Aufgabenblatt 4

1. Gegeben sei die folgende Grammatik: $\mathcal{G} = (\mathcal{A}, \mathcal{V}, \mathcal{S}, \Pi)$
   $\mathcal{V} := \{S, A, M, D\}$, $\mathcal{A} := \{a, b, c\}$,
   $\Pi := \{$
   $\mathcal{S} \to AMD|M$
   $A \to AA|a$
   $D \to DD|d$
   $M \to bMc|\lambda$
   $\}$

   (a) Geben Sie die erzeugte Sprache an.

   (b) Wandeln Sie die gegebene kontextfrei Grammatik $\mathcal{G}$ in eine äquivalente kontextfrei Grammatik $\mathcal{G}$‘ in Chomsky-Normalform um, indem sie jeden Schritt durch eine neue Grammatik beschreiben.

   (c) Geben Sie die zwei verschiedene kontextfreie Ableitung des Wortes *aabbccdd* aus der Sprache $\mathcal{L}(\mathcal{G})$ an.

   (d) Geben Sie den Syntaxbaum oder wenn möglich Syntaxbäume in der Teilaufgabe c) gegebenen Sprache an. Wie nennt man, wenn man das Wort aus der Sprache mit verschiedenen Syntaxbäumen beschreiben kann?

   (e) Gegeben Sei folgende Grammatik in CNF: $\mathcal{G} = (\mathcal{A}, \mathcal{V}, \mathcal{S}, \Pi)$
   $\mathcal{V} := \{S, A, B, C, M, D, X, Y\}$, $\mathcal{A} := \{a, b, c\}$,
   $\Pi := \{$
   $S \to AX|AD|BY|BC|\lambda$
   $X \to MD$
   $A \to AA|a$
   $D \to DD|d$
   $M \to BY|BC$
   $Y \to MC$
   $B \to b$
   $C \to c$
   $\}$

   Zeigen oder Widerlegen Sie mit Hilfe des CYK-Algorithmus, ob die folgenden Wörter in der Sprache $\mathcal{L}$ liegen, die durch die Grammatik $\mathcal{G}$ erzeugt wird:

   i. aabbccdd
   ii. abbcc
   iii. abcdd

2. (a) Geben Sie für die Sprache $\mathcal{L} = \{a^n b^n c^n : n \in \mathbb{N}\}$ die Grammatik des höchstmöglichen Chomsky-Typs an.

   (b) Geben Sie für die Sprache $\mathcal{L}‘ = \{a^{2^n} : n \in \mathbb{N}\}$ eine linear beschränkte Turing-Maschine formal an und zeichnen Sie diese Turing-Maschine, die die Wörter aus der Sprache als Eingabe akzeptiert.

   (c) Prüfen Sie, ob ihre Turing-Maschine *aaaa* als Eingabe akzeptiert, probieren Sie auch, ob *aaa* nicht akzeptiert wird.

3. (a) Geben Sie eine Turing-Maschine M_1 an, die eine binäre Eingabe $\{0, 1\}^n$, $n \geq 1$ um Eins erhöht, wobei der Lesekopf am Ende wieder am Anfang des Wortes sein soll.

   (b) Geben Sie eine Turing-Maschine M_2 an, die für eine Zahl in Binärdarstellung das Einer-Komplement berechnet.

   (c) Geben Sie eine Turing-Maschine M_3 an, die für eine Zahl in Binärdarstellung das Zweier-Komplement berechnet.

4. (a) Geben sei ein deterministischer endlicher Automat $M = (\mathcal{A}, \mathcal{Q}, \delta, q_0, \mathcal{F})$. Geben Sie eine Turingmaschine TM an, die dieselbe Sprache akzeptiert.

   (b) Kann man umgekehrt zu jeder Turingmaschine einen endlichen Automaten finden, der dieselbe Spraache akzeptiert? Begründen Sie ihre Antwort.

Lösungen

1. (a) Die Sprache $\mathcal{L}(\mathcal{G}) = \{a^m b^n c^n d^p : m, p \geq 1; n \geq 0\} \cup \{b^n c^n : n \geq 0\}$

   (b) Wir gehen in 3 Schritten vor und konstruieren zu $\mathcal{G}$ äquivalente Grammatiken $\mathcal{G}_i = (\mathcal{A}, \mathcal{V}_i, \mathcal{S}, \Pi_i))(i = 1, 2, 3)$, die der Normalform immer näher kommen

   - Jedes Terminalzeichen a$\in \mathcal{A}$ wird durch ein Nichtterminalzeichen A ersetzt und in den Produktionsmenge Π in der Form $A \rightarrow a$ hinzugefügt. Die neue Grammatik sieht folgendermaßen aus:
     $\mathcal{G}_1 = (\mathcal{A}, \mathcal{V}_1, \mathcal{S}, \Pi_1)$
     $\mathcal{V}_1 := \{S, A, M, D\}$, $\mathcal{A} := \{a, b, c\}$,
     $\Pi_1 := \{$
     $\mathcal{S} \rightarrow AMD|M$
     $A \rightarrow AA|a$
     $D \rightarrow DD|d$
     $M \rightarrow BMC|\lambda$
     $B \rightarrow b$
     $C \rightarrow c$
     $\}$
   - Die neue Grammatik ist λ-frei, d.h. die Produktionsmenge neben der eventuellen λ-Regel $S \rightarrow \lambda$ nur Regeln der Form

     $$X \rightarrow YZ \ (X, Y, Z \in \mathcal{V})$$
     $$X \rightarrow a \ (X \in \mathcal{V}, a \in \mathcal{A})$$

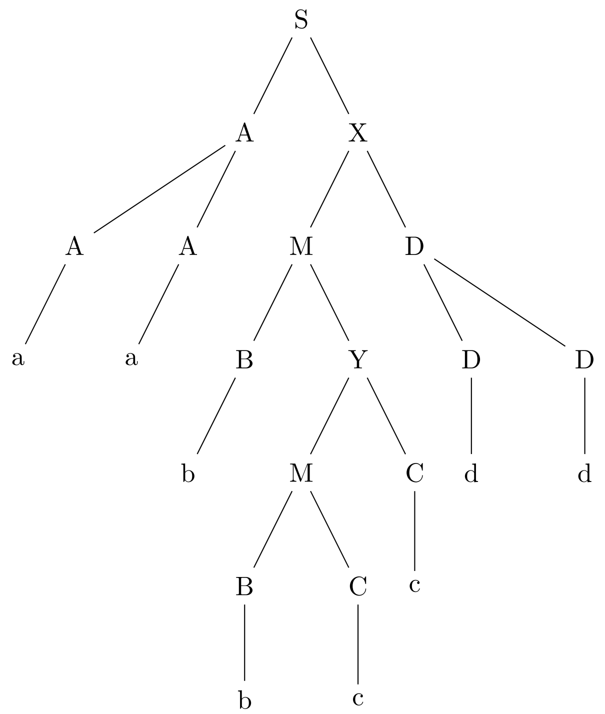
     besitzt.
     Wir erhalten hiermit $\mathcal{G}_2$ aus $\mathcal{G}_1$, indem alle Produktion wie $M \rightarrow \lambda$ an allen Produktionen, in denen rechts M abgeleitet wurde, wie $S \rightarrow AMD|M$ oder $M \rightarrow BMC|\lambda$, die neue Ableitungen hinzukommen: also hier wird λ durch das Ersetzen des Nichtterminals M an die Ableitung vom Startzeichen S verschoben und die neue Ableitungen hinzugefügt, wo M wieder durch λ (Leeres Zeichen) ersetzt wurde.
     $\mathcal{G}_2 = (\mathcal{A}, \mathcal{V}_2, \mathcal{S}, \Pi_2)$
     $\mathcal{V}_2 := \{S, A, B, C, M, D\}$, $\mathcal{A} := \{a, b, c\}$,
     $\Pi_2 := \{$
     $S \rightarrow AMD|AD|M|\lambda$
     $A \rightarrow AA|a$
     $D \rightarrow DD|d$
     $M \rightarrow BMC|BC$
     $B \rightarrow b$
     $C \rightarrow c$
     $\}$
   - In diesem Schritt werden die Ableitungen, die rechts das Wort der Länge mehr als drei erzeugen, durch den Trick ersetzt, wo neue Nichtterminale in die Grammatik reinkommen, aber dadurch uns die Chomsky-Normalform gewährleisten. $\mathcal{G}_3$ sieht dann so aus: $\mathcal{G}_3 = (\mathcal{A}, \mathcal{V}_3, \mathcal{S}, \Pi_3)$
     $\mathcal{V}_3 := \{S, A, B, C, M, D, X, Y\}$, $\mathcal{A} := \{a, b, c\}$,

$\Pi_3 := \{$
$S \rightarrow AX|AD|BY|BC|\lambda$
$X \rightarrow MD$
$A \rightarrow AA|a$
$D \rightarrow DD|d$
$M \rightarrow BY|BC$
$Y \rightarrow MC$
$B \rightarrow b$
$C \rightarrow c$
$\}$

(c) Die erste mögliche kontextfrei Ableitung: $S \Rightarrow AX \Rightarrow AAX \Rightarrow AAMD \Rightarrow AABYD \Rightarrow AABYDD \Rightarrow AAbYDD \Rightarrow AAbYdd \Rightarrow AAbMCdd \Rightarrow AAbBCCdd \Rightarrow AAbbCCdd \Rightarrow^* aabbccdd$

Die zweite mögliche Ableitung: $S \Rightarrow AX \Rightarrow AMD \Rightarrow AAMD \Rightarrow AAMDD \Rightarrow AABYDD \Rightarrow^* aaBYDD \Rightarrow^* aabYdd \Rightarrow aabMCdd \Rightarrow aabBCCdd \Rightarrow aabbCCdd \Rightarrow^* aabbccdd$

(d) Der dazugehörige Syntaxbaum ist nicht mehrdeutig, deshalb gibt es nicht mehrere Syntaxbäume. Der Syntaxbaum sieht folgendermaßen aus.



(e) CYK-Algorithmus:

i. $aabbccdd \in \mathcal{L}$

S							
S	S						
	S	X					
		X					
		S,M					
			Y				
A			S,M			D	
A	A	B	B	C	C	D	D
a	a	b	b	c	c	d	d

ii. $abbcc \notin \mathcal{L}$, da man dass bei der i) sehen kann, da ii) ein Teilwort von i) ist

iii. abcdd

S				
S	X			
	X			
	S, M		D	
A	B	C	D	D
a	b	c	d	d

2. (a) Die Grammatik ist:
$\mathcal{G} = (\mathcal{A}, \mathcal{V}, \mathcal{S}, \Pi)$
$\mathcal{V} := \{S, X, Y\}$, $\mathcal{A} := \{a, b, c\}$,
$\Pi := \{$
$\mathcal{S} \to aSXY|abY$
$YX \to XY$
$bX \to bb$
$bY \to bc$
$cY \to cc$
$\}$
Die gegebene Grammatik ist nicht kontextsensitiv, deshalb sollte man die Produktionsregel $YX \to XY$ ersetzen durch:

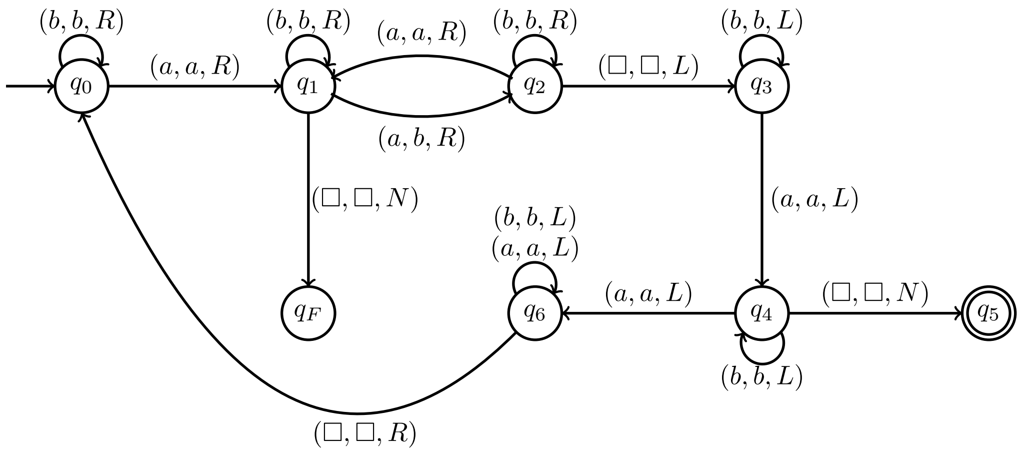
$$\begin{array}{lcl} YX & \to & ZX \\ ZX & \to & ZY \\ ZY & \to & XY \end{array}$$

und desweiteren die Terminale durch die Nichtterminalezeichen ersetzen und die Terminale aus diesen Nichtterminalen ableiten:

$$\begin{array}{lcl} S & \to & ASXY|ABY \\ BX & \to & BB \\ BY & \to & BC \end{array}$$

$$\begin{array}{lcl} CY & \to & CC \\ A & \to & a \\ B & \to & b \\ C & \to & c \end{array}$$

(b) Die linear beschränkte Turing-Maschine:
$M := (\{a\}, \{a, b\}, \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_F, \}, q_5, q_0, \Box, \delta)$, wobei δ in dem Automaten nachvollziehbar ist.



Idee: Die a's solange halbieren, bis entweder ein Rest übrig bleibt oder nur noch ein a da ist (also von $\frac{2}{2} = 1$).

In q_1 ist die Anzahl der a's ungerade, deshalb wird in den Fehlerzustand q_F gewechselt, falls die Eingabe zu Ende ist. In q_2 ist die Anzahl der a's gerade und beim Übergang wird zwischen q_1 und q_2 wird jedes zweite a durch ein b ersetzt also die Anzahl der a's halbiert. Falls die Eingabe in q_2 zu Ende ist, wird die Richtung geändert, d. h. wir wandern auf dem Band rückwarts. In q_4 gibt es nur ein einziges a auf dem Band, falls nun kein weiteres a folgt, sind wir fertig und wechseln in den Endzustand q_5. Ansonsten gehen wir über q_6 an den Anfang des Wortes. Dann wiederholen wir den ganzen Vorgang.

(c) Dieser Aufgabenteil wird den Tutoren überlassen :)

3. (a) Die linear beschränkte Turing-Maschine:
$M_1 := (\{0, 1\}, \{0, 1, \Box\}, \{q_0, q_1, q_2, q_3\}, \{q_3\}, q_0, \Box, \delta)$.

$$\begin{array}{lcl} \delta(q_0, 0) & = & (q_0, 0, R) \\ \delta(q_0, 1) & = & (q_0, 1, R) \\ \delta(q_0, \Box) & = & (q_1, \Box, L) \\ \delta(q_1, 0) & = & (q_2, 1, L) \\ \delta(q_1, 1) & = & (q_1, 0, L) \\ \delta(q_1, \Box) & = & (q_3, 1, N) \end{array}$$

$$
\begin{array}{lcl}
\delta(q_2, 0) & = & (q_2, 0, L) \\
\delta(q_2, 1) & = & (q_2, 1, L) \\
\delta(q_2, \Box) & = & (q_3, \Box, R) \\
\delta(q_3, 0) & = & (q_3, 0, N) \\
\delta(q_3, 1) & = & (q_3, 1, N) \\
\delta(q_3, \Box) & = & (q_3, \Box, N)
\end{array}
$$

(b) $M_2 = (\{0, 1\}, \{0, 1, \Box\}, \{q_0, q_1\}, \{q_1\}, q_0, \Box, \delta_2)$ mit
$\delta_2(q_0, 0) = (q_0, 1, R)$
$\delta_2(q_0, 1) = (q_0, 0, R)$
$\delta_2(q_0, \Box) = (q_1, \Box, N)$

(c) M_3 ist die Komposition von M_1 und M_2, wobei man zuerst M_2 und dann M_1 laufen läßt.

4. (a) $TM = (\mathcal{A}, \mathcal{A} \cup \{\Box\}, \mathcal{Q} \cup \{q_f\}, \{q_f\}, q_0, \Box, \delta_T)$ mit
$\delta_T(q, a) = (\delta(q, a), a, R)$ für alle $q \in \mathcal{Q}$ und $a \in \mathcal{A}$ und
$\delta_T(q, \Box) = (q_f, \Box, R)$ für alle $q \in \mathcal{F}$.

(b) Nein, da endliche Automaten nur Sprachen vom Typ 3 akzeptieren, während Turingmaschinen Sprachen vom Typ 0 akzeptieren, und Typ 3-Spachen eine echte Teilmenge der Typ 0-Sprachen bilden.