Prof. Dr. Titus Dose
Fakultät für Informatik und Mathematik
OTH Regensburg

Theoretische Informatik
Wintersemester 2024/2025
08.01.2025

Übungsblatt 12

(Besprechung am 16.01.2025)

1. **Pumping-Lemma für kontextfreie Sprachen**
   Beweisen Sie, dass folgende Sprache über dem Alphabet $\{a, b, c\}$ nicht kontextfrei ist.
   $$L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$$

2. **kontextfrei = regulär bei unären Alphabeten**
   Ziel dieser Aufgabe ist es, folgende Aussage zu beweisen: jede kontextfreie Sprache $L \subseteq \{a\}^*$ ist regulär.

   Im Folgenden sei $k \in \mathbb{N}$ eine 2-pumping number für L. Beweisen Sie:

   (a) Für $m \geq k$ gilt
   $$a^m \in L \implies \forall_{i \in \mathbb{N}}\, a^{m+i \cdot k!} \in L.$$

   (b) Benutzen Sie (a), um die Existenz von Zahlen $m_1, \ldots, m_s$ mit $s \leq k!$ zu zeigen, so dass gilt
   $$L = \{x \in L \mid |x| < n\} \cup \bigcup_{r=1}^{s} \{a^{m_r + i \cdot k!} \mid i \in \mathbb{N}\}.$$

   (c) Folgern Sie $L \in \mathrm{REG}$.

3. **Kontextsensitive und nichtverkürzende Grammatiken**
   Sei $\Sigma = \{a\}$ und $L = \{a^n | n = 2^k \text{ für ein } k \in \mathbb{N}\} \subseteq \Sigma^*$.

   (a) Geben Sie eine nichtverkürzende Grammatik für L an und begründen Sie, dass die Grammatik die angegebene Sprache erzeugt.

   (b) Wandeln Sie die konstruierte Grammatik mit dem Verfahren aus Theorem 4.31 in eine äquivalente kontextsensitive Grammatik um.

4. **Ein falscher Beweis**
   Der Student Arno Nühm möchte beweisen, dass die kontextsensitiven Sprachen unter Konkatenation abgeschlossen sind. Er hat dazu folgenden Beweis verfasst.

   Seien L_1 und L_2 kontextsensitive Sprachen über einem Alphabet Σ. Dann gibt es kontextsensitive Grammatiken $G_1 = (\Sigma, N_1, S_1, R_1)$ und $G_2 = (\Sigma, N_2, S_2, R_2)$ mit $L(G_1) = L_1$ und $L(G_2) = L_2$. Wir dürfen davon ausgehen, dass $N_1 \cap N_2 = \emptyset$ (ansonsten benennen wir die Nichtterminale von G_2 um, sodass keine Nichtterminale mehr vorliegen, die auch in N_1 sind). Wir definieren die Grammatik $G = (\Sigma, N_1 \cup N_2 \cup \{S\}, S, R_1 \cup R_2 \cup \{S \to S_1 S_2\})$, wobei S ein Nichtterminal ist, welches nicht in $N_1 \cup N_2$ ist.

   Die Grammatik G ist kontextsensitiv. Weiter gilt offenbar $L(G) = L(G_1) \cdot L(G_2)$. $\square$

   (a) Streichen Sie alle falschen Aussagen in Arnos Beweis an.

   (b) Konstruieren Sie Beispielgrammatiken G_1 und G_2, für welche Arnos Beweis fehlschlägt.

   (c) Arnos Beweisidee würde für kontextfreie Sprachen funktionieren. Beschreiben Sie, wie sich aus zwei kontextfreien Grammatiken G und G' kontextfreie Grammatiken für die Sprachen $L(G) \cup L(G')$, $L(G) \cdot L(G')$ und $L(G)^*$ bauen lassen.

Hints

Exercise 1:
Kein Hinweis.

Exercise 2:
Die Aufteilung in Teilaufgaben ist Hinweis genug.

Exercise 3:
Sie können sich an der in den Übungen besprochenen “ähnlichen” Aufgabe orientieren.
Offensichtlich erhält man die Sprache, indem man mit a beginnt und dann die Anzahl der Symbole jeweils verdoppelt. Eine solche Verdopplung kann z.B. dadurch geschehen, dass man ein spezielles Nichtterminal über das Wort laufen lässt und dabei jedes a durch aa ersetzt. Nach einem Durchlauf muss dieses Nichtterminal dann beseitigt werden. Außerdem müssen Sie sicherstellen, dass Verdopplungen immer komplett durchgeführt werden und nicht in der Wortmitte stoppen.

Exercise 4:
Kein Hinweis.

Extra tasks

1. **More closure properties**
   Let Σ be an alphabet and $L \subseteq \Sigma^*$. We define

   $$\frac{1}{2}L \overset{\text{df}}{=} \{w \in \Sigma^* \mid \exists_{x \in \Sigma^*} |x| = |w| \wedge wx \in L\}.$$

   We want to see: if $L \in \text{FA}$, then $\frac{1}{2}L \in \text{FA}$.

   (a) Explain how a finite automaton A for L can be transformed into a finite automaton A' for $\frac{1}{2}L$.

   (b) Define A' formally as a quintuple.

Solutions

Solution for exercise 1:
Es ist hinreichend zu zeigen, dass L nicht 2-pumpable ist.
Zu zeigen: Für jedes $k \in \mathbb{N}^+$ existiert ein Wort $w \in L$ mit $|w| \geq k$, sodass für jede Zerlegung $w = rstuv$ mit $|stu| \leq k$ und $su \neq \varepsilon$ ein $i \in \mathbb{N}$ mit $rs^i tu^i v \notin L$ existiert.
Sei $k \in \mathbb{N}^+$ beliebig.
Wähle $w = a^k b^k c^k$.
Sei eine beliebige Zerlegung $w = rstuv$ mit $|stu| \leq k$ und $su \neq \varepsilon$ gegeben.
Wegen $|stu| \leq k$ kann es nicht sein, dass in stu sowohl mindestens ein a als auch mindestens ein c vorkommt. Folglich ist die Anzahl an a's oder die Anzahl an c's in $rs^0 tu^0 v = rtv$ genau k. Wegen $su \neq \varepsilon$ hat rtv aber weniger Buchstaben als $rstuv$; es gibt also einen Buchstaben, der in rtv weniger als k mal vorkommt, weswegen in rtv die Anzahlen an a's, b's und c's nicht gleich sind, also $rtv \notin L$.

Solution for exercise 2:
Dies ist nur eine Skizze (Rückfragen gerne im Diskussionsforum):

(a) Dies folgt aus dem Pumping-Lemma: Sei $w \in L^{\geq k}$. Da k eine 2-pumping number ist und $|w| \geq k$, gibt es eine Zerlegung $w = rstuv$ mit $|stu| \leq k$ und $su \neq \varepsilon$, sodass für alle $i \in \mathbb{N}$ das Wort $rs^i tu^i v$ in L ist. Das Wort $rs^i tu^i v$ lässt sich schreiben als $a^{|w|+(i-1)\cdot(|s|+|u|)}$. Wegen $|s| + |u| \leq |stu| \leq k$ ist $|s| + |u|$ ein Teiler von $k!$.
Wählt man nun $i = \iota \cdot \frac{k!}{|s|+|u|} + 1$ für ein beliebiges $\iota \in \mathbb{N}$, so erhalten wir mit Obigem, dass $a^{|w|+\iota \cdot k!} \in L$, was zu zeigen war.

(b) Für jedes $i \in \{1, \ldots, k!\}$ wähle $\alpha_i = \min(\{|w| \mid w \in L^{\geq k}, |w| \equiv i \mod k!\})$, wobei wir $\min(\emptyset)$ als -1 definieren.
Dann wählen wir $m_1, \ldots, m_s$ als die positiven Zahlen aus $\{\alpha_1, \ldots, \alpha_{k!}\}$.
Dann gilt $\subseteq$ in der zu zeigenden Gleichung nach Wahl der $m_1, \ldots, m_s$ und $\supseteq$ nach (a) und Wahl der $m_1, \ldots, m_s$.

(c) L ist eine Vereinigung von $s + 1$ regulären Mengen (die erstgenannte Menge ist endlich und daher regulär, die anderen Mengen lassen sich durch reguläre Ausdrück $a^{m_r} + (a^{k!})^*$ beschreiben) und somit selbst regulär.

Solution for exercise 3:
Wir konstruieren zunächst eine nichtverkürzende Grammatik.

Terminale: a

Nichtterminale:
S $\widehat{=}$ Startsymbol, erzeugt mehrere Symbole V
V $\widehat{=}$ Verdoppler, laufen von links nach rechts über das bisher erzeugte Wort und verdoppeln jedes a
E $\widehat{=}$ Endsymbol, markiert uns das rechte Ende des Wortes; wird ganz am Ende in a umgewandelt, deshalb wird E wie ein a behandelt (bei Verdopplung)

nichtverkürzende Grammatik:

$$\left.\begin{array}{l}\text{S} \to \text{VS}\\ \text{S} \to \text{E}\end{array}\right\}\ \text{Phase 1}$$

$$\left.\begin{array}{l}\text{VE} \to \text{aE}\\ \text{Va} \to \text{aaV}\end{array}\right\}\ \text{Phase 2}$$

$$\left.\text{E} \to \text{a}\right\}\ \text{Phase 3}$$

Phase 1:
- der Befehl S → VS erzeugt Wörter der Form VV... VS
- wird das erste Mal der Befehl S → E ausgeführt, so können keine weiteren Befehle aus Phase 1 angewendet werden
- am Ende von Phase 1 haben wir ein Wort VV... VE

Phase 2:
- wird eventuell gar nicht ausgeführt, sondern gleich Phase 3
- zuerst wird VE → aE ausgeführt und damit das E (das ja ein gedachtes a ist) verdoppelt
- jetzt kann auch Va → aaV angewendet werden
- durch Va → aaV werden jeweils alle a's verdoppelt und wenn das V rechts angekommen ist, wird es durch VE → aE gelöscht

Phase 3:
- E → a kann angewendet werden, wenn alle V gelöscht sind (danach können nämlich keine V mehr gelöscht werden)

Wir wandeln diese Grammatik nun in eine vom Typ 1 um.
1. Schritt: Ersetzen Terminale in allen Regeln, d.h. wir ersetzen

- VE → aE durch VE → AE
- Va → aaV durch VA → AAV
- E → a durch E → A

und fügen die Regel A → a hinzu.
2. Schritt: Dazu ersetzen wir die einzige nicht-kontextsensitive Regel VA → AAV durch

$\mathrm{VA} \to \mathrm{D_1A}$	D_1 ist ein neues Nichtterminal
$\mathrm{D_1A} \to \mathrm{D_1D_2}$	D_2 ist ein neues Nichtterminal
$\mathrm{D_1D_2} \to \mathrm{AD_2}$	
$\mathrm{AD_2} \to \mathrm{AAV}$	

Kontextsensitive Grammatik:
$G = (\Sigma, N, S, R)$ mit

$$\Sigma = \{\mathrm{a}\}$$
$$N = \{\mathrm{S}, \mathrm{V}, \mathrm{E}, \mathrm{A}, \mathrm{D_1}, \mathrm{D_2}\}$$
$$R = \{\mathrm{S} \to \mathrm{VS}, \mathrm{S} \to \mathrm{E}, \mathrm{VE} \to \mathrm{AE}, \mathrm{VA} \to \mathrm{D_1A}, \mathrm{D_1A} \to \mathrm{D_1D_2}, \mathrm{D_1D_2} \to \mathrm{AD_2}, \mathrm{AD_2} \to \mathrm{AAV}, \mathrm{E} \to \mathrm{A}, \mathrm{A} \to \mathrm{a}\}$$

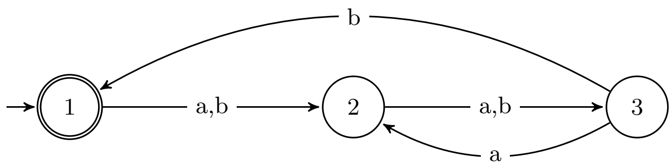
Solution for exercise 4:

1. Der einzige Fehler ist im letzten Satz des Beweises. Für die von Arno konstruierte Grammatik gilt zwar $L(G) \supseteq L(G_1) \cdot L(G_2)$, aber es kann auch $L(G) \not\subseteq L(G_1) \cdot L(G_2)$ gelten, s.u.
2. Betrachte die Grammatiken $G_1 = (\{a, b\}, \{S_1\}, S_1, \{S_1 \to a\})$ und $G_2 = (\{a, b\}, \{S_2\}, S_2, \{aS_2 \to ab\})$. Beide Grammatiken sind kontextsensitiv und wegen $L(G_2) = \emptyset$ ist auch $L(G_1) \cdot L(G_2) = \emptyset$. Jedoch lässt sich mit der Grammatik G wie folgt das Wort ab erzeugen:
$$S \Rightarrow S_1S_2 \Rightarrow aS_2 \Rightarrow ab.$$
3. Seien kontextfreie $G_1 = (\Sigma, N_1, S_1, R_1)$ und $G_2 = (\Sigma, N_2, S_2, R_2)$ gegeben. Wir gehen im Folgenden davon aus, dass $N_1 \cap N_2 = \emptyset$ und $S \notin N_1 \cup N_2$ (sonst Nichtterminale umbenennen).
Vereinigung: Grammatik für $L(G_1) \cup L(G_2)$: $(\Sigma, N_1 \cup N_2 \cup \{S\}, S, R_1 \cup R_2 \cup \{S \to S_1, S \to S_2\})$
Konkatenation: Grammatik für $L(G_1) \cup L(G_2)$: $(\Sigma, N_1 \cup N_2 \cup \{S\}, S, R_1 \cup R_2 \cup \{S \to S_1S_2\})$
Iteration: Grammatik für $L(G_1)^*$: $(\Sigma, N_1 \cup \{S\}, S, R_1 \cup \{S \to SS_1, S \to S_1\})$

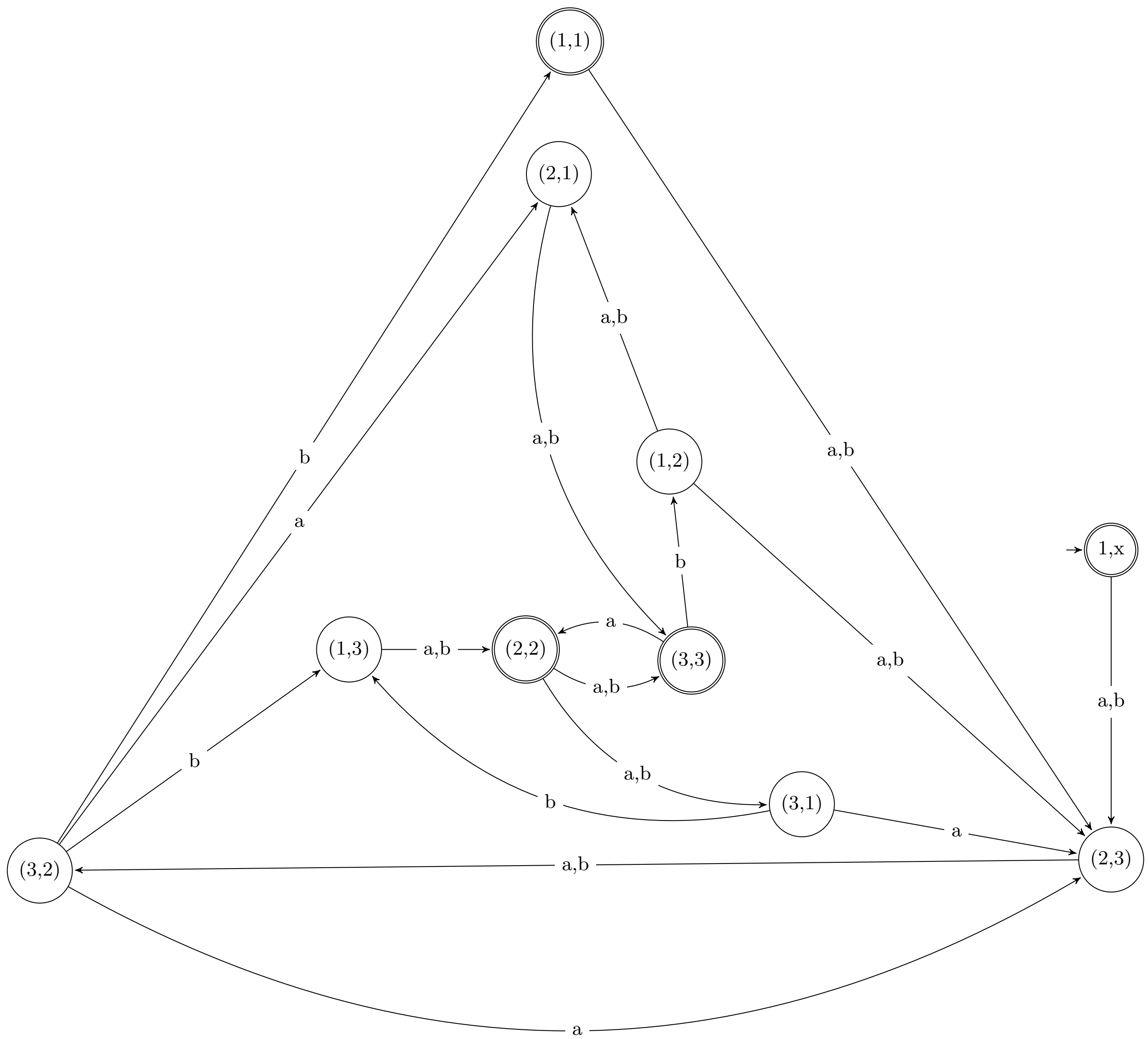
Solution for extra task 1:
We sketch three ways of proving $\frac{1}{2}L$ to be in FA. Let $A = (\Sigma, S, \delta, s_0, F)$ be a DFA with $L(A) = L$.

1. Consider the NFA $A' = (\Sigma, S', \delta', (s_0, x), F')$ with $S' = S \times (S \cup \{x\})$ (for some $x \notin S$), $F' = \begin{cases} \{(s,s) \mid s \in S\} \cup \{(s_0, x)\} & s_0 \in F \\ \{(s,s) \mid s \in S\} & s_0 \notin F \end{cases}$,
and
$$\delta'((s, s'), a) = \begin{cases} \{(\delta(s,a), z) \mid z \in S, \delta(z, e) = s' \text{ for some } e \in \Sigma\} & \text{if } s' \neq x \\ \{(\delta(s,a), z) \mid z \in S, \delta(z, e) \in F \text{ for some } e \in \Sigma\} & \text{if } s' = x \end{cases}$$
for all $(s, s') \in S'$ and $a \in \Sigma$.
It can now be shown that $L(A') = \frac{1}{2}L$.

Example: Consider the DFA



Our above construction leads to the following NFA for $\frac{1}{2}L(A)$:



Other examples can be generated using the following script and embedding it into the automaton tool

```
def half_L_construction(A):
    """
    DFA -> NFA
    when given a DFA A (not containing a state named x) as input, the algorithm returns an NFA for 1/2 L(A)
    """
    [Sigma, S, delta, s0, F] = A
    x = "x"
    S_ = {(s,s_) for s in S for s_ in S} | {(s0, x)}
    F_ = {(s,s) for s in S}
    if s0 in F:
      F_ |= {(s0,x)}

    delta_ = {}
    for (s1, s2) in S_:
      for a in Sigma:
        delta_[(s1,s2), a] = {(s, s_) for s in S for s_ in S for e in Sigma\
          if delta[s1,a] == s and delta[s_, e] == s2}
        if s2 == x:
          delta_[(s0,x), a] = {(s,s_) for s in S for s_ in S for e in Sigma\
            if delta[s0,a] == s and delta[s_, e] in F}
    return [Sigma, S_, delta_, (s0,x), F_]
```

2. Define for each state $q \in S$:

$$L_q = \{w \in \Sigma^* \mid \overline{\delta}(s_0, w) = q\} \quad \text{and} \quad L'_q = \{w \in \Sigma^* \mid \exists_{x \in \Sigma^{|w|}} \overline{\delta}(q, x) \in F\}.$$

The language L_q is in FA as it is accepted by the DFA $(\Sigma, S, \delta, s_0, \{q\})$. Moreover, $L'_q \in$ FA as it is accepted by the NFA $(\Sigma, S, \delta', q, F)$ with $\delta' \colon S \times \Sigma \to \mathcal{P}(S)$ defined via $\delta'(s, a) = \{\delta(s, e) \mid e \in \Sigma\}$.
Then it holds $\frac{1}{2}L = \bigcup_{q \in S} \left(L_q \cap L'_q\right)$ and then by iterated application of Theorem 3.21, $\frac{1}{2}L$ is regular.

3. For a word $w \in \Sigma^*$ we define

$$R_w = \{(p, q) \in S \times S \mid \exists_{x \in \Sigma^{|w|}} \overline{\delta}(p, x) = q\}.$$

Consider the DFA $A' = (\Sigma, S', \delta', R_\varepsilon, F')$ with $S' = S \times \mathcal{P}(S \times S)$, $\delta' \colon S' \times \Sigma \to S'$ defined via

$$\delta'((s, R), a) = \Big(\delta(s, a), \{(p, q) \in S \times S \mid \exists_{r \in S} \exists_{e \in \Sigma} (p, r) \in R \wedge \delta(r, e) = q\}\Big) \qquad \text{for all } s \in S,\ R \subseteq \mathcal{P}(S \times S), \text{ and } a \in \Sigma$$

and

$$F' = \{(p, R) \mid p \in S \wedge R \in \mathcal{P}(S \times S) \wedge (\{p\} \times F) \cap R \neq \emptyset\}$$

One can now prove that for all words $w \in \Sigma^*$ it holds $\overline{\delta'}((s_0, R_\varepsilon), w) = (\overline{\delta}(s_0, w), R_w)$.