technische universität dortmund

Thomas Schwentick

Gaetano Geck, Lutz Oettershagen,
Christopher Spinrath, Marco Wilhelm

Lehrstuhl 1
Logik in der Informatik

Aufgabe 5.1 [Kontextfreie Grammatiken konstruieren] **5 Punkte**

In dieser Aufgabe sollen Sie kontextfreie Grammatiken konstruieren.

a) Gesucht ist eine kontextfreie Grammatik G für die Sprache L, welche genau die Wörter enthält, die mit einer (möglicherweise leeren) Folge von a beginnen, gefolgt von doppelt oder genau halb so vielen b wie a. Zum Beispiel enthält L die Wörter $aabbbb$ und $aaaaaabbb$. **(2 Punkte)**

b) Geben Sie eine kontextfreie Grammatik für das JSON (JavaScript Object Notation) Datenaustauschformat an. Sie finden unter `https://json.org` die Spezifikation des Formats. Wir beschränken uns auf eine Abstraktion, so dass wir Zeichenketten (Strings) durch das Symbol `str` und Zahlen durch das Symbol `num` darstellen.

Somit lautet unser Alphabet

$$\Sigma = \{\texttt{[}, \texttt{]}, \texttt{\{}, \texttt{\}}, \texttt{,}, \texttt{:}, \texttt{str}, \texttt{num}, \texttt{true}, \texttt{false}, \texttt{null}\}.$$

Geben Sie eine kontextfreie Grammatik an, welche genau die gültigen JSON *values* beschreibt. **(3 Punkte)**

Lösung:

a) Die kontextfreie Grammatik G könnte folgendermaßen aussehen:

$$\begin{array}{lcl}
S & \to & A \mid B \\
A & \to & aaAb \mid \varepsilon \\
B & \to & aBbb \mid \varepsilon
\end{array}$$

Die erste Regel entspricht der Auswahl zwischen doppelt oder genau halb so vielen a wie b. Denn mit der zweiten Produktion erzeugen wir das leere Wort oder für jedes b zwei a. Wohingegen wir mit der dritten Produktion das leere Wort oder für jedes a zwei b erzeugen. Dies geschieht jeweils in der benötigten Reihenfolge, d.h. as vor bs.

b) Eine mögliche Lösung ist folgende kontextfreie Grammatik wobei V das Startsymbol ist:

$$\begin{array}{lcl}
V & \to & \texttt{str} \mid \texttt{num} \mid \texttt{true} \mid \texttt{false} \mid \texttt{null} \mid O \mid A \\
P & \to & \texttt{str:}V \mid \texttt{str:}V\texttt{,}P \\
O & \to & \texttt{\{}P\texttt{\}} \mid \texttt{\{\}} \\
A' & \to & V \mid V\texttt{,}A' \\
A & \to & \texttt{[}A'\texttt{]} \mid \texttt{[]}
\end{array}$$

Vom Startsymbol V können wir die verschiedenen JSON values ableiten. Wobei wir das Symbol O für Objekte und A für Arrays nutzen. Objekte bestehen aus geschweiften Klammern sowie darin einer (möglicherweise leeren) Liste von jeweils `str:` und einer Value. Wir nutzen die zweite Produktion, um entweder Objekte mit nur einem `str:`Value oder mehreren zu erzeugen, wobei in diesem Fall die Einträge paarweise durch ein Kommata getrennt werden. Ähnlich gehen wir für Arrays mit den letzten beiden Regeln vor.

Aufgabe 5.2 [Chomsky-Normalform] 7 Punkte

In dieser Aufgabe soll der Algorithmus, der eine kontextfreie Grammatik in die Chomsky-Normalform bringt, angewendet werden. Dabei soll in den folgenden Teilaufgaben einzelne Teilschritte des Algorithmus (CNF1 – CNF5) mit der jeweils angegebenen Grammatik durchgeführt werden.

a) CNF1: Gegeben sei die kontextfreie Grammatik G_0 mit den folgenden Regeln:

$$\begin{array}{lclclcl}
S & \rightarrow & Bb & \mid & Da & \mid & E \\
A & \rightarrow & aAB & \mid & aA & \mid & C \\
B & \rightarrow & bBD & \mid & Bb & \mid & C \\
C & \rightarrow & c & \mid & D & \mid & B \\
D & \rightarrow & a & & & & \\
E & \rightarrow & Ed & & & &
\end{array}$$

Geben Sie die Grammatik nach Entfernen der nicht erzeugenden und nicht erreichbaren Variablen an. Beschreiben Sie Ihr Vorgehen, die Zwischenschritte und geben Sie die Menge V_e und den Erreichbarkeitsgraphen an. **(2 Punkte)**

b) CNF2/CNF3: Gegeben sei die kontextfreie Grammatik G_1 mit den folgenden Regeln:

$$\begin{array}{lclclcl}
S & \rightarrow & BBb & \mid & bc & & \\
A & \rightarrow & B & \mid & Aa & \mid & c \\
B & \rightarrow & BABAa & \mid & Bb & \mid & C \\
C & \rightarrow & c & \mid & A & \mid & B
\end{array}$$

Geben Sie die Grammatiken nach Trennen der Variablen und Terminalsymbole (CNF2) an und verkürzen Sie dann die rechten Seiten (CNF3). Beschreiben Sie Ihr Vorgehen und geben Sie die Zwischenschritte an. **(2 Punkte)**

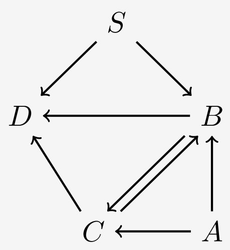
c) CNF4/CNF5: Gegeben sei die kontextfreie Grammatik G_3 mit den folgenden Regeln:

$$\begin{array}{lclclclcl}
S & \rightarrow & W_bA & \mid & BW_c & & & & \\
A & \rightarrow & B & \mid & AW_a & \mid & W_c & & \\
B & \rightarrow & BW_a & \mid & BW_b & \mid & C & & \\
C & \rightarrow & W_c & \mid & A & \mid & B & \mid & \varepsilon \\
W_a & \rightarrow & a & & & & & & \\
W_b & \rightarrow & b & & & & & & \\
W_c & \rightarrow & c & & & & & &
\end{array}$$

Geben Sie die Menge V' an und entfernen Sie die ε-Regeln (CNF4). Entfernen Sie anschließend die Einheits-Regeln (CNF5). Geben Sie die Zwischenschritte und die Menge U an. **(3 Punkte)**

Lösung:

a) Es werden zuerst die nicht erzeugenden Variablen entfernt und für die resultierende kontextfreie Grammatik der Erreichbarkeitsgraph konstruiert. Es ist $V_e = \{S, A, B, C, D\}$. Die Variable E ist nicht erzeugend und wird deshalb weggelassen. Erreichbarkeitsgraph:



Symbol A kann vom Startsymbol aus nicht erreicht werden. Die kontextfreie Grammatik nach Entfernen der nicht erzeugenden und nicht erreichbaren Variablen (CNF1):

$$\begin{array}{lclllll}
S & \rightarrow & Bb & \mid & Da & & \\
B & \rightarrow & bBD & \mid & Bb & \mid & C \\
C & \rightarrow & c & \mid & D & \mid & B \\
D & \rightarrow & a & & & &
\end{array}$$

b) Wir ersetzen die Terminalsymbole in rechten Regelseiten durch neue Variablen, fügen die benötigten Regeln für die Terminalsymbole ein und erhalten damit die folgende kontextfreie Grammatik (CNF2):

$$\begin{array}{lclllll}
S & \rightarrow & BBW_b & \mid & W_bW_c & & \\
A & \rightarrow & B & \mid & AW_a & \mid & W_c \\
B & \rightarrow & BABAW_a & \mid & BW_b & \mid & C \\
C & \rightarrow & W_c & \mid & A & \mid & B \\
W_a & \rightarrow & a & & & & \\
W_b & \rightarrow & b & & & & \\
W_c & \rightarrow & c & & & &
\end{array}$$

Anschließend verkürzen wir die rechten Seiten mit Algorithmus CNF3. Die kontextfreie Grammatik nach Verkürzen der rechten Seiten:

$$\begin{array}{lclllll}
S & \rightarrow & BS_1 & \mid & W_bW_c & & \\
S_1 & \rightarrow & BW_b & & & & \\
A & \rightarrow & B & \mid & AW_a & \mid & W_c \\
B & \rightarrow & BB_1 & \mid & BW_b & \mid & C \\
B_1 & \rightarrow & AB_2 & & & & \\
B_2 & \rightarrow & BB_3 & & & & \\
B_3 & \rightarrow & AW_a & & & & \\
C & \rightarrow & W_c & \mid & A & \mid & B \\
W_a & \rightarrow & a & & & & \\
W_b & \rightarrow & b & & & & \\
W_c & \rightarrow & c & & & &
\end{array}$$

c) Beim Anwenden des Algorithmus CNF4 berechnen wir zunächst die Menge $V' = \{A, B, C\}$, welche alle Symbole enthalten, für die es eine Ableitung zum leeren Wort gibt. Wir erhalten die folgende kontextfreie Grammatik nach Entfernen der ε-Regeln.

$$
\begin{array}{lllllllllll}
S & \to & W_bA & \mid & BW_c & \mid & W_c & \mid & W_b & & \\
A & \to & B & \mid & AW_a & \mid & W_c & \mid & W_a & & \\
B & \to & BW_a & \mid & BW_b & \mid & C & \mid & W_a & \mid & W_b \\
C & \to & W_c & \mid & A & \mid & B & & & & \\
W_a & \to & a & & & & & & & & \\
W_b & \to & b & & & & & & & & \\
W_c & \to & c & & & & & & & &
\end{array}
$$

Es ist $U = \{(A,B),(A,C),(A,W_a),(A,W_b),(A,W_c),(B,C),(B,A),(B,W_a),(B,W_b),(B,W_c),(C,A),(C,B),(C,W_a),(C,W_b),(C,W_c),(S,W_b),(S,W_c)\}$. Jetzt entfernen wir die Einheits-Regeln, d.h. Regeln der Form $X \to Y$. Da C nun nicht mehr erreichbar ist, muss es entfernt werden. Wir erhalten folgende kontextfreie Grammatik.

$$
\begin{array}{lllllllllllll}
S & \to & W_bA & \mid & BW_c & \mid & c & \mid & b & & & & \\
A & \to & AW_a & \mid & a & \mid & b & \mid & c & \mid & BW_a & \mid & BW_b \\
B & \to & AW_a & \mid & a & \mid & b & \mid & c & \mid & BW_a & \mid & BW_b \\
W_a & \to & a & & & & & & & & & & \\
W_b & \to & b & & & & & & & & & & \\
W_c & \to & c & & & & & & & & & &
\end{array}
$$

Aufgabe 5.3 [Sprachen und kontextfreie Grammatiken] **3 Punkte**

Sei G die kontextfreie Grammatik mit den Produktionen

$$S \to aSbb \mid abb$$

und $L = \{a^n b^m \mid n, m \in \mathbb{N} \textit{ und } m \geq 2n\}$ gegeben.

a) Zeigen Sie, dass $L(G) \subseteq L$ gilt. Zeigen Sie dazu für ein beliebiges Wort $w \in L(G)$, dass w von der Form $a^n b^m$ mit $m, n \in \mathbb{N}$ und $m \geq 2n$ ist. Nutzen Sie dafür vollständige Induktion über die Länge k einer Ableitung von w. **(2 Punkte)**

b) Gilt $L(G) \supseteq L$? **(1 Punkt)**

Lösung:

a) Es ist zu zeigen, dass jedes Wort $w \in L(G)$ die Form $a^n b^m$ mit $m, n \in \mathbb{N}$ und $m \geq 2n$ hat. Ein Wort ist genau dann in $L(G)$, wenn es vom Startsymbol S abgeleitet werden kann. Wir nutzen vollständige Induktion über die Länge $k \geq 1$ der Ableitung von w, um zu zeigen, dass w die Form $a^n b^m$ mit $m \geq 2n$ hat.

Induktionsanfang: Wir haben nur eine Produktion, welche zur Länge $k = 1$ führen kann, dies ist $S \to abb$ und damit hat w die Form $a^1 b^2$.

Induktionsschritt: Unsere Induktionsvoraussetzung lautet, die Aussage ist wahr für $k - 1 \geq 1$, d.h. ein Wort, welches wir in $k - 1$ Schritten ableiten können ist in L. Wir behaupten, ein Wort, welches sich mit k Schritten ableiten lässt, ist in L. Eine Ableitung der Länge $k > 1$ beginnt mit Produktion $S \to aSbb$. Wir können eine Ableitung der

Länge k aufschreiben als $S \Rightarrow aSbb \Rightarrow^{(k-1)} aubb$. Da S die einzige Variable in der Satzform $aSbb$ ist, muss $S \Rightarrow^{(k-1)} u$ gelten und mit der Induktionsvoraussetzung folgern wir für das Teilwort u, dass dieses von der Form $a^n b^m$ mit $m \geq 2n$ ist. Damit ist w von der Form $a^{n'} b^{m'}$ mit $n' = n + 1$ und $m' = m + 2$ und folglich $m' \geq 2n'$. □

b) Nein, wir geben ein Gegenbeispiel an und zeigen, dass es ein Wort $w \in L$ gibt, welches nicht in $L(G)$ ist, und somit ist $L(G) \not\supseteq L$. Das Wort $w = a^1 b^3$ ist in $\{a^n b^m | n, m \in \mathbb{N} \; und \; m \geq 2n\}$, da $3 \geq 2 \cdot 1$ gilt. Jedoch kann w unmöglich ausgehend vom Startsymbol von S ableitet werden. Da $abb \neq w$ müssen wir die Produktion $S \rightarrow aSbb$ anwenden. Jede weitere Anwendung einer der Produktionen führt zu mindestens zwei a-Symbolen in der Satzform, und damit, weil G kontextfrei ist, notwendigerweise auch in allen Wörtern, die aus der Satzform ableitbar sind. Folglich ist w zwar in L, jedoch nicht in $L(G)$. □