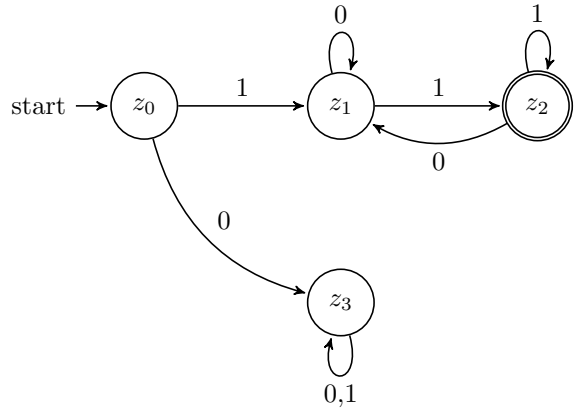
a) $L(M) = \{01, 0011, 000111\}$



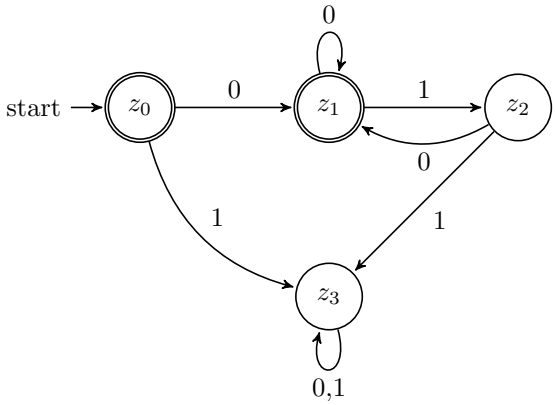
b) $L(M) = \{w \in \Sigma^* | w \text{ enthält 101 als Substring.}\}$



c) $L(M) = \{w \in \Sigma^* | |w| \geq 2 \text{ und das Wort beginnt und endet mit 1.}\}$



d) $L(M) = \{w \in \Sigma^* | \text{Jeder 1 folgt eine 0 und geht eine 0 voran.}\}$



1.5

Ein **Monoid** ist eine nichtleere Menge M zusammen mit einer Operation $\star$, die assoziativ auf M ist und ein neutrales Element $\in M$ besitzt. Konkateniert man drei Wörter $w_1, w_2, w_3 \in \Sigma^*$, so erhält man immer das gleiche Wort, egal wie man klammert:

$$(w_1 w_2) w_3 = w_1 (w_2 w_3) = w_1 w_2 w_3$$

Die Konkatenation ist also assoziativ. Das leere Wort $\epsilon \in \Sigma^*$ ist das neutrale Element wegen $\epsilon w = w\epsilon = w$.

1.6

Nach Definition der erweiterten Übergangsfunktion gilt für Symbole $a \in \Sigma$:

$$\hat{\delta}(z, a) = \hat{\delta}(z, \epsilon a) = \delta(\hat{\delta}(z, \epsilon), a) = \delta(z, a)$$

1.7

Ein solches Wort muss mindestens einen Zustand mehrfach besuchen. Also enthält es einen Schleifenbestandteil, der beliebig oft durchlaufen werden kann, ohne die Sprache zu verlassen. Alle Worte, die so entstehen, sind Element von $L(M)$.

1.8

Angenommen, es gibt eine Zahl N wie im Pumping Lemma. Wir betrachten das Wort $x = 0^{N+1}1^N \in L$. Offenbar ist $|x| = 2N + 1 > N$. Nach dem Pumping Lemma muss gelten $x = uvw$ mit $|uv| \leq N$. Also besteht uv aus lauter Nullen und v aus mindestens einer 0. Bestenfalls ist $v = 1$ und es gilt $uw = 0^N 1^N \notin L$. Widerspruch!

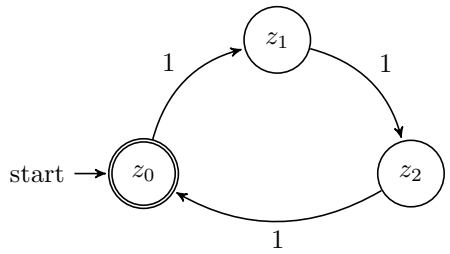
1.9

Äquivalenzklassen von R_L sind:

1. $[\epsilon] = \{w \in \Sigma^* | w = 1^{3n}, n \geq 0\}$
2. $[1] = \{w \in \Sigma^* | w = 1^{3n+1}, n \geq 0\}$

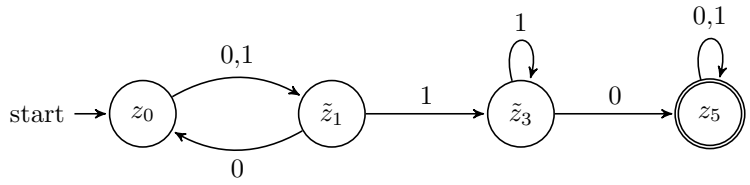
3. $[11] = \{w \in \Sigma^* | w = 1^{3n+2}, n \geq 0\}$

Der dazu gehörige DFA:

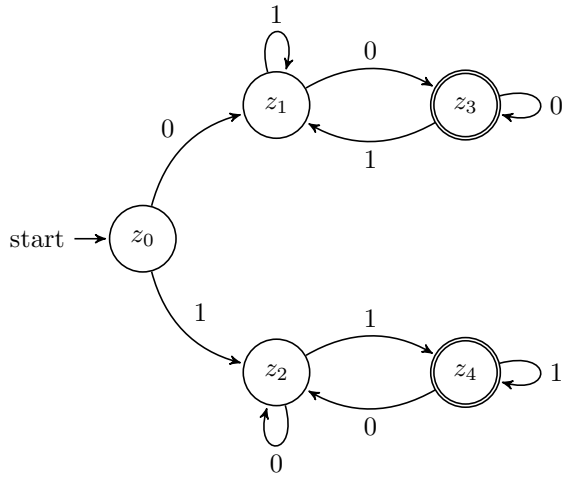


1.10

z_1 und z_2 bzw. z_3 und z_4 sind äquivalent. Deshalb hat der Minimalautomat diese Form:



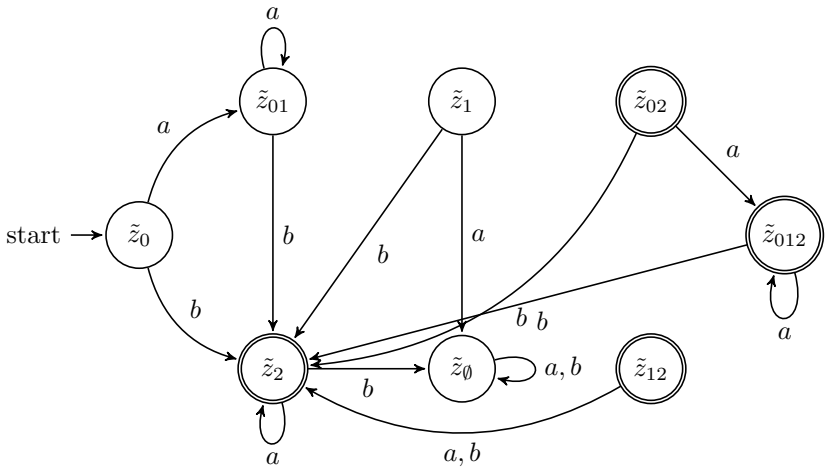
1.11



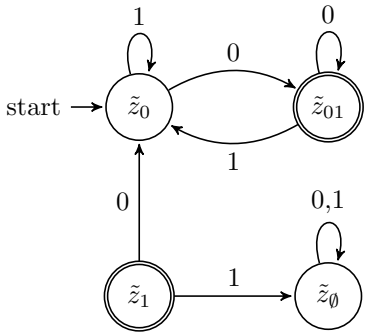
2 Aufgaben zu NFAs

2.1

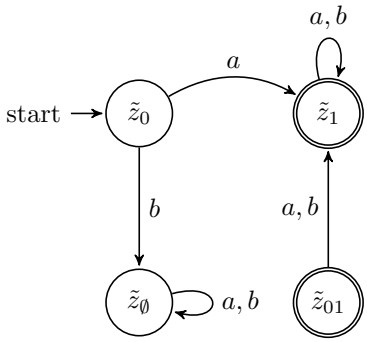
a) $L(M) = \{w \in \Sigma^* | w = a^i b a^j \text{ mit } i, j \geq 0\}$



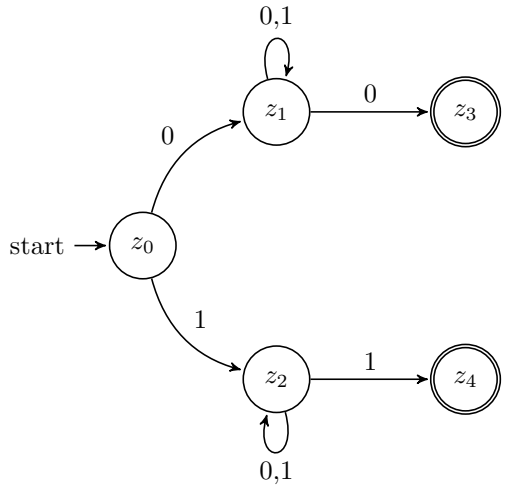
b) $L(M) = \{w \in \Sigma^* | w = u0 \text{ mit } u \text{ beliebig} \in \Sigma^*.\}$



c) $L(M) = \{w \in \Sigma^* | w = au \text{ mit } u \text{ beliebig} \in \Sigma^*.\}$



2.2



2.3

$r = (a^*(ab)(ab)^*b)^*$

3 Aufgaben zu PDAs

3.1

Für beide Sprachen lässt sich die Nicht-Regularität mit dem Pumping Lemma beweisen. Wir betrachten nur die Sprache $PALI(\Sigma)$, der Beweis für die Sprache L verläuft analog. Sei N die Zahl aus dem Pumping Lemma. Wir betrachten das Wort $x = a_1 \dots a_{N-1} a_N a_N a_{N-1} \dots a_1$ mit $a_i \in \Sigma$. Dann ist $x \in \Sigma$ und $|x| = 2N > N$. x sollte also einen aufpumpbaren Bestandteil v besitzen. Wegen $x = uvw$ und $uv \leq N$ muss v komplett in der ersten Worthälfte liegen. Durch das Aufpumpen von v geht dann die Palindrom- Eigenschaft verloren. Widerspruch!

3.2

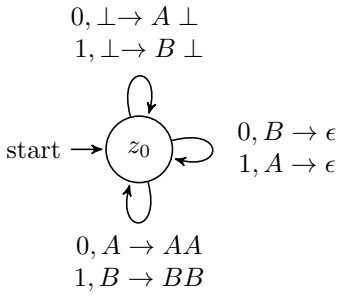
Es handelt sich um einen NPDA ohne ϵ-Übergänge. Mit $\Sigma = \{a, b\}, \Gamma = \{X, \perp\}, Z = \{z_0, z_1\}$ gilt für die Übergangsfunktion $\delta : Z \times \Sigma \times \Gamma \to \mathcal{P}(Z \times \Sigma^*)$ mit

$$\begin{aligned}
\delta(z_0, a, \perp) &= (z_0, X\ \perp) \\
\delta(z_0, b, \perp) &= (z_0, X\ \perp) \\
\delta(z_0, a, X) &= \{(z_0, XX), (z_1, X)\} \\
\delta(z_0, b, X) &= (z_0, XX) \\
\delta(z_1, a, X) &= (z_1, \epsilon) \\
\delta(z_1, b, X) &= (z_1, \epsilon)
\end{aligned}$$

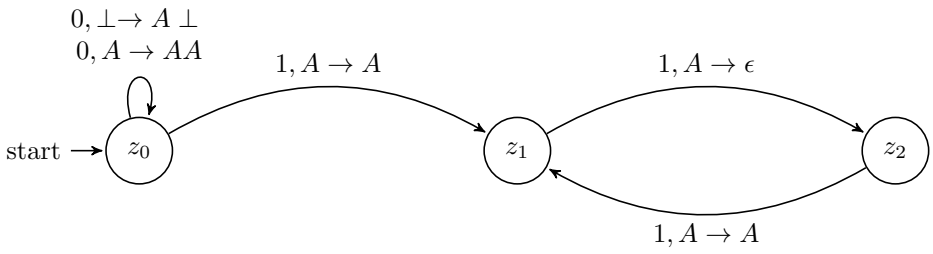
Die akzeptierte Sprache ist $L = \{w \in \Sigma^* | w = uav \text{ mit } |u| = |v|\}$.

3.3

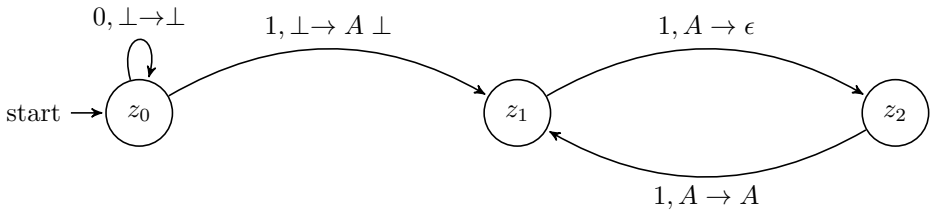
a) $L = \{w \in \Sigma^* | N_0(w) = N_1(w)\}$.



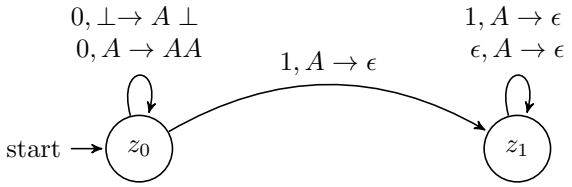
b) $L = \{w \in \Sigma^* | w = 0^n 1^{2n}, n \geq 1\}$.



c) $L = \{w \in \Sigma^* | w = 0^n 1^{2m}, n, m \geq 1\}$.



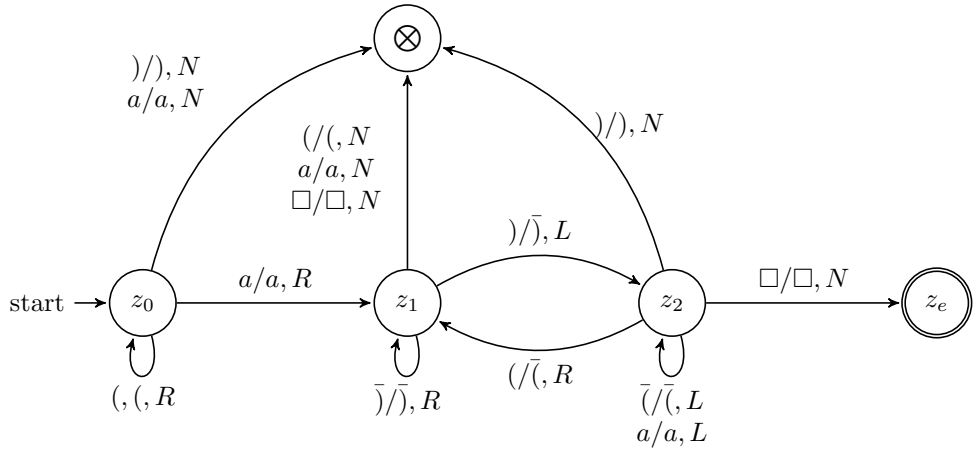
d) $L = \{w \in \Sigma^* | w = 0^n 1^m, n \geq m \geq 1\}$.



4 Aufgaben zu TMs

4.1

$\Sigma = \{(,), a\}, \Gamma = \{(,), a, \bar{(}, \bar{)}, \square\}$.



4.2

$$
\begin{aligned}
\delta(z_0, 0) &= (z_0, 0, R) \\
\delta(z_0, 1) &= (z_1, 1, R) \\
\delta(z_0, \square) &= (\otimes, \square, N) \text{ Inputzahl ist gleich Null.} \\
\delta(z_1, 0) &= (z_1, 0, R) \\
\delta(z_1, 1) &= (z_1, 1, R) \\
\delta(z_1, \square) &= (z_2, \square\, L) \\
\delta(z_2, 0) &= (z_2, 1, L) \\
\delta(z_2, 1) &= (z_3, 0, L) \\
\delta(z_3, 0) &= (z_3, 0, L) \\
\delta(z_3, 1) &= (z_3, 1, L) \\
\delta(z_3, \square) &= (z_e, \square, R)
\end{aligned}
$$

4.3

- Im Startzustand z_0 lese das erste Zeichen $a_1 \in \Sigma$. Ersetze a_1 durch $\bar{a}_1$ und markiere a_1 so als gelesen. Wechsle in einen Zustand z_{a_1}, der das gelesene Zeichen kodiert.
- Laufe ans Wortende und ersetze das erste gefundene $\square$-Zeichen durch $\tilde{a}$.
- Laufe an den Wortanfang und suche das erste unmarkierte Zeichen a_2.
- Wechsle in den Startzustand z_0 und fahre fort wie oben.
- Falls kein unmarkiertes Zeichen mehr gefunden wird, laufe an den Wortanfang und entferne alle Markierungen.

4.4

a) Die Idee hierbei ist, sukzessive jede zweite 1 zu löschen, d. h., die Inputzahl durch 2 zu teilen, bis nur noch eine 1 übrig ist. Ist dies nicht möglich, weil eine ungerade Anzahl 1 auf dem Band steht, ist die Zahl keine Zweierpotenz.

$$
\begin{aligned}
\delta(z_0, 1) &= (z_1, 1, R) \\
\delta(z_0, \star) &= (z_0, \star, R) \\
\delta(z_1, 1) &= (z_2, \star, R) \\
\delta(z_1, \star) &= (z_1, \star, R) \\
\delta(z_1, \square) &= (z_e, \square\, N) \text{ Nur eine 1 übrig.} \\
\delta(z_2, 1) &= (z_3, 1, R) \\
\delta(z_2, \star) &= (z_2, \star, R) \\
\delta(z_2, \square) &= (z_4, \square, L) \text{ Laufe zum Wortanfang.} \\
\delta(z_3, 1) &= (z_2, \star, R) \\
\delta(z_3, \star) &= (z_3, \star, R) \\
\delta(z_3, \square) &= (\otimes, \square, N) \text{ Ungerade Anzahl 1.} \\
\delta(z_4, 1) &= (z_4, 1, L) \\
\delta(z_4, \star) &= (z_4, \star, L) \\
\delta(z_4, \square) &= (z_0, \square, R)
\end{aligned}
$$

b) Idee: Zähle die Anzahl der Divisionen in Teil a). Dazu ersetzen wir den obigen Übergang $\delta(z_2, \square) = (z_4, \square, L)$ durch

$$\delta(z_2, \square) = (z_5, \square, R)$$

und nutzen den neuen Zustand, um die Anzahl der Divisionen zu notieren. Danach laufen wir wieder an den Wortanfang:

$$
\begin{aligned}
\delta(z_5, \square) &= (z_6, 1, L) \\
\delta(z_5, 1) &= (z_5, 1, R) \\
\delta(z_6, 1) &= (z_6, 1, L) \\
\delta(z_6, \square) &= (z_4, \square, L)
\end{aligned}
$$

4.5

Bei der Angabe der Übergangsfunktion kann der Input von Band 3 ignoriert werden, da dieses nur dazu dient, das Ergebnis zu notieren. Auf Band 1 und Band 2 wird der Input nicht verändert.

$$
\begin{aligned}
\delta(z_0, 0, 0, \square) &= (z_0, 0, 0, 0, L, L, L) \\
\delta(z_0, 0, 1, \square) &= (z_0, 0, 1, 1, L, L, L) \\
\delta(z_0, 1, 0, \square) &= (z_0, 1, 0, 1, L, L, L) \\
\delta(z_0, 1, 1, \square) &= (z_1, 1, 1, 0, L, L, L) \\
\delta(z_0, \square, 1, \square) &= (z_0, \square, 1, 1, L, L, L) \\
\delta(z_0, 1, \square, \square) &= (z_0, 1, \square, 1, L, L, L)
\end{aligned}
$$

$$
\begin{aligned}
\delta(z_0, \square, 0, \square) &= (z_0, \square, 0, 0, L, L, L) \\
\delta(z_0, 0, \square, \square) &= (z_0, 0, \square, 0, L, L, L) \\
\delta(z_0, \square, \square, \square) &= (z_e, \square, \square, \square, R, R, R) \\
\delta(z_1, 0, 0, \square) &= (z_0, 0, 0, 1, L, L, L) \\
\delta(z_1, 0, 1, \square) &= (z_1, 0, 1, 0, L, L, L) \\
\delta(z_1, 1, 0, \square) &= (z_1, 1, 0, 0, L, L, L) \\
\delta(z_1, 1, 1, \square) &= (z_1, 1, 1, 1, L, L, L) \\
\delta(z_1, \square, 1, \square) &= (z_1, \square, 1, 0, L, L, L) \\
\delta(z_1, 1, \square, \square) &= (z_1, 1, \square, 0, L, L, L) \\
\delta(z_1\square, 0, \square) &= (z_0, \square, 0, 1, L, L, L) \\
\delta(z_1, 0, \square, \square) &= (z_0, 0, \square, 1, L, L, L) \\
\delta(z_1, \square, \square, \square) &= (z_e, \square, \square, 1, N, N, N)
\end{aligned}
$$

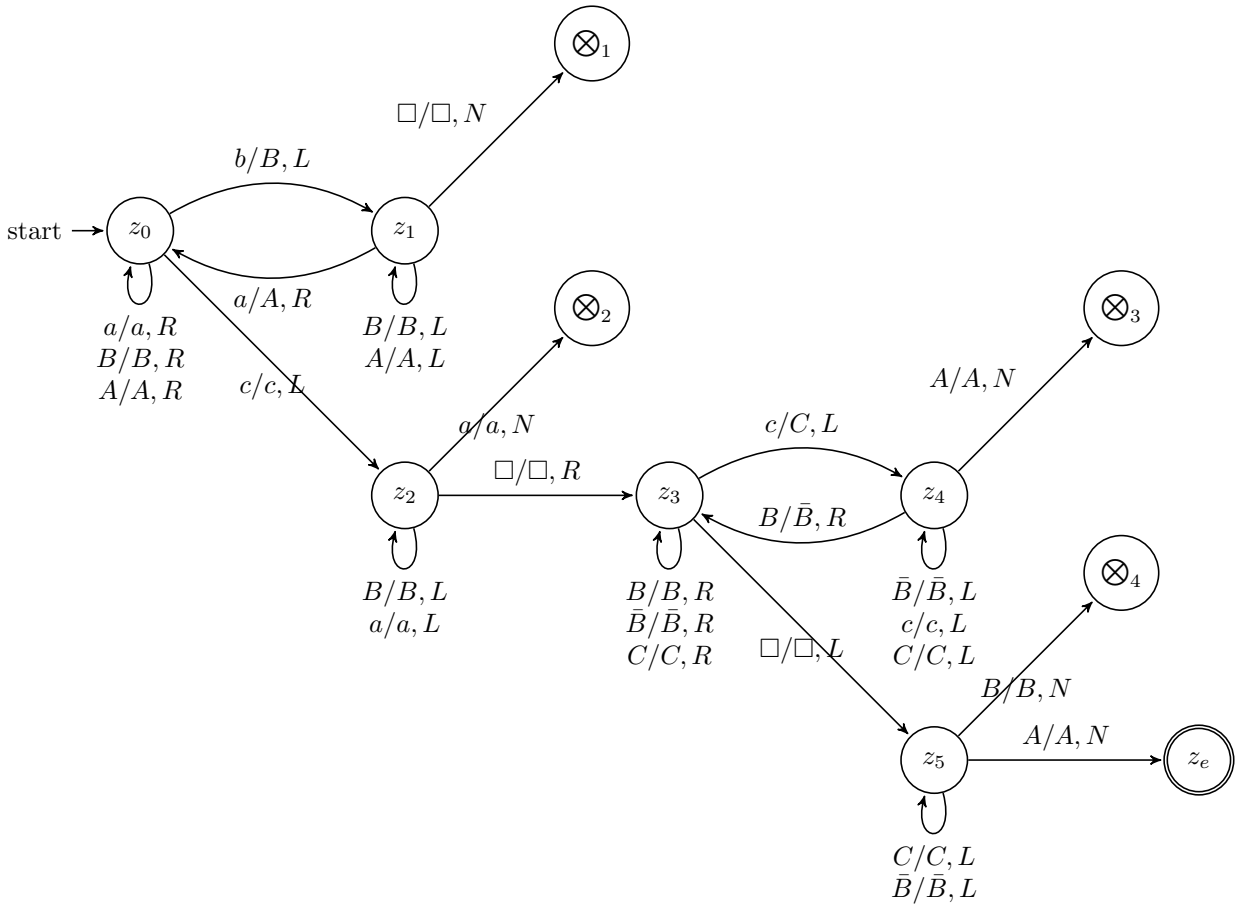
4.6

Wir betrachten die Sprache $L = \{w \in \Sigma^* | w = a^n b^n c^n, n \geq 1\}$ über dem Alphabet $\Sigma = \{a, b, c\}$.

a) Idee für eine 2-Band TM: Schreibe die Anzahl der a's auf Band 2 und vergleiche dann nach einander mit den b's und c's.

$$
\begin{aligned}
\delta(z_0, a, \square) &= (z_0, a, A, R, R) \\
\delta(z_0, b, \square) &= (z_1, b, \square, N, L) \\
\delta(z_1, b, A) &= (z_1, b, A, R, L) \text{ Laufe auf Band 1 vorwärts und auf Band 2 zurück} \\
\delta(z_1, c, \square) &= (z_2, c, \square, N, R) \text{ Anzahl } a = \text{Anzahl } b. \\
\delta(z_1, c, A) &= (\otimes, c, A, N, N) \text{ Anzahl } a > \text{Anzahl } b. \\
\delta(z_1, b, \square) &= (\otimes, b, \square, N, N) \text{ Anzahl } a < \text{Anzahl } b. \\
\delta(z_2, c, A) &= (z_2, c, A, R, R) \text{ Laufe auf beiden Bändern vorwärts.} \\
\delta(z_2, \square, \square) &= (z_e, \square, \square, N, N) \text{ Anzahl } a = \text{Anzahl } c. \\
\delta(z_2, c, \square) &= (\otimes, c, \square, N, N) \text{ Anzahl } c > \text{Anzahl } a. \\
\delta(z_2, \square, A) &= (\otimes, \square, A, N, N) \text{ Anzahl } c < \text{Anzahl } a.
\end{aligned}
$$

b) Das Übergangsdiagramm einer 1-Band-TM, die L akzeptiert:



Die einzelnen Zustände haben die folgenden Aufgaben:

- z_0 sucht und markiert ein b
- z_1 sucht und markiert dazu passendes a.
- $\otimes_1$ ist ein Fehlerzustand, der angenommen wird, falls Anzahl $b >$ Anzahl a.
- z_2 sucht nach überschüssigen a's.
- $\otimes_2$ ist ein Fehlerzustand, der angenommen wird, falls Anzahl $a >$ Anzahl b.
- z_3 wird angenommen, falls Anzahl $a =$ Anzahl b. Er sucht und markiert ein c.
- z_4 sucht ein zu c passendes B und markiert es.
- $\otimes_3$ ist ein Fehlerzustand, der angenommen wird, falls Anzahl $c >$ Anzahl B.
- z_5 sucht nach überschüssigen B's.
- $\otimes_4$ ist ein Fehlerzustand, der angenommen wird, falls Anzahl $B >$ Anzahl c.
- Der Endzustand z_e wird angenommen, falls Anzahl $B =$ Anzahl c.

4.7

Die im Buch auf Seite 100 beschriebene deterministische TM akzeptiert zwar die Sprache $L = \{w \in \Sigma^* | w = uu, u \in \Sigma^* \text{ beliebig.}\}$, ist aber keine LBTM, da sie für die Zählsymbole X zusätzliche Felder auf dem Band beschreiben muss. Eine **nichtdeterministische** LBTM könnte einfach die Wortmitte raten:

- Im Startzustand z_0 lese das erste Zeichen $a_1 \in \Sigma$. Ersetze a_1 durch $\bar{a}_1$ und markiere a_1 so als gelesen. Wechsle in einen Zustand z_{a_1}, der das gelesene Zeichen kodiert.
- Laufe nach rechts und rate, wo die zweite Worthälfte beginnt. Vergleiche das Zeichen dort mit a_1. Falls es mit a_1 übereinstimmt, ersetze es durch X und laufe zum Wortanfang zurück. Falls nicht, gehe in den Ablehnzustand.
- Suche das erste unmarkierte Zeichen a_2 und gehe in den Startzustand.
- Vergleiche a_2 mit dem ersten Zeichen rechts nach X, usw.
- Falls keine unmarkierten Zeichen und keine Zeichen nach den X mehr vorhanden sind, gehe in den Akzeptanzzustand.

4.8

M sei eine TM, die L entscheidet, d. h., sie akzeptiert die Wörter aus L und für $x \notin L$ geht M nach endlich vielen Schritten in einen Ablehnzustand. Durch Umdefinieren des Ablehnzustands in einen Akzeptanzzustand wird aus M also eine TM $\bar{M}$, die $\bar{L}$ akzeptiert.