Prof. Javier Esparza
Philipp Czerner, Martin Helfrich

Technische Universität München
Lehrstuhl für Theoretische Informatik

Einführung in die Theoretische Informatik

Sommersemester 2021 – Übungsblatt 7

- Das Übungsblatt ist in zwei Teile gegliedert: den Vorbereitungsteil, den Sie vor der Übung selbstständig bearbeiten sollen, und den Übungs-/Nachbereitungsteil, der Aufgaben enthält, die in der Übung besprochen werden und von Ihnen anschließend zur Nachbereitung verwendet werden können.
- Das ist nicht das Hausaufgabenblatt! Die Hausaufgaben finden Sie auf einem separaten Blatt.

Notation von PDA-Regeln: Anstatt der in den Folien verwendeten Schreibweise $(q, YZ) \in \delta(p, a, X)$ für die Ersetzungsregeln eines PDA kann man alternativ $pX \xrightarrow{a} qYZ$ schreiben wobei $p, q \in Q$, $X \in \Gamma$, $YZ \in \Gamma^*$, $a \in \Sigma \cup \{\varepsilon\}$.

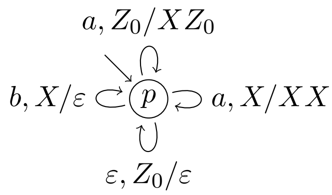
Beispiel: Den PDA mit δ:

$$\delta(p, a, Z_0) = \{(p, XZ_0)\} \qquad \delta(p, a, X) = \{(p, XX)\}$$
$$\delta(p, b, X) = \{(p, \varepsilon)\} \qquad \delta(p, \varepsilon, Z_0) = \{(p, \varepsilon)\}$$

schreibt man alternativ:

$$pZ_0 \xrightarrow{a} pXZ_0 \qquad pX \xrightarrow{a} pXX \qquad pX \xrightarrow{b} p \qquad pZ_0 \xrightarrow{\varepsilon} p$$

oder man stellt diesen als Graph mit Knotenmenge Q dar, wobei die Kante (p, q) dann mit "$a, X/YZ$" beschriftet ist (siehe *Hopcroft et al., Introduction to Automata Theory, Kapitel 6*):



Vorbereitung (→ vor der Übung selbständig zu bearbeiten)

Individualaufgabe Ü7.1. (*Wichtige Begriffe*)

Überprüfen Sie, dass Sie die folgenden Begriffe oder Notationen korrekt definieren können.

- nützlich, erzeugend, erreichbar (Symbole)
- CYK-Algorithmus
- Kellerautomat (PDA)
- Unterschied zwischen $L_\epsilon(A)$ und $L_F(A)$ für einen PDA A
- deterministischer Kellerautomat (DPDA)
- deterministische kontextfreie Sprache (DCFL)

Individualaufgabe Ü7.2. (*Kahoot*)

Falls Sie das Kahoot aus der Vorlesung verpasst haben: Spielen Sie es jetzt! Link[1]

Individualaufgabe Ü7.3. (*Automata Tutor: “CYK & PDAs”*)

Lösen Sie die Aufgaben Ü7.3 (a–f) auf Automata Tutor.[2] **Achtung:** Für die Übungsaufgaben haben Sie beliebig viele Versuche. Für die Aufgaben in Hausaufgabe H7.1 nicht! Bei den *PDA construction* Aufgaben darf ihr konstruierter PDA nicht zu viele Zustände haben (siehe Aufgabenstellung). Wenn Sie einen ε-Übergang angeben wollen, geben Sie statt ε bitte `E` ein (siehe Hinweisbox über Canvas). Die Simulation bei PDAs ist deaktiviert. Bitte wundern Sie sich nicht, dass bei einem Klick auf `Start Simulation` nichts passiert. **Tipp:** Für den Aufgabentyp “CYK” können Sie sich zum Üben weitere Aufgaben von AT generieren lassen. Klicken Sie dafür auf `Home > My Autogenerated Problems` und wählen Sie den Aufgabentyp und gewünschten Schwierigkeitsgrad.

Lösungsskizze.

(a)

1,5 S				
1,4 S	2,5 S			
1,3	2,4 S	3,5		
1,2	2,3 L	3,4	4,5	
1,1 S	2,2 A	3,3 S	4,4 B	5,5 S
x	a	x	b	x

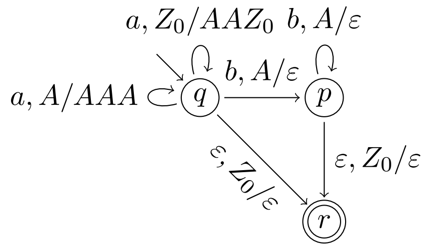
(b)

1,4 O			
1,3 O	2,4 Y		
1,2 O, U	2,3 Y	3,4 S	
1,1 O, K, U	2,2 O, K, U	3,3 S	4,4 S
d	d	v	v

(c) $\epsilon, bb, aababbab \in L$ und $abaab, bbb, aaabaaa \notin L$

(d) $abaababb, abba, ababbaaaba \in L$ und $baa, aab, bbbbb \notin L$

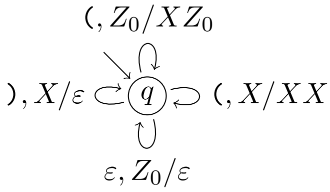
(e)



---

[1] Falls der Link nicht mehr funktioniert, teilen Sie dies bitte der Übungsleitung mit. Die Teilnehmeranzahl ist leider auf 2000 begrenzt.

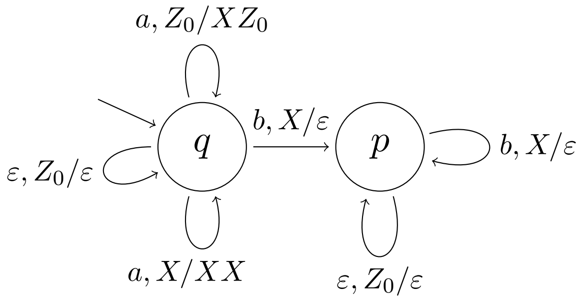
[2] Wenn Sie Automata Tutor noch nicht verwendet haben, folgen Sie erst den Schritten in Ü1.2, um sich richtig zu registrieren.

(f)


Individualaufgabe Ü7.4. (*Mein erster PDA*)

Zeichnen Sie einen PDA, der die Sprache $L = \{a^i b^i \mid i \geq 0\}$ erkennt.

Lösungsskizze. Idee: Für jedes a legen wir ein X auf den Stack und überprüfe dann, ob die die Anzahl von bs mit der Anzahl an X auf dem Stack übereinstimmt.
Ein PDA A der mit leerem Keller die Sprache L akzeptiert (also $L_\epsilon(A) = L$) ist:



Übung und Nachbereitung

Fokusaufgabe Ü7.5. (*PDAs*)

Geben Sie für die folgenden Sprachen jeweils einen Kellerautomaten A_i in einer der oben aufgeführten Darstellungsarten an, so dass $L_i = L(A_i)$. Der Automat soll mit *leerem Stack* akzeptieren. Geben Sie dann zusätzlich für jeden Automaten jeweils ein nicht-leeres Wort w mit akzeptierendem Lauf an.

(a) $L_1 = \{a^n b^{3n} \mid n \geq 0\}$

(b) $L_2 = \{a^n b^m \in \{a,b\}^* \mid n \leq m \leq 2n\}$

(c) $L_3 = \{w \in \{a,b\}^* \mid 2 \cdot |w|_a = 3 \cdot |w|_b\}$

Übungsaufgabe Ü7.6. (*CFG bereinigen*)

Die CFG G bestehe aus folgenden Produktionen über dem Alphabet $\Sigma = \{a, b\}$:

$$\begin{array}{lcl} S & \to & ASA \mid aB \\ A & \to & B \mid S \mid CB \\ B & \to & b \mid \varepsilon \\ C & \to & aC \\ D & \to & aSCb \mid a \end{array}$$

(a) Beschreiben Sie in eigenen Worten, wann ein Nichtterminal *nützlich* in einer Grammatik ist.

(b) Reduzieren Sie die Grammatik G auf die nützlichen Nichtterminale.

Lösungsskizze.

(a) Ein Nichtterminal ist nützlich, wenn es eine Ableitung vom Startsymbol zu einem Wort (also einer Folge von Terminalen) gibt, in der das Nichtterminal verwendet wird. Nützliche Symbole sind stets erzeugend und erreichbar. Erzeugend bedeutet, dass von dem Nichtterminal ausgehend ein Wort produziert werden kann; erreichbar bedeutet, dass vom Startsymbol ausgehend das Nichtterminal produziert werden kann. Der Begriff ist relevant, da ähnlich wie bei Automaten angenommen wird, dass alle Zustände erreichbar sind, man bei Grammatiken annehmen möchte, dass Sie nur nützliche Nichtterminale enthalten. Deswegen benötigt man eine klare Definition und eine Möglichkeit, eine Grammatik, die die Anforderung nicht erfüllt, in eine mit nur nützlichen Nichtterminalen zu überführen.

(b) *Erzeugende Variablen*:

- Intuition: C kann kein Wort produzieren, da eine Ableitung, die ein C enthält, niemals "terminiert", also nie nur Terminale enthält.
- Formal bauen wir iterativ die Menge aller Nichtterminale, die eine Möglichkeit haben, ein Wort aus nur Terminalen zu erzeugen.
- $P_0 := \{X \mid \exists (X, \Gamma) \in P.\ \Gamma \in \Sigma^*\}$
- $P_{k+1} := P_k \cup \{X \in V \mid \exists (X, \Gamma) \in P.\ \forall Y \in V.\ (|\Gamma|_Y > 0 \to Y \in P_k)\}$ bis $P_{k+1} = P_k$.
- Führt auf:

  $$P_0 = \{B, D\} \qquad P_1 = \{B, D, A, S\} = P_2$$

  Damit kann C samt $C \to aC$, $A \to CB$ und $D \to aSCb$ entfernt werden.

Erreichbare Variablen:

- Intuition: D kann von S aus nicht erreicht werden.
- Formal bauen wir die Menge aller Nichtterminale, die von S aus erzeugt werden können.
- $R_0 := \{S\}$
- $R_{k+1} := R_k \cup \{Y \in V \mid \exists (X, \Gamma) \in P.\ X \in R_k \wedge |\Gamma|_Y > 0\}$ bis $R_{k+1} = R_k$.
- Führt auf:

  $$R_0 = \{S\} \qquad R_1 = \{S, A, B\} = R_2$$

  Damit kann D und $D \to a$ entfernt werden.

Die reduzierte (bereinigte) Grammatik ist also:

$$S \to ASA \mid aB \qquad A \to B \mid S \qquad B \to b \mid \varepsilon$$

Hinweis: Diese Grammatik haben wir in Aufgabe Ü6.6 in CNF überführt.

Übungsaufgabe Ü7.7. (*CYK*)

(a) Beschreiben Sie in eigenen Worten, wie die Indizes in einer CYK-Tabelle zu verstehen sind.

(b) Beschreiben Sie in eigenen Worten, wie man den Inhalt eines Feldes in der CYK-Tabelle berechnet.

(c) Wir betrachten die Grammatik $G = (\{S, T, U, A, B, C\}, \{a, b, c\}, P, S)$ in CNF mit den folgenden Produktionen P:

$$\begin{aligned} S &\to TS \mid CT \mid a & \qquad A &\to a \\ T &\to AU \mid TT \mid c & \qquad B &\to b \\ U &\to SB \mid AB & \qquad C &\to c \end{aligned}$$

Bestimmen Sie mit dem CYK-Algorithmus, ob $ccaab \in L(G)$ und $aabcc \in L(G)$. Geben Sie dabei auch die berechneten Tabellen an.

Lösungsskizze.

(a) Das Feld $F_{x,y}$ beinhaltet alle Nichtterminale, die das Teilwort vom x-ten Terminal bis zum y-ten Terminal erzeugen können. (Wir fangen hier mit 1 an zu zählen.) Beispiel: Wenn das Wort $w = abaaab$ ist, dann beinhaltet $F_{2,4}$ die Terminale, die baa erzeugen.

(b) Für Felder $F_{x,x}$ (in der untersten Zeile) sucht man nach allen Nichtterminalen X, die eine Produktion $X \to a$ haben wobei a das x-te Zeichen des betrachteten Wortes ist.
Für Felder $F_{x,y}$ betrachtet man alle Kombinationen aus Feldern $F_{x,x} \times F_{x+1,y}$, $F_{x,x+1} \times F_{x+2,y}$ und so weiter.[3] Wenn (A, B) in einer der Mengen vorkommt und es eine Produktion $X \to AB$ gibt, dann fügt man X zum Feld $F_{x,y}$ hinzu.

(c) Nach dem CYK-Algorithmus ergeben sich folgende Berechnungstabellen:

1,5 S, T				
1,4	2,5 S, T			
1,3 S	2,4	3,5 T		
1,2 S, T	2,3 S	3,4	4,5 U	
1,1 C, T	2,2 C, T	3,3 S, A	4,4 S, A	5,5 B
c	c	a	a	b

1,5 S, T				
1,4 T	2,5			
1,3 T	2,4	3,5		
1,2	2,3 U	3,4	4,5 S, T	
1,1 S, A	2,2 S, A	3,3 B	4,4 C, T	5,5 C, T
a	a	b	c	c

[3] Tipp: Man legt seinen linken Zeigefinger auf das Feld $F_{x,x}$ und seinen rechten Zeigefinger auf $F_{x+1,y}$. Der linke Finger geht immer ein Feld höher, der rechte immer ein Feld nach unten rechts.

Also ist $ccaab \in L(G)$ und $aabcc \in L(G)$.

Übungsaufgabe Ü7.8. (*PDA Einschränkungen*)

(a) Wir beschränken die Größe des Kelleralphabets Γ von PDAs und zeigen, dass jede kontextfreie Sprache von einem PDA mit $|\Gamma| = 2$ erkannt werden kann. Skizzieren Sie hierzu eine allgemeine Übersetzung von einem PDA mit $|\Gamma| > 2$ zu einem PDA mit $|\Gamma'| = 2$, so dass beide Automaten die gleiche Sprache erkennen.

(b) Wir beschränken die Kellerhöhe von PDAs auf maximal k Kellerzeichen und nennen diese PDAs *k-bounded-Stack- PDA*. Insbesonders kann ein solcher PDA keine PUSH-Operationen ausführen sollten danach mehr als k Symbole auf dem Stack liegen. Zeigen Sie, dass k-bounded-Stack- PDA genau die regulären Sprachen erkennen, indem Sie eine allgemeine Übersetzung von PDAs zu $\varepsilon-$ NFA angeben.

Lösungsskizze.

(a) Wir encodieren binär die Stacksymbole Γ mit Hilfe von Γ'. Sei k eine passende Konstante, so dass $|\Gamma| \leq 2^k$. Der neue PDA liest immer genau k Zeichen über eine Reihe von Hilfszuständen und rekonstruiert somit $X \in \Gamma$ und führt dann die passende Transition aus. Hierauf werden wieder mit einer Reihe von Hilfszuständen die neuen Stacksymbole auf den Stack gepusht.

(b) Da der Stack beschränkt ist, kann der $\varepsilon-$ NFA den Stack in den Zuständen encodieren: $Q' = Q \times \bigcup_{i=0}^{k} \Gamma^i$ und dementsprechend die Transitionen simulieren. Falls wir $L_\varepsilon(M)$ betrachten, definieren wir $F' = \{(q, \varepsilon) \mid q \in Q\}$. Falls wir $L_F(M)$ betrachten, definieren wir $F' = \{(q, \alpha) \in Q' \mid q \in F\}$.