Theoretische Informatik

Prof. Dr. Peter Thiemann
Marius Weidner
Simon Dorer

Universität Freiburg
Institut für Informatik
Sommersemester 2026

Übungsblatt 12 – Lösungen

Abgabe: Dienstag, 21.07.2026, 16:00 Uhr

Hinweis:
Am Mittwoch, den 22.07.2026, findet zur regulären Vorlesungszeit eine Fragestunde zur Klausur und zu den Inhalten der Vorlesung statt. Überlegen Sie sich gerne Fragen hierzu!

Aufgabe 12.1 (CNF-Transformation; 8 Punkte)
Wir betrachten die kontextfreie Grammatik

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

mit

$$\Sigma = \{a, b, c\}, \qquad N = \{S, A, Q, R\}$$

und

$$P = \left\{ \begin{array}{l} S \to ASQ \mid aR, \\ A \to Q \mid b, \\ Q \to \varepsilon \mid R, \\ R \to cA \mid a \end{array} \right\}.$$

Transformieren Sie G mit den Verfahren aus der Vorlesung in eine Grammatik G_{CNF} in Chomsky-Normalform. Geben Sie bei jedem Schritt die resultierende Grammatik als vollständiges 4-Tupel an.

Gehen Sie dabei wie folgt vor:

(a) Sep: Transformieren Sie G in eine separierte Grammatik G_{Sep}.

(b) Bin: Transformieren Sie G_{Sep} in eine Grammatik G_{Bin}, in der alle rechten Regelseiten Länge höchstens zwei haben.

(c) Del: Transformieren Sie G_{Bin} mit dem Verfahren Del in eine Grammatik G_{Del} ohne ε-Regeln. Geben Sie die dabei berechnete Menge Nullable als Zwischenschritt an.

(d) Unit: Transformieren Sie G_{Del} in eine äquivalente Grammatik G_{CNF} ohne Kettenregeln. Geben Sie den Kettenregelgraphen an und erklären Sie kurz, wie Sie mit darin enthaltenen Zyklen umgehen.

Lösung:

(a) SEP: Wir führen für jedes Terminalsymbol ein neues Nichtterminalsymbol mit der entsprechenden Terminalregel ein. Es ergibt sich

$$G_{\text{SEP}} = (\Sigma, N_{\text{SEP}}, P_{\text{SEP}}, S),$$

wobei

$$\Sigma = \{a, b, c\}, \qquad N_{\text{SEP}} = \{S, A, Q, R, Y_a, Y_b, Y_c\}$$

und

$$P_{\text{SEP}} = \left\{ \begin{array}{l} S \to ASQ \mid Y_a R, \\ A \to Q \mid Y_b, \\ Q \to \varepsilon \mid R, \\ R \to Y_c A \mid Y_a, \\ Y_a \to a, \\ Y_b \to b, \\ Y_c \to c \end{array} \right\}.$$

(b) BIN: Nur $S \to ASQ$ hat eine rechte Regelseite der Länge größer als zwei. Wir führen das neue Nichtterminalsymbol M_1 ein und ersetzen diese Regel durch $S \to AM_1$ und $M_1 \to SQ$. Damit erhalten wir

$$G_{\text{BIN}} = (\Sigma, N_{\text{BIN}}, P_{\text{BIN}}, S),$$

wobei

$$\Sigma = \{a, b, c\}, \qquad N_{\text{BIN}} = \{S, A, Q, R, M_1, Y_a, Y_b, Y_c\}$$

und

$$P_{\text{BIN}} = \left\{ \begin{array}{l} S \to AM_1 \mid Y_a R, \\ M_1 \to SQ, \\ A \to Q \mid Y_b, \\ Q \to \varepsilon \mid R, \\ R \to Y_c A \mid Y_a, \\ Y_a \to a, \\ Y_b \to b, \\ Y_c \to c \end{array} \right\}.$$

(c) DEL: Wir führen ein frisches Startsymbol S' und die Regel $S' \to S$ ein. In der so erweiterten Grammatik $\mathcal{G}'$ gilt

$$\text{Nullable}(G') = \{A, Q\}.$$

Zunächst ist Q wegen $Q \to \varepsilon$ nullable; anschließend wird A wegen $A \to Q$ nullable. Weitere Nichtterminalsymbole kommen nicht hinzu.

Aus $S \to AM_1$ ergänzen wir wegen $A \in \text{Nullable}(G')$ die Regel $S \to M_1$. Aus $M_1 \to SQ$ entsteht wegen $Q \in \text{Nullable}(G')$ die Regel $M_1 \to S$. Schließlich entsteht aus $R \to Y_cA$ die Regel $R \to Y_c$. Nach dem Entfernen von $Q \to \varepsilon$ erhalten wir

$$G_{\text{DEL}} = (\Sigma, N_{\text{DEL}}, P_{\text{DEL}}, S'),$$

wobei

$$\Sigma = \{a, b, c\}, \qquad N_{\text{DEL}} = \{S', S, A, Q, R, M_1, Y_a, Y_b, Y_c\}$$

und

$$P_{\text{DEL}} = \left\{ \begin{array}{l} S' \to S, \\ S \to AM_1 \mid M_1 \mid Y_aR, \\ M_1 \to SQ \mid S, \\ A \to Q \mid Y_b, \\ Q \to R, \\ R \to Y_cA \mid Y_a \mid Y_c, \\ Y_a \to a, \\ Y_b \to b, \\ Y_c \to c \end{array} \right\}.$$

Da $S \notin \text{Nullable}(G')$, wird keine Regel $S' \to \varepsilon$ hinzugefügt.

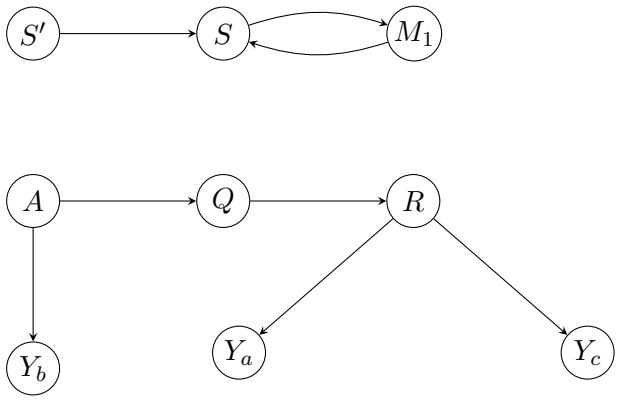
(d) UNIT: Der Kettenregelgraph von G_{DEL} besitzt die Knotenmenge

$$N_{\text{DEL}} = \{S', S, A, Q, R, M_1, Y_a, Y_b, Y_c\}$$

und die Kantenmenge

$$E = \{(S', S), (S, M_1), (M_1, S), (A, Q), (A, Y_b), (Q, R), (R, Y_a), (R, Y_c)\}.$$

Graphisch ergibt sich:



Die Kanten $S \to M_1$ und $M_1 \to S$ bilden einen Zyklus. Wir identifizieren M_1 mit S, ersetzen also überall M_1 durch S, und entfernen die entstehenden Regeln

$S \to S$. Anschließend übernehmen wir entlang aller verbleibenden Kettenregelkanten die jeweils erreichbaren Nicht-Kettenregeln. Eine mögliche resultierende Grammatik ist

$$G_{\text{CNF}} = (\Sigma, N_{\text{CNF}}, P_{\text{CNF}}, S'),$$

wobei

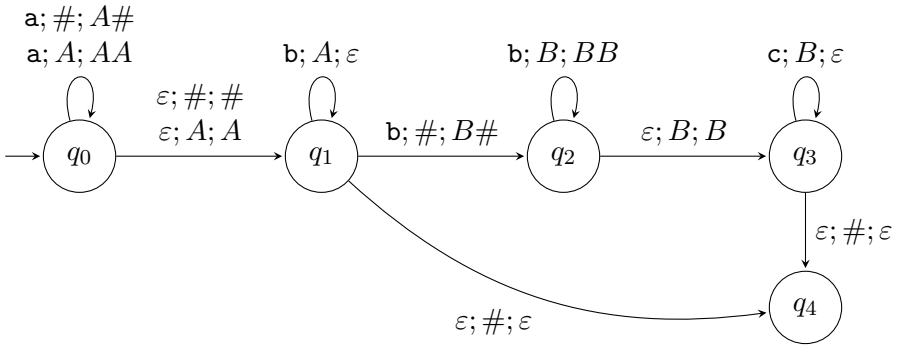
$$\Sigma = \{a, b, c\}, \qquad N_{\text{CNF}} = \{S', S, A, Q, R, Y_a, Y_b, Y_c\}$$

und

$$P_{\text{CNF}} = \left\{ \begin{array}{l} S' \to AS \mid SQ \mid Y_a R, \\ S \to AS \mid SQ \mid Y_a R, \\ A \to Y_c A \mid a \mid b \mid c, \\ Q \to Y_c A \mid a \mid c, \\ R \to Y_c A \mid a \mid c, \\ Y_a \to a, \\ Y_b \to b, \\ Y_c \to c \end{array} \right\}.$$

Alle Regeln haben die Form $X \to x$ oder $X \to YZ$, und das neue Startsymbol S' kommt auf keiner rechten Regelseite vor. Somit ist G_{CNF} in Chomsky-Normalform.

Aufgabe 12.2 (Die Sprache eines Kellerautomaten; 5 Punkte)
Betrachten Sie den Kellerautomaten $\mathcal{K}$ mit dem Kellerbodensymbol #:



(a) Entscheiden Sie für die Wörter `abc` und `aabbbbcc`, ob sie von $\mathcal{K}$ akzeptiert werden. Geben Sie für jedes akzeptierte Wort einen akzeptierenden Lauf an und begründen Sie für jedes nicht akzeptierte Wort kurz, warum kein akzeptierender Lauf existiert.

(b) Erklären Sie kurz, welche Bedeutung die Kellersymbole A und B während eines Laufs haben.

(c) Geben Sie die Sprache $L(\mathcal{K})$ in Mengenschreibweise an.

Lösung:

(a) Das Wort `abc` wird nicht akzeptiert: Beim Lesen von `a` legt der Automat ein A auf den Keller. Das Symbol `b` entfernt dieses A wieder. Danach liegt das Kellerbodensymbol # oben, und für das noch ungelesene Symbol `c` existiert keine Transition. Somit kann kein Lauf die Konfiguration $(q, \varepsilon, \varepsilon)$ erreichen.

Das Wort `aabbbbcc` wird akzeptiert. Ein akzeptierender Lauf ist

$$\begin{aligned}(q_0, \texttt{aabbbbcc}, \#) &\triangleright (q_0, \texttt{abbbbcc}, A\#)\\ &\triangleright (q_0, \texttt{bbbbcc}, AA\#)\\ &\triangleright (q_1, \texttt{bbbbcc}, AA\#)\\ &\triangleright (q_1, \texttt{bbbcc}, A\#)\\ &\triangleright (q_1, \texttt{bbcc}, \#)\\ &\triangleright (q_2, \texttt{bcc}, B\#)\\ &\triangleright (q_2, \texttt{cc}, BB\#)\\ &\triangleright (q_3, \texttt{cc}, BB\#)\\ &\triangleright (q_3, \texttt{c}, B\#)\\ &\triangleright (q_3, \varepsilon, \#)\\ &\triangleright (q_4, \varepsilon, \varepsilon).\end{aligned}$$

(b) Die A-Symbole speichern die Anzahl der gelesenen `a`-Symbole. Für jedes anschließend gelesene `b`-Symbol wird zunächst ein A vom Keller entfernt. Dadurch stellt der Automat sicher, dass mindestens so viele `b`- wie `a`-Symbole gelesen werden.

Die B-Symbole speichern anschließend den Überschuss der gelesenen `b`- gegenüber den gelesenen `a`-Symbolen. Für jedes weitere `b` wird ein B auf den Keller gelegt, das durch genau ein nachfolgendes `c` wieder entfernt werden muss.

(c) Die Zustände erzwingen zunächst, dass jedes Eingabewort die Form $\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k$ besitzt. Von den j Symbolen `b` entfernen genau i die gespeicherten A-Symbole. Die verbleibenden $j - i$ Symbole `b` erzeugen jeweils ein B, und die k Symbole `c` müssen genau diese B-Symbole entfernen. Daher gilt $j - i = k$. Somit ist

$$\boxed{L(\mathcal{K}) = \{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i, j, k \in \mathbb{N} \text{ und } j = i + k\}.}$$

Aufgabe 12.3 (Semantik primitiv-rekursiver Terme; 3.5 Punkte)
Für $k, c \in \mathbb{N}$ sei der konstante Term

$$\mathtt{const}_c^k := \underbrace{\mathcal{S} \circ [\mathcal{S} \circ [\cdots \mathcal{S} \circ [\mathcal{O}^k] \cdots]]}_{c\text{-mal } \mathcal{S}} \in \mathrm{PREC}^k.$$

Betrachten Sie den folgenden Term $\mathtt{t} \in \mathrm{PREC}^2$:

$$\mathtt{t} := \mathrm{PR}\left(\mathtt{const}_1^1, \mathtt{mult} \circ [\pi_1^3, \pi_3^3]\right) \circ [\mathtt{sub} \circ [\pi_2^2, \mathtt{const}_2^2], \mathtt{add} \circ [\pi_1^2, \mathtt{const}_3^2]].$$

Bestimmen Sie die Semantik

$$[\![\mathtt{t}]\!]_2(x, y).$$

Geben Sie das Ergebnis als möglichst einfachen arithmetischen Ausdruck an und begründen Sie Ihre Antwort kurz.

Lösung:

Setze zunächst

$$\mathtt{p} := \mathrm{PR}\left(\mathtt{const}_1^1, \mathtt{mult} \circ [\pi_1^3, \pi_3^3]\right) \in \mathrm{PREC}^2$$

sowie

$$p := [\![\mathtt{p}]\!]_2, \qquad mult := [\![\mathtt{mult}]\!]_2, \qquad sub := [\![\mathtt{sub}]\!]_2.$$

Nach dem Schema der primitiven Rekursion gelten für $n, y \in \mathbb{N}$ die Rekursionsgleichungen

$$p(0, y) = [\![\mathtt{const}_1^1]\!]_1(y) = 1$$

und

$$\begin{aligned} p(n+1, y) &= mult\left([\![\pi_1^3]\!]_3(p(n, y), n, y), [\![\pi_3^3]\!]_3(p(n, y), n, y)\right) \\ &= p(n, y) \cdot y. \end{aligned}$$

Damit gilt

$$p(n, y) = y^n.$$

Die äußere Komposition setzt für das Rekursionsargument

$$n = sub(y, 2) = y \dot{-} 2$$

und für den Parameter das Argument $x + 3$ ein. Folglich ist

$$\boxed{[\![\mathtt{t}]\!]_2(x, y) = (x+3)^{\dot{y-2}}}$$

Aufgabe 12.4 (Semantik eines LOOP-Programms; 3.5 Punkte)

Betrachten Sie das folgende Programm $P \in \mathrm{LOOP}^3$:

```
LOOP x_2 DO
    LOOP x_1 DO
        x_1 := x_1 + 1
    END;
    x_3 := x_3 + 1
END;
LOOP x_3 DO
    x_1 := x_1 + 1
END
```

Bestimmen Sie die Semantik

$$[\![P]\!]_3(x, y, 0).$$

Welche Funktion

$$f = \pi_1^3 \circ [\![P]\!]_3 \circ \mathrm{init}_{2,3}$$

berechnet das Programm? Begründen Sie Ihre Antwort kurz, indem Sie die Wirkung der Zuweisungen und der Schleifen erklären.

Lösung:

Beim Eintritt in die innere Schleife sei der Wert von x_1 gleich r. Ihr Rumpf wird genau r-mal ausgeführt und erhöht x_1 dabei jeweils um 1. Nach der inneren Schleife gilt somit

$$x_1 = r + r = 2r.$$

Ist r_i der Wert von x_1 nach i Durchläufen der äußeren Schleife, so gilt daher

$$r_0 = x \qquad \text{und} \qquad r_{i+1} = 2r_i.$$

Somit ist $r_i = 2^i x$.

Die Anzahl der Durchläufe der äußeren Schleife wird beim Eintritt als y festgelegt. Gleichzeitig wird das anfangs mit 0 initialisierte Arbeitsregister x_3 in jedem Durchlauf um 1 erhöht. Nach y Durchläufen gilt deshalb

$$x_1 = 2^y x \qquad \text{und} \qquad x_2 = y \qquad \text{und} \qquad x_3 = y.$$

Die letzte Schleife wird $x_3 = y$-mal ausgeführt und erhöht x_1 dabei jeweils um 1. Damit ist

$$\boxed{[\![P]\!]_3(x, y, 0) = (2^y x + y, y, y)}.$$

Durch die Projektion auf die erste Komponente ergibt sich schließlich

$$\boxed{f(x, y) = 2^y x + y}.$$

Aufgabe 12.5 (Erfahrungen; Datei: `NOTES.md`)

Notieren Sie Ihre Erfahrungen mit diesem Übungsblatt (benötigter Zeitaufwand, Probleme, Bezug zur Vorlesung, Interessantes, etc.).

Editieren Sie hierzu die Datei `NOTES.md` im Abgabeordner dieses Übungsblattes auf unserer Webplattform. Halten Sie sich an das dort vorgegebene Format, da wir den Zeitbedarf automatisch statistisch auswerten. Die Zeitangabe `4.5 h` steht dabei für 4 Stunden und 30 Minuten.