Musterlösungen zu Grundlagen der theoretischen Informatik (skizziert)

(3a) $L_2 = (a+\varepsilon)(ba)^*(b+\epsilon)$ oder $L_2 = \varepsilon + L_2'$ mit $L_2' = a(ba)^*(b+\varepsilon) + b(ab)^*(a+\varepsilon)$ oder $L_2' = a + (a+\varepsilon)(ba)^*b(a+\varepsilon)$
$L_3 = \varepsilon + L_3'$ mit $L_3' = L_2' + L_2(cL_2')^*cL_2$
$L_4 = \varepsilon + L_3' + L_3(dL_3')^*dL_3$

(3b) $(aa^+ + bb^+ + cc^+)^*$

(3c) $l_n = l_{n-1} + (|\Sigma| - 1)l_{n-2}$. (Wörter mit mindestens drei gleichen Buchstaben am Ende, plus Wörter, deren letzter Block genau die Länge 2 hat). $l_1 = 0, l_2 = |\Sigma|$. Für Σ_2 ergeben sich die doppelten Fibonacci-Zahlen. Für Σ_3 erhält man $l_n = (2^{n+1} - 2(-1)^n)/3$. Allgemein ist

$$l_n = \left[\left(\tfrac{1}{2} + \sqrt{|\Sigma| - \tfrac{3}{4}}\right)^n - \left(\tfrac{1}{2} - \sqrt{|\Sigma| - \tfrac{3}{4}}\right)^n\right] \Big/ \sqrt{|\Sigma| - \tfrac{3}{4}}.$$

(8b) keine Kongruenzrelation für $|\Sigma| \geq 2$: $a = d = \varepsilon$ und $b, c \in \Sigma$, $b \neq c$.

(8c) Es ist eine Kongruenzrelation. Die Eigenschaften einer Äquivalenzrelation weist man leicht nach.
Wenn $a = b = \varepsilon$, ist $ac = c$ und $bd = d$, und die Aussage ist trivial. Wenn $a \neq \varepsilon \neq b$ ist, dann hat ac den selben Anfangsbuchstaben wie a und bd wie d.

(8d) keine Kongruenzrelation: $a = d = 0$, $b = c = 1$.

(10a) $F = \{q_6\}$,

δ	q_0	q_1	q_2	q_3	q_4	q_5	q_6
0	q_1	q_1	q_3	q_1	q_5	q_1	q_6
1	q_0	q_2	q_0	q_4	q_0	q_6	q_6

(10b) der gleiche Automat wie (10a), mit $F' = Q - F$.

(10f) Die Wörter $\varepsilon, 0, 01, 010, 0101, 01010, 010101$ sind in verschiedenen Klassen $L_q(M)$, weil die Relation in der letzten Zeile von Aufgabe 9 nicht erfüllt ist.
Zum Beispiel für $w_1 = 010$ und $w_2 = 01010$ gilt: für $x = 11$: $w_1x \notin L$, aber $w_2x \in L$. Im Allgemeinen kann man x so wählen, dass $wx = 0101011$ ist, wobei w das kürzere der Worte w_1 und w_2 ist. Daher hat M mindestens 7 Zustände. Dieses zeigt auch, dass der Automat aus Aufgabe 10a der Minimalautomat ist.

(14) Man beginnt mit

$$L_{ij}^{(0)} = \{a \in \Sigma | (i, a, j) \in \delta\} \quad (\cup\{\varepsilon\}, \text{falls } i = j \text{ ist}).$$

Am Ende muss man $L(M)$ als Vereinigung der $L_{q_0,f}^n$ auch über alle Startzustände $q_0 \in Q_0$ bilden.

(15) NEA's: Erzeuge einen neuen (einzigen) Endzustand f. Für alle Übergänge, die zu einem bisherigen akzeptierenden Zustand geführt haben, füge einen entsprechenden Übergang zu f hinzu. Formal:

$$\delta' := \delta \cup \{\, (q, a, f) \mid (q, a, r) \in \delta \text{ für ein } r \in F \,\}.$$

Falls $\varepsilon \in L(M)$ ist, wird f überdies zu einem Anfangszustand erklärt.

(Man kann auch erreichen, dass es nur einen einzigen Anfangszustand **und** nur einen einzigen Endzustand gibt.)

DEA's: Solche Automaten erkennen nicht alle Sprachen. Beispiel: $0^* + 1^*$. Mit Aufgabe 9 kann man zeigen, dass $\delta(q_0, 0) \neq \delta(q_0, 1)$ sein muss. Beides sind jedoch akzeptierende Zustände.

(16) Zu jedem NEA gibt es einen äquivalenten DEA. Dieser ist ein NEA mit einem einzigen Anfangszustand. Die Zusatzfrage geht ähnlich wie Aufgabe 15.

(17a) DEA und NEA benötigen beide mindestens 5 Zustände. Nichtdeterminismus bringt keine Ersparnis.

(17b) Der NEA kommt mit $5 = n + 1$ Zuständen aus (Umkehrung von 17a). Ein DEA braucht mindestens $16 = 2^n$ Zustände. Im allgemeinen ist der Unterschied zwischen NEA und DEA in diesem Beispiel exponentiell.

(19) $Q = Q_1 \times Q_2$, $q = (q_{10}, q_{20})$, $F = F_1 \times F_2$, $\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$

(26a) Wende das Pumping-Lemma auf $w = 0^n 1 0^n$ an. L_1 ist nicht regulär.

(26b) Wende das Pumping-Lemma auf $w = 0^{n^2}$ an. $n^2 + k$ ist für kein k mit $1 \leq k \leq n$ eine Quadratzahl, und somit ist $0^{n^2+k} \notin L_2$. L_2 ist nicht regulär.

(26c–e) $L_5 \cap \{00, 01, 10, 11\}^* \cap 0^* 1^n$ ist die Sprache $\{\, 0^n 1^n \,\}$, die bekanntlich nicht regulär ist. Da die regulären Sprachen abgeschlossen unter $\cap$ sind, ist auch L_5 nicht regulär. Dasselbe Argument zeigt, dass L_4 nicht regulär ist. L_3 ist ebenfalls nicht regulär.

(26f) $L_6 = \{01, 10\}^* \cdot \{\varepsilon, 0, 1\}$ ist regulär.

(28) $h_3(L_4) = (01)^*$ und $h_2(L_1) = 1^*$ sind regulär. $h_1(L)$ ist genau dann regulär, wenn L regulär ist.

(30) Beginnend von rechts, wird jeweils das erste nicht markierte Bit von y und von x markiert. Wenn dieses Bit von y gleich 1 ist, wird es zur entsprechenden Stelle von x addiert, und gegebenenfalls ein Übertrag nach links weitergereicht. Ein B-Symbol am Anfang von x wird dabei wie eine 0 behandelt. Wenn alle Stellen von y markiert sind, werden alle Markierungen gelöscht, und der Automat hält.

δ	0	1	$\bar{0}$	$\bar{1}$	#	$	B	Kommentar
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	$(q_0, \bar{0}, R)$	$(q_0, \bar{1}, R)$	$(q_0, \#, R)$	$(q_1, \$, L)$	–	fahre nach rechts
q_1	$(q_2, \bar{0}, L)$	$(q_4, \bar{1}, L)$	$(q_1, \bar{0}, L)$	$(q_1, \bar{1}, L)$	$(q_7, \#, R)$	–	–	finde erstes nicht markiertes
								Bit y_0 von y und markiere es
q_2	$(q_2, 0, L)$	$(q_2, 1, L)$	–	–	$(q_3, \#, L)$	–	–	falls $y_0 = 0$, finde erstes unmar-
q_3	$(q_0, \bar{0}, R)$	$(q_0, \bar{1}, R)$	$(q_3, \bar{0}, L)$	$(q_3, \bar{1}, L)$	–	–	$(q_0, \bar{0}, R)$	kiertes Bit von x, markiere es.
q_4	$(q_4, 0, L)$	$(q_4, 1, L)$	–	–	$(q_5, \#, L)$	–	–	falls $y_0 = 1$, finde erstes Bit
q_5	$(q_0, \bar{1}, R)$	$(q_6, \bar{0}, L)$	$(q_5, \bar{0}, L)$	$(q_5, \bar{1}, L)$	–	–	$(q_0, \bar{1}, R)$	von x, markiere es, addiere 1,
q_6	$(q_0, 1, R)$	$(q_6, 0, L)$	–	–	–	–	$(q_0, 1, R)$	gegebenenfalls Übertrag
q_7	–	–	$(q_7, \bar{0}, R)$	$(q_7, \bar{1}, R)$	–	$(q_8, \$, L)$	–	Ende, fahre ganz nach rechts
q_8	$(q_8, 0, L)$	$(q_8, 1, L)$	$(q_8, 0, L)$	$(q_8, 1, L)$	$(q_8, \#, L)$	–	(q_F, B, R)	nach links, lösche Markierungen

$q_0\texttt{101\#11\$} \vdash \texttt{1}q_0\texttt{01\#11\$} \vdash \texttt{10}q_0\texttt{1\#11\$} \vdash \texttt{101}q_0\texttt{\#11\$} \vdash \texttt{101\#}q_0\texttt{11\$} \vdash \texttt{101\#1}q_0\texttt{1\$} \vdash \texttt{101\#11}q_0\texttt{\$} \vdash \texttt{101\#1}q_1\texttt{1\$} \vdash \texttt{101\#}q_4\texttt{1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{101}q_4\texttt{\#1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}q_5\texttt{1\#1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{1}q_6\texttt{0}\bar{\texttt{0}}\texttt{\#1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{11}q_0\bar{\texttt{0}}\texttt{\#1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{11}\bar{\texttt{0}}q_0\texttt{\#1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{11}\bar{\texttt{0}}\texttt{\#}q_0\texttt{1}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{11}\bar{\texttt{0}}\texttt{\#1}q_0\bar{\texttt{1}}\texttt{\$} \vdash \cdots \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}\bar{\texttt{1}}\bar{\texttt{1}}q_0\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}\bar{\texttt{1}}q_1\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}q_1\bar{\texttt{1}}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}q_1\texttt{\#}\bar{\texttt{1}}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}q_7\bar{\texttt{1}}\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}\bar{\texttt{1}}q_7\bar{\texttt{1}}\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}\bar{\texttt{1}}\bar{\texttt{1}}q_7\texttt{\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}\bar{\texttt{1}}q_8\texttt{1\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}\texttt{\#}q_8\bar{\texttt{1}}\texttt{1\$} \vdash \texttt{10}\bar{\texttt{0}}\bar{\texttt{0}}q_8\texttt{\#11\$} \vdash \texttt{10}\bar{\texttt{0}}q_8\bar{\texttt{0}}\texttt{\#11\$} \vdash \texttt{10}q_8\bar{\texttt{0}}\texttt{0\#11\$} \vdash \texttt{1}q_8\texttt{000\#11\$} \vdash q_8\texttt{1000\#11\$} \vdash q_8\texttt{B1000\#11\$} \vdash q_F\texttt{1000\#11\$}$.

(33) Fall 1. $y = \varepsilon$:

Fall 1L. $\delta(q, B) = (q', a, L)$.

Fall 1Lε. $x = \varepsilon$: Dann ist $x' = \varepsilon$ und $y' = Ba$.

Fall 1L+. $x = ux_0$ mit $x_0 \in \Gamma, u \in \Gamma^*$: Dann ist $x' = u$ und $y' = x_0a$.

Fall 1N. $\delta(q, B) = (q', a, N)$: Dann ist $x' = x$ und $y' = a$.

Fall 1R. $\delta(q, B) = (q', a, R)$: Dann ist $x' = xa$ und $y' = \varepsilon$.

Fall 2. $y = y_1z$ mit $y_1 \in \Gamma, z \in \Gamma^*$:

Fall 2L. $\delta(q, y_1) = (q', a, L)$:

Fall 2Lε. $x = \varepsilon$: Dann ist $x' = \varepsilon$ und $y' = Baz$.

Fall 2L+. $x = ux_0$ mit $x_0 \in \Gamma, u \in \Gamma^*$: Dann ist $x' = u$ und $y' = x_0az$.

Fall 2N. $\delta(q, y_1) = (q', a, N)$: Dann ist $x' = x$ und $y' = az$.

Fall 2R. $\delta(q, y_1) = (q', a, R)$: Dann ist $x' = xa$ und $y' = z$.

In allen Fällen kann man noch, wenn man will, $x' = B$ oder $y' = B$ durch $x' = \varepsilon$ beziehungsweise $y' = \varepsilon$ ersetzen.

(35a) Wahr. Man muss nur beim Entscheidungsalgorithmus die Antworten JA und NEIN vertauschen.

Bei einem Beweis auf der Ebene von Turingmaschinen muss man zunächst sicherstellen, dass die Maschine nur dann in einen akzeptierenden Zustand geht, wenn sie anhält. Man kann eine Turingmaschine, die immer hält, so modifizieren, dass es nur zwei Haltezustände gibt, einen akzeptierenden und einen nicht akzeptierenden. Durch Vertauschen dieser beiden Zustände erhält man eine Turingmaschine für die komplementäre Sprache.

(35b) Wahr. Überprüfe zuerst, ob $w \in L_1$ ist. Wenn ja, akzeptiere das Wort w. Wenn nein, überprüfe, ob $w \in L_2$ ist, und akzeptiere w genau dann wenn $w \in L_2$ ist.

(35c) Wahr. Die Aussage ist logisch äquivalent zu (35a).

(35d) Falsch. Wenn D eine beliebige unentscheidbare Sprache ist, dann ist auch ihr Komplement $\bar{D}$ unentscheidbar. $D \cup \bar{D} = \Sigma^*$ ist jedoch trivialerweise entscheidbar.

(36a) Entscheidungsalgorithmus für das Eingabewort $w = w_1 \ldots w_n$: Für alle $i = 0, \ldots, n$, teste ob $w_1 \ldots w_{i-1} \in L_1$ und $w_i \ldots w_n \in L_2$ ist. Wenn ja, akzeptiere w. Wenn die Schleife beendet wird, verwirf w.

(36b) M_1 und M_2 seien zwei Maschinen mit $L(M_1) = L_1$ und $L(M_2) = L_2$, Für das Eingabewort $w = w_1 \ldots w_n$ gehen wir der Reihe nach alle Tripel (k_1, k_2, i) mit $k_1, k_2 \geq 0$, $0 \leq i \leq n$ durch (in irgendeiner passenden Reihenfolge), und testen durch Simulation ob M_1 das Wort $w_1 \ldots w_{i-1}$ in k_1 Schritten akzeptiert und ob M_2 das Wort $w_i \ldots w_n \in L_2$ in k_2 Schritten akzeptiert. Wenn ja, akzeptieren wir w.

(37) Die Vorwärtsrichtung ergibt sich aus (35a). Wenn sowohl L als auch $\bar{L}$ rekursiv aufzählbar ist, gibt es Maschinen M und M' mit $L(M) = L$ und $L(M') = \bar{M}$. Lass die Maschinen M und M' „parallel" laufen; das heißt, simuliere abwechselnd einen Schritt von M und einen Schritt von M'. Mindestens eine der beiden Maschinen muss halten, und dann hat man entschieden, ob das Wort zur Sprache L gehört.

(39) Wenn M_0 eine beliebige Turingmaschine ist, dann gilt nach Definition, mit $M = M_0$ und $w = \langle M_0 \rangle$:

$$\langle M_0 \rangle \langle M_0 \rangle \in U' \iff M_0 \text{ akzeptiert das Wort } \langle M_0 \rangle \langle M_0 \rangle \text{ nicht}$$

Daher ist $L(M_0) \neq U'$, weil sich die beiden Sprachen zumindest im Wort $\langle M_0 \rangle \langle M_0 \rangle$ unterscheiden. Es gibt also keine Turingmaschine, die U' akzeptiert, und U' ist nicht einmal rekursiv aufzählbar.

(40a) L_1 is rekursiv. Es genügt, die ersten k Schritte der Maschine M zu simulieren, um die Zugehörigkeit zur Sprache L_1 zu entscheiden.

(40b–c) L_2 ist rekursiv, mit dem gleichen Algorithmus wie für Aufgabe 40a. Die rekursiven Sprachen sind abgeschlossen gegenüber Vereinigung und Komplement, daher ist auch L_3 is rekursiv.

(40d) L_4 is rekursiv aufzählbar, aber nicht rekursiv. Eine Turingmaschine für L_4 braucht nur die Maschine M zu simulieren, und die Eingabe zu akzeptieren, wenn M mindestens k Schritte läuft und irgendwann danach in einem akzeptierenden Zustand hält.

Wenn L_4 rekursiv wäre, dann wäre auch die Sprache $(L_4 \cup L_1) \cap 1\{0,1\}^*$ (das sind die Eingaben mit $k = 0$) rekursiv; dies ist aber die universelle Sprache U.

(41) Wenn L endlich ist, ist es trivialerweise entscheidbar. Wenn L unendlich ist, muss $L = 0^*$ sein, denn mit 0^i gehören auch alle 0^j mit $j < i$ zu L. Auch diese Sprache ist klarerweise entscheidbar.

(44b) Wenn die Sprache L rekursiv ist, dann kann eine Maschine M alle Wörter von Σ^* in der passenden Reihenfolge untersuchen, und diejenigen, die zu L gehören, auf das Ausgabeband schreiben.

Nehmen wir umgekehrt an, dass es eine solche Maschine M gibt. Falls die Sprache L endlich ist, ist sie sowieso rekursiv. Falls die Sprache L unendlich ist, dann brauchen wir zur Entscheidung, ob ein Wort $w \in L$ ist, nur zu warten, bis die Maschine M ein Wort schreibt, das länger als w ist. Wenn das Wort w bis dahin noch nicht auf das Band geschrieben wurde, dann gehört es nicht zu L.

(46) Zwischen den Randmarkierungen A und B fährt das „Schiffchen" C/D beliebig oft hin und her. Beim Weg nach rechts verdoppelt das C alle a's, die es trifft. Am rechten Rand wird das C in D umgewandelt, und es fährt an den linken Rand, wobei es die a's unverändert überspringt. Am linken Rand wird D wieder in C umgewandelt, und er ganze Prozess kann von vorne beginnen. In jedem Durchlauf wird die Gesamtanzahl der a's verdoppelt. Am rechten Rand kann das C auch mit dem B zu E verschmelzen.

Das E wandert dann nach links und löst sich zusammen mit dem A auf. Damit ist der Ableitungsprozess beendet.

Die erzeugte Sprache ist $\{ a^{2^i} \mid i \geq 1 \}$.

(48) $S \to S + B \mid S - B \mid B \qquad F \to V \uparrow F \mid V$
$B \to B * F \mid B \div F \mid F \qquad V \to (S) \mid Z$

(54a) L_1 ist nicht kontextfrei. Beweis durch Widerspruch: N sei die Konstante aus dem Pumping-Lemma. Dann ist $0^N 1^{N^2} = uvxyz$ und alle Wörter uv^ixy^iz gehören zu L_1. Die Möglichkeiten $vxy \in 0^+$ und $vxy \in 1^+$ führen direkt zu einem Widerspruch, ebenso $v \in 0^+1^+$, $y \in 0^+1^+$, oder $y = \varepsilon$. Es ist also $v \in 0^*$ und $y = 1^k$ für $1 \leq k \leq N$ Somit ist $uv^2xy^2z \in 0^*1^{N^2+k}$. Jedoch ist N^2+k keine Quadratzahl. Die nächstgrößere Quadratzahl nach N^2 ist $(N+1)^2 = N^2 + 2N + 1 > N + k$

(54b) $S \to 0S \mid 0S1 \mid \varepsilon$ erzeugt L_2.

(54c) L_3 ist nicht kontextfrei. Beweis durch Widerspruch mit dem Pumping-Lemma. N sei die Konstante aus dem Pumping-Lemma. Dann ist $w = 10^N10^N10^N10^N = uvxyz$ und alle Wörter uv^ixy^iz gehören zu L_3. Wegen $|vxy| \leq N$ kann vxy höchstens eine 1 enthalten. Wenn vy eine Eins enthält, dann enthält $uxz \in L_3$ genau drei Einsen, ein Widerspruch. Es folgt, dass $vy \in 0^+$ ist. Somit ist jedes Wort uv^ixy^iz von der Form $0^j10^k10^l10^m$. Ein solches Wort ist nur dann ein Doppelwort, wenn $j = l$ und $k = m$ ist. vxy kann nur Nullen aus einem oder aus zwei aufeinanderfolgenden Nullerblöcken enthalten. Das heißt, die Wörter $uv^ixy^iz = 0^j10^k10^l10^m$ haben zwei oder drei Nullblöcke konstanter Länge und ein oder zwei benachbarte Nullblöcke variabler Länge. Unter diesen Einschränkungen kann aber die Bedingung $j = l$ und $k = m$ nicht aufrechterhalten werden.

(54d) L_4: $S \to 0S1 \mid T_1 \mid U_1, \qquad T_1 \to 0T_10 \mid T_2, \qquad T_2 \to 1T_20 \mid \varepsilon,$
$U_1 \to 1U_11 \mid U_2, \qquad U_2 \to 0U_21 \mid \varepsilon.$

L_5: $S \to 0S1 \mid S_{10}S_{01} \mid S_{01}S_{10}$, $S_{01} \to 0S_{01}1 \mid \varepsilon$, $S_{10} \to 1S_{10}0 \mid \varepsilon$. (Mit Aufgabe 69 könnte man auch aus $L_5 = \{\text{gleich viele Nullen und Einsen}\} \cap 0^*1^*0^*1^*$ folgern, dass L_5 kontextfrei ist.)

(54e) S: $\#(0) = 2 \cdot \#(1)$. $\quad S_+$: $\#(0) = 2 \cdot \#(1) + 1$. $\quad S_-$: $\#(0) = 2 \cdot \#(1) - 1$.

$S \to 1S_+S_+ \mid 0S_- \mid \varepsilon$, $\quad S_+ \to 1S_+S_+S_+ \mid 0S$, $\quad S_- \to 1S_+ \mid 0S_-S_-$.

(55) Dass jedes Wort der erzeugten Sprache diese Eigenschaft hat, beweist man leicht mit vollständiger Induktion nach der Länge der Ableitung.

Wir beweisen nun mit Induktion nach der Länge n, dass jedes Wort w der Länge n mit der angegebenen Eigenschaft von der Grammatik erzeugt wird. Der Induktionsanfang $n = 0$ ist offensichtlich erfüllt. Falls in w jeder nichtleere Präfix x *mehr* a's als b's enthält, dann ist $w = aw'$, und w' hat die angegebene Eigenschaft. Also wird w nach Induktion durch die Ableitung $S \to aS \xrightarrow{*} aw'$ erzeugt.

Im anderen Fall sei w_1 der kürzeste nichtleere Präfix von w, in dem die a's und b's ausgeglichen sind. Dann ist $w = w_1w_2$ und $w_1 = aw_1'b$. Man überlegt sich leicht, dass w_1' und w_2 zur Sprache gehören, und somit kann w abgeleitet werden, indem man mit der zweiten Regel anfängt: $S \to aSbS \xrightarrow{*} aw_1'bS \xrightarrow{*} aw_1'bw_2$.

(56) Die Grammatik

$$\begin{aligned}
&S \to T \mid U \\
&T \to \textbf{if } (B)\ T \textbf{ else } T \mid \text{print(0);} \mid \text{print(1);} \mid \text{print(2);} \\
&U \to \textbf{if } (B)\ S \mid \textbf{if } (B)\ T \textbf{ else } U \\
&B \to \text{true} \mid \text{false}
\end{aligned}$$

ist eine äquivalente eindeutige Grammatik. Aus T sind alle Ausdrücke ableitbar, bei denen kein **else** mehr "fehlt".[1] Durch geeignete Klammerung kann man übrigens erreichen, dass das angegebene Programm auch 0 oder 1 druckt.[2]

(58) In einer Grammatik in CNF mit n Variablen, die eine endliche Sprache erzeugt, ist das längste Wort höchstens 2^{n-1} Zeichen lang. Der Beweis geht durch vollständige Induktion. Der Fall $n = 1$ ist einfach. Wir können annehmen, dass die Grammatik keine überflüssigen Variablen enthält. Der erste Ableitungsschritt sei von der Form $S \to AB$. Da die Sprache endlich ist, kann man aus A oder B keine Satzform ableiten, die die Variable S enthält. Daher ist nach Induktionannahme das längste Wort, das aus A erzeugt wird höchstens 2^{n-2} Zeichen lang, und ebenso für B. (Man kann A als das Startsymbol einer neuen Grammatik auffassen, bei der alle Regeln entfernt wurden, in denen S vorkommt.)

(59) Die Ableitung eines Wortes w mit $|w| = n$ hat genau $2n - 1$ Ableitungsschritte, für $n \geq 1$. Jede Regel der Form $A \to BC$ verlängert die Satzform um 1, daher werden diese Regeln genau $(n-1)$-mal angewendet. Außerdem muss man noch n Regeln der Form $A \to a$ anwenden.

(62) Aus welchen Variablen kann man Terminalworte erzeugen? Aus A und C, sodann aus D, S, und schließlich aus S. Die Variable B ist überflüssig. Somit kann man die Regeln $S \to AB$ und $C \to aB$ streichen. Damit kann man aus S nur noch Satzformen mit den Variablen C und A erzeugen. Die Variablen D und E sind nicht erreichbar. Die äquivalente Grammatik ohne überflüssige Variablen ist

$$S \to CA, \quad A \to a, \quad C \to c.$$

(63)

c	d	c	c	d
C	B,D	C	C	D
S,D	A	$\emptyset$	S,D	
C,A	$\emptyset$	S,D		
$\emptyset$	A,S			
C,A,S,D				

(66a) Der Automat ist ein binärer Zähler: Wenn man die Folge der Kellerinhalte an den Übergängen von q_0 nach q_1 betrachtet, dann wird von 0^n bis 1^n hochgezählt, wobei die niedrigwertigste Stelle ganz oben auf dem Stapel liegt. Der Eingabekopf wird

[1] siehe auch `http://www.inf.fu-berlin.de/~rote/Lere/Grundlagen%20der%20theoretischen%20Informatik/bison/Bedingungen.y`

[2] siehe auch `http://www.inf.fu-berlin.de/~rote/Lere/Grundlagen%20der%20theoretischen%20Informatik/bison/test-Aufgabe56`

immer synchron mit dem Wachsen und Schrumpfen des Kellers bewegt. Bei Eingabe von a^n benötigt der Automat $2^{n+2}-2$ Schritte.

$$
\begin{array}{l}
(q_0, \Diamond\underline{\$}, Z_0)\\
(q_1, \Diamond\underline{\$}, Z_0)\\
(q_1, \Diamond\underline{\$}, \varepsilon)
\end{array}
$$

$$
\begin{array}{l}
(q_0, \Diamond\underline{a}\$, Z_0)\\
(q_0, \Diamond a\underline{\$}, 0Z_0)\\
(q_1, \Diamond a\underline{\$}, 0Z_0)\\
(q_0, \Diamond a\underline{\$}, 1Z_0)\\
(q_1, \Diamond a\underline{\$}, 1Z_0)\\
(q_1, \Diamond\underline{a}\$, Z_0)\\
(q_1, \Diamond\underline{a}\$, \varepsilon)
\end{array}
$$

$$
\begin{array}{l}
(q_0, \Diamond\underline{a}a\$, Z_0)\\
(q_0, \Diamond a\underline{a}\$, 0Z_0)\\
(q_0, \Diamond aa\underline{\$}, 00Z_0)\\
(q_1, \Diamond aa\underline{\$}, 00Z_0)\\
(q_0, \Diamond aa\underline{\$}, 10Z_0)\\
(q_1, \Diamond aa\underline{\$}, 10Z_0)\\
(q_1, \Diamond a\underline{a}\$, 0Z_0)\\
(q_0, \Diamond a\underline{a}\$, 1Z_0)\\
(q_0, \Diamond aa\underline{\$}, 01Z_0)\\
(q_1, \Diamond aa\underline{\$}, 01Z_0)\\
(q_0, \Diamond aa\underline{\$}, 11Z_0)\\
(q_1, \Diamond aa\underline{\$}, 11Z_0)\\
(q_1, \Diamond a\underline{a}\$, 1Z_0)\\
(q_1, \Diamond\underline{a}a\$, Z_0)\\
(q_1, \Diamond\underline{a}a\$, \varepsilon)
\end{array}
$$

$$
\begin{array}{l}
(q_0, \Diamond\underline{a}aa\$, Z_0)\\
(q_0, \Diamond a\underline{a}a\$, 0Z_0)\\
(q_0, \Diamond aa\underline{a}\$, 00Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 000Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 000Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 100Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 100Z_0)\\
(q_1, \Diamond aa\underline{a}\$, 00Z_0)\\
(q_0, \Diamond aa\underline{a}\$, 10Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 010Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 010Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 110Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 110Z_0)\\
(q_1, \Diamond aa\underline{a}\$, 10Z_0)\\
(q_1, \Diamond a\underline{a}a\$, 0Z_0)\\
(q_0, \Diamond a\underline{a}a\$, 1Z_0)\\
(q_0, \Diamond aa\underline{a}\$, 01Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 001Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 001Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 101Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 101Z_0)\\
(q_1, \Diamond aa\underline{a}\$, 01Z_0)\\
(q_0, \Diamond aa\underline{a}\$, 11Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 011Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 011Z_0)\\
(q_0, \Diamond aaa\underline{\$}, 111Z_0)\\
(q_1, \Diamond aaa\underline{\$}, 111Z_0)\\
(q_1, \Diamond aa\underline{a}\$, 11Z_0)\\
(q_1, \Diamond a\underline{a}a\$, 1Z_0)\\
(q_1, \Diamond\underline{a}aa\$, Z_0)\\
(q_1, \Diamond\underline{a}aa\$, \varepsilon)
\end{array}
$$

(66b) Für $q \in Q$, $i \geq 1$, haben wir

$$
\begin{aligned}
E(q,i,0) = E(q,i,1) &= (q_1, i-1)\\
E(q,i,Z_0) &= (q_1, i)
\end{aligned}
$$

Für $i = 0$ hängt die Entladefunktion davon ab, wie man $\delta(\cdot,\cdot,\Diamond)$ definiert. Wir können zum Beispiel $\delta(q,\gamma,\Diamond) = \delta(q,\gamma,a)$ oder $\delta(q,\gamma,\Diamond) = (q,\varepsilon,N)$ setzen.

(76) Ja.

(77) Nein. Zum Beispiel ist die Sprache

$$\{\, 00^i1^{2i} \mid i \geq 1 \,\} \cup \{\, 10^{2i}1^i \mid i \geq 1 \,\}$$

deterministisch kontextfrei, nicht aber ihre Umkehrung.