Prof. Javier Esparza
Philipp Czerner, Kevin Kappelmann, Valentin Krasotin
Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2024 – Hausaufgabenblatt 7

- Die Aufgaben werden in folgender Reihenfolge korrigiert: H7.2, H7.3.
- Die Knobelaufgabe bitte separat auf Moodle abgeben. Sie wird korrigiert.

AT-Aufgabe H7.1. (*VerCYKendes KlapPDAch*) unkorrigiert (2+2+1+1+3+3 Punkte)

Bearbeiten Sie folgende Aufgabe mit Automata Tutor.

Bearbeiten Sie die Hausaufgaben H7.1 (a–d). Bei den *PDA Construction*-Aufgaben darf ihr konstruierter PDA nicht zu viele Zustände oder zu viele Stacksymbole haben (siehe Aufgabenstellung). Wenn Sie einen ε-Übergang angeben wollen, geben Sie statt ε bitte `E` ein (siehe Hinweisbox).

Lösungsskizze.

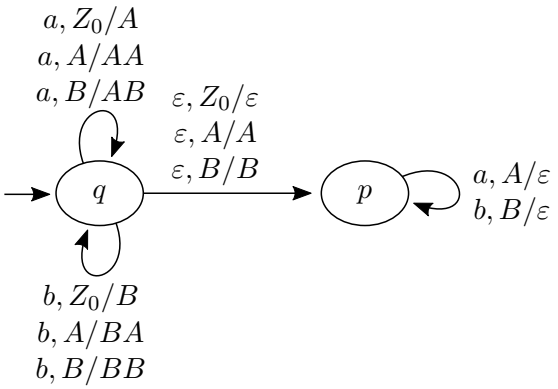
(a)

$_{1,5}$ G, T				
$_{1,4}$ G	$_{2,5}$ T			
$_{1,3}$ S, T	$_{2,4}$ T, G	$_{3,5}$ T		
$_{1,2}$ T	$_{2,3}$ S, G	$_{3,4}$ G	$_{4,5}$ G	
$_{1,1}$ G	$_{2,2}$ S	$_{3,3}$ S, L	$_{4,4}$ S	$_{5,5}$ S
y	t	u	t	t

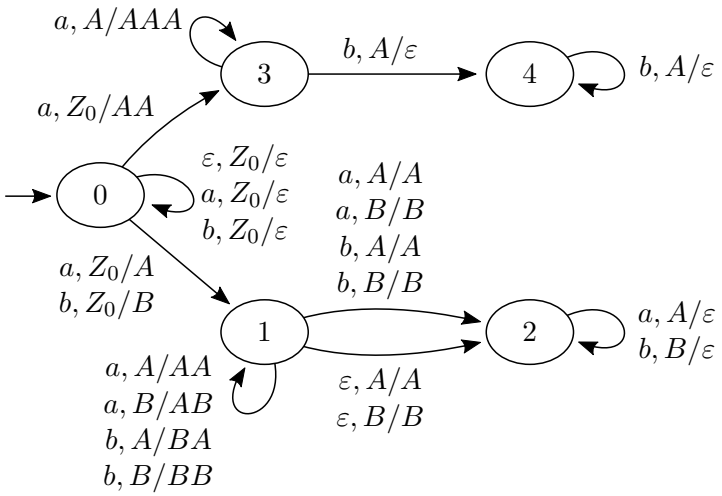
(b)

$_{1,5}$ C, F, L, S				
$_{1,4}$ C, L, S	$_{2,5}$ C, F, L, S			
$_{1,3}$ C	$_{2,4}$ C, L, S	$_{3,5}$ C, L, S		
$_{1,2}$ S, C	$_{2,3}$ C	$_{3,4}$ D, L	$_{4,5}$ C, S	
$_{1,1}$ F, L	$_{2,2}$ C, D, L, S	$_{3,3}$ S	$_{4,4}$ F, L	$_{5,5}$ C, D, L, S
d	e	p	d	e

(c)



(d)



Aufgabe H7.2. (*Zuschnitt*) 7 Punkte

Sei $\Sigma := \{a, b\}$, $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik in CNF und $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA. Wir wollen nun eine kontextfreie Grammatik $G' = (V', \Sigma, P', S')$ für $L(G) \cap L(M)$ erzeugen, und damit beweisen, dass die kontextfreien Sprachen abgeschlossen unter Schnitt mit regulären Sprachen sind.

Dazu verwenden wir Variablen $V' := \{S'\} \cup \{X_{q,r} : X \in V, q, r \in Q\}$. Die Idee ist, dass $X_{q,r}$ genau die Wörter erzeugt, die sowohl von X erzeugt werden können, als auch im DFA von Zustand q nach r gehen. Formal soll also $L_{G'}(X_{q,r}) = \{w \in L_G(X) : \hat{\delta}(q, w) = r\}$ gelten. Zusätzlich ist S' ein besonderes Startsymbol.

Konstruieren Sie G'. Geben Sie also insbesondere die Produktionen P' an.

Hinweis: G' muss nicht in CNF sein.

Lösungsskizze. Wir erzeugen folgende Produktionen:

- $S' \to S_{q_0,r}$ für alle $r \in F$,
- $X_{q,r} \to Y_{q,s} Z_{s,r}$ für alle $(X \to YZ) \in P$ und $q, s, r \in Q$, und
- $X_{q,r} \to c$ für alle $(X \to c) \in P$, falls $\delta(q, c) = r$.

Aufgabe H7.3. (*Ge-Stern*) 1+1+1+2+3 Punkte

Sei $G = (V, \Sigma, P, S)$ eine CFG. Wir betrachten Wörter, die sowohl Terminale als auch

Nichtterminale enthalten können. Für eine Sprache $L \subseteq (V \cup \Sigma)^*$ definieren wir

$$pre^*(L) = \{\alpha \in (V \cup \Sigma)^* \mid \exists \beta \in L : \alpha \to^* \beta\}$$

als die Menge aller Wörter über $V \cup \Sigma$, aus denen man ein Wort in L ableiten kann. Ein polynomieller Algorithmus A ist bekannt[1], der für einen NFA N über dem Alphabet $V \cup \Sigma$ einen NFA für $pre^*(L(N))$ zurückgibt. Insbesondere ist $pre^*(L)$ regulär, falls L regulär ist.

Geben Sie unter Verwendung von A ein Verfahren an, das

(a) für ein gegebenes Wort $w \in \Sigma^*$ entscheidet, ob $w \in L(G)$.

(b) entscheidet, ob $L(G) = \emptyset$.

(c) für ein gegebenes Nichtterminal $X \in V$ entscheidet, ob X erreichbar ist.

(d) für einen gegebenen DFA M entscheidet, ob $L(G) \subseteq L(M)$.

(e) entscheidet, ob $|L(G)| = \infty$.

Lösungsskizze.

(a) $\{w\}$ ist eine reguläre Sprache über $V \cup \Sigma$, d.h. man kann A auf einen NFA für $\{w\}$ anwenden und erhält einen NFA N mit $L(N) = pre^*(\{w\})$. Man überprüft nun, ob $S \in L(N)$.

(b) Σ^* ist eine reguläre Sprache; man kann analog zu (a) überprüfen, ob $S \in pre^*(\Sigma^*)$.

(c) $(V \cup \Sigma)^*\{X\}(V \cup \Sigma)^*$ ist eine reguläre Sprache; man überprüft, ob $S \in pre^*((V \cup \Sigma)^*\{X\}(V \cup \Sigma)^*)$.

(d) Durch Vertauschen der Endzustände und Nicht-Endzustände in M erhält man einen DFA $\overline{M}$ mit $L(\overline{M}) = \overline{L(M)}$. Es gilt $L(G) \subseteq L(M) \Leftrightarrow L(G) \cap L(\overline{M}) = \emptyset \Leftrightarrow S \notin pre^*(L(\overline{M}))$. Man wendet also A auf $\overline{M}$ an und überprüft dann, ob $S \in pre^*(L(\overline{M}))$.

(e) Man überprüft für jedes Nichtterminal $X \in V$, ob X nützlich ist und ob X ein Wort in $(V \cup \Sigma)^*$ erzeugen kann, in dem X, mindestens ein Terminal, und kein weiteres Nichtterminal vorkommt. $|L(G)| = \infty$ gilt genau dann, wenn es ein solches X gibt. X ist genau dann nützlich, wenn $S \in pre^*(\Sigma^*\{X\}\Sigma^*)$ und $X \in pre^*(\Sigma^*)$. Die zweite Eigenschaft ist äquivalent zu $X \in pre^*(\Sigma^+\{X\}\Sigma^* \cup \Sigma^*\{X\}\Sigma^+)$. All das sind reguläre Sprachen über $V \cup \Sigma$, d.h. A ist anwendbar.

Quizaufgabe H7.4. (*Würze ∈ Kürze*) unkorrigiert (5 Punkte)

Der kleine Theodor muss morgen in der Schule einen Grammatiktest schreiben. Eigentlich sollte er lernen, aber er würde viel lieber den Blümchen beim Wachsen zuschauen. Die Grammatik hat er sich auch schon aufgeschrieben, aber leider passt sie nicht auf seinen Spickzettel. Können Sie Ihre moralischen Bedenken überwinden und ihn dabei unterstützen?

[1] Kiefer, S., Kretínský, J., Kucera, A., *Taming the Infinities of Concurrency*, pp. 255-280

Die Grammatik G sei über die folgenden Produktionen gegeben:

$$\begin{array}{ll} S \to SS \mid AD \mid DB \mid T & E \to aABb \mid bBAa \mid EabU \\ A \to aT \mid aaD & T \to S \mid ETb \mid aAU \mid \varepsilon \\ B \to Ub \mid BB & U \to WDW \mid aEb \mid aU \\ C \to aV \mid \varepsilon & W \to aB \mid bAUb \mid bWa \\ D \to Sb \mid b & V \to aSb \mid ab \end{array}$$

(a) Eliminieren Sie alle unnützen Symbole aus G mit den aus der Vorlesung bekannten Verfahren. Geben Sie ihren Rechenweg an.

(b) Leider ist G noch nicht klein genug. Geben Sie eine Grammatik G' mit $L(G') = L(G)$ an, die höchstens zwei Produktionen enthält. Beschreiben Sie ihr Vorgehen.

Lösungsskizze. **(a)** Zuerst berechnen wir die Menge der erzeugenden Symbole. Jede Zeile entspricht hierbei einem Schritt der Induktion aus Satz 4.37.

$$\begin{array}{l} \{a, b\} \\ \cup \{C, D, T, V\} \\ \cup \{A, S\} \\ \cup \emptyset \end{array}$$

Hier ist die Induktion (bzw. der Fixpunktalgorithmus) also abgeschlossen, und wir erhalten folgende Grammatik:

$$\begin{array}{ll} S \to SS \mid AD \mid T & D \to Sb \mid b \\ A \to aT \mid aaD & T \to S \mid \varepsilon \\ C \to aV \mid \varepsilon & V \to aSb \mid ab \end{array}$$

Nun bestimmen wir die Symbole, die erreichbar sind, induktiv nach Satz 4.40.

$$\begin{array}{l} \{S\} \\ \cup \{A, D, T\} \\ \cup \emptyset \end{array}$$

Schließlich ergibt sich die folgende Grammatik.

$$\begin{array}{ll} S \to SS \mid AD \mid T & D \to Sb \mid b \\ A \to aT \mid aaD & T \to S \mid \varepsilon \end{array}$$

(b) Zuerst eliminieren wir T, da offensichtlich T und S äquivalent sind.

$$\begin{array}{ll} S \to SS \mid AD \mid \varepsilon & D \to Sb \mid b \\ A \to aS \mid aaD \mid a \end{array}$$

Die Produktionen von D lassen sich einsetzen.

$$\begin{array}{l} S \to SS \mid ASb \mid Ab \mid \varepsilon \\ A \to aS \mid aaSb \mid aab \mid a \end{array}$$

Ebenso die von A.

$$S \to SS \mid aSSb \mid aaSbSb \mid aabSb \mid aSb \mid aaSbb \mid aabb \mid ab \mid \varepsilon$$

Die Produktionen $S \to aSSb \mid aaSbSb \mid aabSb \mid aaSbb \mid aabb \mid ab$ lassen sich nun entfernen, da man jeweils die rechte Seite auch in folgender Grammatik ableiten kann.

$$S \to SS \mid aSb \mid \varepsilon$$

Diese Grammatik erzeugt genau die balancierten Klammerwörter (mit a als öffnende und b als schließende Klammer), wie aus der Vorlesung (Beispiel 4.10) bekannt. Dafür können wir aber auch eine kleinere Grammatik angeben:

$$S \to aSbS \mid \varepsilon$$

Knobelaufgabe H7.5.

Seien $n \in \mathbb{N}_{>0}$, $\Sigma := \{a, b\}$, $A, B \subseteq \Sigma^n$. Wir schreiben $|x|_A := |\{u : x = uvw, v \in A\}|$, bezeichnen also damit die Anzahl an Vorkommnissen von einem Wort in A in x. Es gilt etwa $|abaaba|_{\{ab,ba\}} = 4$.

Wir wollen herausfinden, ob die Sprache $L' := \{w \in \Sigma^* : |w|_A = |w|_B\}$ regulär ist. Dafür definieren wir $\lambda(wc) := |w|_A - |w|_B$ für $w \in \Sigma^*, c \in \Sigma$. Beachten Sie, dass λ das letzte Zeichen seines Argumentes ignoriert; insbesondere gilt $\lambda(wtz) = \lambda(wt) + \lambda(tz)$ für $w, z \in \Sigma^*$ und $t \in \Sigma^n$. Offensichtlich ist L' genau dann regulär, wenn $L := \lambda^{-1}(0)$ regulär ist.

Zeigen Sie nun, dass L genau dann regulär ist, wenn es keine $u \in A, v \in B$ und $x, y \in \Sigma^*$ gibt, sodass $\lambda(uxu) > 0 > \lambda(vyv)$ gilt.

Lösungsskizze. „⇒“: Wir nehmen an, es gäbe solche u, v, x, y. Es gelten $\lambda(u(xu)^i) = i\lambda(uxu)$ und $\lambda(v(yv)^i) = i\lambda(vyv)$ für $i \in \mathbb{N}$. Da $\lambda(vyv) < 0$, gibt es insbesondere ein k mit $\lambda(uv(yv)^k u) < 0$. Setze nun $r := \lambda(uxu) > 0$, $s := -\lambda(uv(yv)^k u) > 0$, und $w_i := u(xu)^{si}$ für $i \in \mathbb{N}$. Unser Ziel ist jetzt, zu zeigen, dass die L^{w_i} paarweise verschieden sind.

Seien also $i < j$ beliebig. Dann wählen wir $z := (v(yv)^k u)^{ri}$. Es gilt

$$\begin{aligned}
\lambda(w_i z) &= \lambda(u(xu)^{si}(v(yv)^k u)^{ri}) \\
&= \lambda(u(xu)^{si}) + \lambda(u(v(yv)^k u)^{ri}) \\
&= si \cdot \lambda(u(xu)) + ri \cdot \lambda(u(v(yv)^k u)) \\
&= si \cdot r - ri \cdot s = 0
\end{aligned}$$

Also ist $z \in L^{w_i}$. Analog gilt $\lambda(w_j z) = sj \cdot r - ri \cdot s = (j - i)sr \neq 0$ und $z \notin L^{w_j}$. Somit hat L unendlich viele Residualsprachen und ist nicht regulär.

„⇐“: Wir nehmen an, dass die rechte Seite gilt, und wollen schlussfolgern, dass L regulär ist. Wir können oBdA annehmen, dass es kein $u \in A, x \in \Sigma^*$ gibt, sodass $\lambda(uxu) > 0$ gilt. (Ansonsten können wir A und B tauschen.) Wir behaupten nun, dass λ nach oben beschränkt ist.

Angenommen, das Gegenteil sei der Fall. Dann gibt es ein $w \in \Sigma^*$ mit $\lambda(w) > n|A|$, und somit auch $|w|_A > n|A|$. Wir wählen ein solches w mit minimaler Länge. Nach dem Schubfachprinzip existiert ein $u \in A$, das mindestens $(n + 1)$-mal in w vorkommt.

Wir können also w als $w = w'uxuw''$ schreiben, mit $w', w'', x \in \Sigma^*$. Nach Annahme gilt $\lambda(uxu) \leq 0$, dies impliziert

$$\begin{aligned}\lambda(w) &= \lambda(w'uxuw'') \\ &= \lambda(w'u) + \lambda(uxu) + \lambda(uw'') \\ &\geq \lambda(w'u) + \lambda(uw'') \\ &= \lambda(w'uw'')\end{aligned}$$

Also ist die Länge von w nicht minimal, ein Widerspruch.

Unsere Behauptung ist somit gezeigt und es gibt ein $k \in \mathbb{N}$ mit $\lambda(w) \leq k$ für alle $w \in \Sigma^*$. Jetzt zeigen wir, dass für jedes $w \in \Sigma^*$ mit $\lambda(w) < -k$ gilt, dass $L^w = \emptyset$. Zunächst folgern wir $|w| \geq n$ (da $\lambda(w) < -k \leq 0$) und schreiben w als $w = w't$, mit $w' \in \Sigma^*, t \in \Sigma^n$.

Angenommen, es gäbe ein $z \in L^w$. Dann gilt

$$0 = \lambda(wz) = \lambda(w'tz) = \lambda(w't) + \lambda(tz) < \lambda(tz) - k$$

Es folgt $\lambda(tz) > k$, ein Widerspruch. Wir können also einen DFA für L konstruieren, der beim Einlesen λ berechnet, indem er sich die letzten n Zeichen merkt. Der gespeicherte Wert kann nie k überschreiten, und sobald er $-k$ unterschreitet, kann der DFA in einen Fangzustand gehen.