GZ Theoretische Informatik (WS16/17)

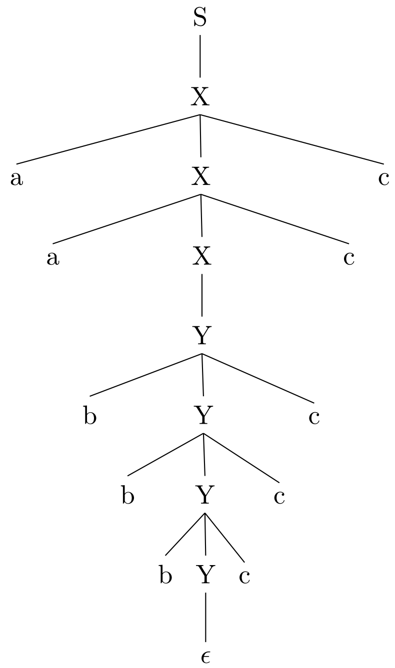
Lösungsvorschlag zu Aufgabenblatt 6

Aufgabe 1

(a) $G = (\{a, b, c\}, \{S, X, Y\}, S, P)$ mit
$P = \{S \to X, S \to \epsilon, X \to aXc, X \to Y, Y \to bYc, Y \to \epsilon\}$

(b) Die Grammatik G fängt damit an, dass sie dem Wort beliebig viele a's und genau so viele c's hinzufügt. Dann fügt sie zwischen die a's und c's entweder ϵ oder beliebig viele b's und noch einmal genau so viele c's ein. So ergibt sich $\#_a(u) + \#_b(u) = \#_c(u)$. Dabei ist außerdem gewährleistet, dass die Reihenfolge der Zeichen im Wort eingehalten wird, da niemals ein b oder c vor einem a und niemals ein c vor einem b stehen kann.

(c) Für das Wort $aabbbccccc$ gilt folgender Ableitungsbaum:



(d) Wir folgen dem Beweisschema aus Vorlesung 9:
Behauptung: $L(G) = \{a^n b^m c^{n+m} | n, m \in N\}$.
Beweis:

⊆: Zunächst zeigen wir, dass $L(G) \subseteq \{a^n b^m c^{n+m} | n, m \in N\}$, also dass jedes Wort, das mit G erzeugt werden kann, auch ein Wort der Sprache L ist.
Dafür zeigen wir, dass folgende Aussagen nach jedem Ableitungsschritt für den abgeleiteten String u gelten:

(i) $\#_a(u) + \#_b(u) = \#_c(u)$
(ii) Kein b oder c kommt vor einem a und kein c kommt vor einem b.

Regeln, die S ersetzen: Die Regeln $S \to X$, $S \to Y$ und $S \to \epsilon$ können nur höchstens einmal zu Beginn der Ableitung angewendet werden, da es keine Regel gibt, die ein S auf der rechten Seite enthält. Sowohl für das Wort X als auch für ϵ sind (i) und (ii) offensichtlich erfüllt.

Regeln, die X ersetzen: Sei der bisher abgeleitete String $w = uXu'$. Da die einzigen Regeln, die ein X produzieren die Regeln $S \to X$ und $X \to aXc$ sind, folgt, dass $u \in \{a\}^*$ und $u' \in \{c\}^*$ und dass $|u| = |u'|$. Insbesondere kann w kein b enthalten. w erfüllt also (i) und (ii) Wir können also entweder $X \to aXc$ oder $X \to Y$ anwenden. Dabei wird das Wort $uaXcu'$, bzw. uYu' abgeleitet. Sowohl (i) als auch (ii) sind hierbei also erfüllt.

Regeln, die Y ersetzen: Sei der bisher abgeleitete String $w = uYu'$. Die einzigen Regeln, die w produziert haben können, sind $X \to Y$ und $Y \to bYc$. Falls w durch $X \to Y$ produziert wurde, gilt, dass w aus $w' = uXu'$ abgeleitet wurde. Da wir oben schon gezeigt haben, dass w' (i) und (ii) erfüllt, muss auch w (i) und (ii) erfüllen. Falls w durch $Y \to bYc$ abgeleitet wurde, muss demnach $u \in \{a^*b^*\}$ und $u' \in \{c^*\}$ und $\#_a(u) + \#_b(u) = \#_c(u)$ gelten. w erfüllt also (i) und (ii).

Wenden wir darauf nun die Regeln $Y \to bYc$ oder $Y \to \epsilon$ an, dann erhalten wir $ubYbu'$, bzw. uu'. Offensichtlich erfüllen beide Wörter (i) und(ii).

Als nächstes zeigen wir, dass $L(G) \supseteq \{a^n b^m c^{n+m} | n, m \in N\}$. Hierzu beweisen wir $\forall n, m \in \mathbb{N} : X \Rightarrow_G^* a^n b^m c^{n+m}$ per Induktion über n.

$\subseteq$: Induktionsanfang: $n = 0$

$X \Rightarrow_G Y \Rightarrow_G^m b^m Y c^m \Rightarrow_G b^m c^m$

Induktionsannahme: $\forall m \in \mathbb{N} : X \Rightarrow_G^* a^n b^m c^{n+m}$ für festes $n \in \mathbb{N}$

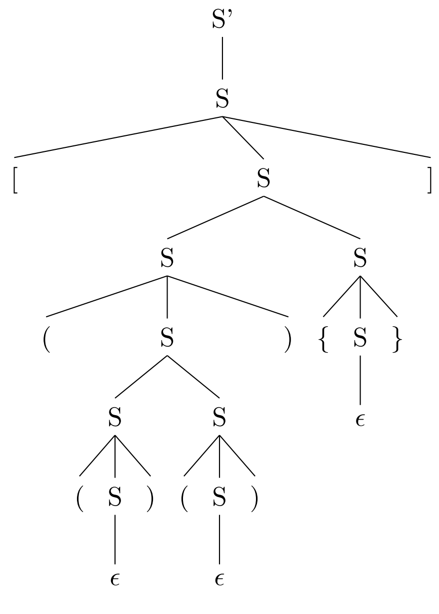
Induktionsschritt: $n \mapsto n + 1$

$X \Rightarrow_G aXc \Rightarrow_G^* aa^n b^m c^{n+m} c = a^{n+1} b^m c^{(n+1)+m}$ mithilfe der *Induktionsannahme*

Insgesamt ergibt sich $\forall n, m \in \mathbb{N} : S \Rightarrow_G X \Rightarrow_G^* a^n b^m c^{n+m}$.

Aufgabe 2

(a) Kontextfrei: Es ist folgendes G mit $L(G) = L_a$:
$G = (\{\{, \}, (,), [,]\}, \{S, S'\}, S', P)$ mit
$P = \{S' \to \epsilon, S' \to S, S \to SS, S \to \{S\}, S \to [S], S \to (S), S \to \epsilon\}$
Das Wort $[(()())\{\}]$ hat folgende Linksableitung:
$\underline{S'} \Rightarrow_G \underline{S} \Rightarrow_G [\underline{S}] \Rightarrow_G [\underline{S}S] \Rightarrow_G [(\underline{S})S] \Rightarrow_G [(\underline{S}S)S] \Rightarrow_G [((\underline{S})S)S] \Rightarrow_G [(()\underline{S})S] \Rightarrow_G [(()(\underline{S}))S] \Rightarrow_G [(()())\underline{S}] \Rightarrow_G [(()())\{\underline{S}\}] = [(()())\{\}]$
und folgenden Ableitungsbaum:



(b) Nicht kontextfrei: Beweis mit Pumping-Lemma für NKA-Sprachen:
Sei $N \in \mathbb{N}$ beliebig. Sei $z = a^N b^{N+1} c^{N+2}$ und die Unterteilung $z = uvwxy$ mit $|vwx| \leq N$ und $|vx| > 0$ beliebig. Fallunterscheidung:

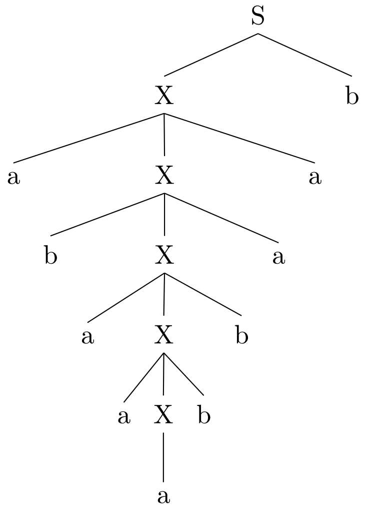
Fall 1: v oder x ist nicht Element $\{a\}^* \cup \{b\}^* \cup \{c\}^*$, also besteht aus verschiedenen Zeichen. Durch aufpumpen mit $i > 1$ entsteht ein Wort, das nicht in der Sprache L_b liegt, da im Wort a's nach b's oder b's nach c's auftauchen.

Fall 2: v enthält nur a's. Dann kann x keine c's enthalten. O.B.d.A nehmen wir an, dass x nur b's enthält. Wir wählen $i = 3$. Dann enthält das Wort $uv^i wx^i y$ $N + (i-1) * |v| = N + 2 * |v|$ viele a's aber weiterhin nur $N + 2 \leq N + 2 * |v|$ c's. Also ist $uv^i wx^i y \notin L_b$.

Fall 3: x enthält nur c's. Dann kann v keine a's enthalten. O.B.d.A nehmen wir an, dass v nur b's enthält. Wir wählen $i = 0$. Dann ist $uv^i wx^i y = uwy = a^N b^{N+1-|v|} c^{N+2-|x|}$. Da $|vx| > 0$, ist entweder $\#_a(uwv) \geq \#_b(uwv)$ oder $\#_b(uwv) \geq \#_c(uwv)$. Es ist also $uwy \notin L_b$.

L_b ist also keine NKA-Sprache.

(c) Kontextfrei: Für folgendes G gilt $L(G) = L_c$:
$G = (\{a, b\}, \{S, X\}, S, P)$ mit
$P = \{S \to Xb, X \to aXa, X \to aXb, X \to bxa, X \to bXb, X \to a\}$
Das Wort *abaaabbaab* hat folgende Linksableitung:
$\underline{S} \Rightarrow_G \underline{X}b \Rightarrow_G a\underline{X}ab \Rightarrow_G ab\underline{X}aab \Rightarrow_G aba\underline{X}baab \Rightarrow_G abaa\underline{X}bbaab \Rightarrow_G abaaabbaab$
und folgenden Ableitungsbaum:



(d) Nicht kontextfrei: Beweis mit Pumping-Lemma für NKA-Sprachen:
Sei $N \in \mathbb{N}$ beliebig. Seien $t = a^N b^N$, $z = tatb = a^N b^N a^{N+1} b^{N+1}$ und die Unterteilung $z = uvwxy$ mit $|vwx| \leq N$ und $|vx| > 0$ beliebig. Fallunterscheidung:

Fall 1: vx ist Teil des ersten t von z. Dann gilt für $i = 2$, dass $uv^iwx^iy = a^N t' b^N a^{N+1} b^{N+1}$, mit $0 < |t'| \leq N$. Das ergibt, dass kein a genau in der Mitte von uv^2wx^2y geben kann. Also ist $uv^2wx^2y \notin L_d$.

Fall 2: vx ist Teil von tb. Dann gilt für $i = 0$, dass $uv^iwx^iy = a^N b^N a^{N+1-m} b^{N+1-n}$, mit $0 < m + n \leq N$. Das ergibt, dass kein a genau in der Mitte von $uvwxy$ geben kann. Also ist $uwy \notin L_d$.

Fall 3: vwx enthält das mittlere a. Dann gilt für $i = 0$,dass $uv^iwx^iy = a^N b^{N-m} a^{N+1-n} b^N b$, mit $0 < m+n \leq N$. Da $m + n > 0$, a^N und b^N können nicht beide zweimal vorkommen. Es ist also $uwy \notin L_d$.

L_d ist also keine NKA-Sprache.

Aufgabe 3

(a) Eine Grammatik erzeugt Terminale durch ihre Ableitungsschritte. Der EA konsumiert sie durch Übergänge. Da jede Produktion einer linkslinearen Grammatik höchstens ein Terminal erzeugt, können wir uns für eine Produktion $(X \to Ya)$ die *Aufleitung* $(Y \to^a X)$ definieren, die wir mit dem Automaten abarbeiten.
Sei die Grammatik $G = (\Sigma, V, S, P)$ gegeben. Den NEA $M = (\Sigma, Q, s \in Q, F \subset Q, \Delta)$, der genau alle Wörter akzeptieren soll, die G produziert, konstruieren wir auf folgende Weise:

$$Q = V \cup \{s'\}$$

$$s = s'$$

$$F = \{S\}$$

Warum entspricht der Enzustand gerade S ? Weil es für das Startsymbol keine weiteren Aufleitungen mehr gibt. Wir konstruieren nun Δ:
Für jede Produktion $(X \to a)$ mit $a \in \Sigma \cup \{\epsilon\}$ fügen wir eine Übergangsregel (s', a, X) hinzu. Warum? Weil wenn man ein Nichtterminal auf a produziert, hat man das finale Wort abgeleitet. Und dieses Wort soll nun vom NEA eingelesen werden. Da der EA aber nur einen Startzustand haben kann, muss man von diesem aus

mit a-Übergängen zu jedem solchen Nichtterminal gelangen können. Für jede Produktion $(X \to Ya)$ fügen wir eine Übergangsregel (Y, a, X) zu Δ hinzu, die die der Produktion entsprechende Aufleitung $(Y \to^a X)$ simuliert. So ermöglichen wir, dass ein Automat, der sich im Zustand Y befindet, beim Einlesen von a in den Zustand X wechselt. Daraus ergibt sich die formale Definition von Δ:
$\Delta = \{(s', a, X) \mid (X \to a) \in P, a \in \Sigma \cup \{\epsilon\}\} \cup \{(Y, a, X) \mid (X \to Ya) \in P; X, Y \in V, a \in \Sigma\}$
Somit akzeptiert M genau die Worte, die G erzeugt.

(b) Zu einer jeden linkslinearen Grammatik

$$G = (\Sigma, V, S, P)$$

gibt es eine rechtslineare Grammatik

$$G' = (\Sigma, V, S, P')$$

mit

$$P' = \{(X \to aY) \mid (X \to Ya) \in P\},$$

die nach Konstruktion die Sprache

$$L' = \{w^R | w \in L(G)\} = (L(G))^R$$

akzeptiert. Da in der Vorlesung gezeigt wurde, dass Sprachen zu rechtslinearen Grammatiken regulär sind, ist L' als solche regulär. $L(G)$ ist nun aber genau $\{w^R \mid w \in L'\} = (L')^R$. Aus der Vorlesung wissen wir, dass wenn L regulär ist, auch L^R regulär ist. Somit ist $(L')^{R^R} = L(G)$ regulär.