Abschlussklausur Theoretische Informatik I

29. August 2018

---

1. **Bitte ausfüllen:**

   Vorname:

   Nachname:

   Matrikelnummer:

   Unterschrift:

2. Achten Sie darauf, dass Ihre Klausur vollständig ist und getackert bleibt (16 Blätter)!

3. Benutzen Sie **nur das an dieses Blatt angeheftete Papier**. Bei Bedarf können wir weitere Leerblätter austeilen. Wenn der Platz auf der Vorderseite des jeweiligen Aufgabenblatts nicht ausreicht, machen Sie kenntlich, wo Sie die Bearbeitung der Aufgabe fortsetzen.

4. Als Hilfsmittel sind **ausschließlich** Sprachwörterbücher und ein zweiseitig **handgeschriebenes DIN A4-Blatt** erlaubt. Elektronische Geräte müssen während der Klausur ausgeschaltet bleiben. Täuschungsversuche werden mit 0 Punkten gewertet und dem Prüfungsamt gemeldet.

5. Schreiben Sie leserlich und bearbeiten Sie Ihre Klausur **nicht mit Bleistift** und auch **nicht in roter oder grüner Farbe!**

6. Wir werden das Deckblatt während der Klausur auf korrekte Daten überprüfen. Legen Sie dazu Ihren **Studentenausweis** und einen **amtlichen Lichtbildausweis** bereit.

Die Bearbeitungszeit beträgt 120 Minuten.

Mit 40 Punkten ist die Klausur sicher bestanden.

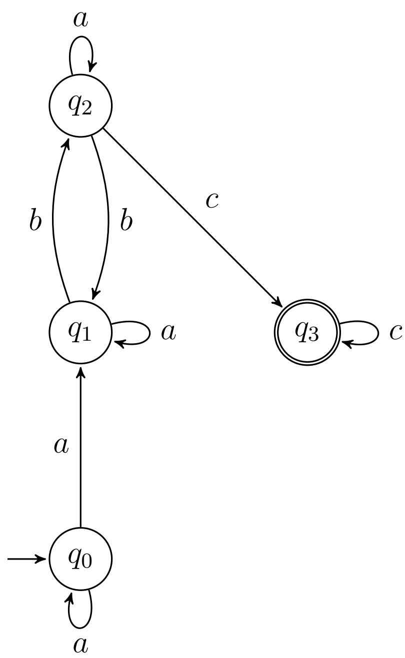
Punkteverteilung: (wird von den Korrektoren ausgefüllt)

Aufgabe	1	2	3	4	5	6	7	8	9	10	Σ
Max.	10	10	10	10	10	10	10	10	10	10	**100**
Punkte											

1. Ardens Lemma

10 Punkte

Gegeben sei der folgende NFA A über dem Alphabet $\Sigma = \{a, b, c\}$:



Geben Sie das zu A gehörige Gleichungssystem an und lösen Sie dieses unter Verwendung von Ardens Lemma.

2. CYK-Algorithmus

10 Punkte

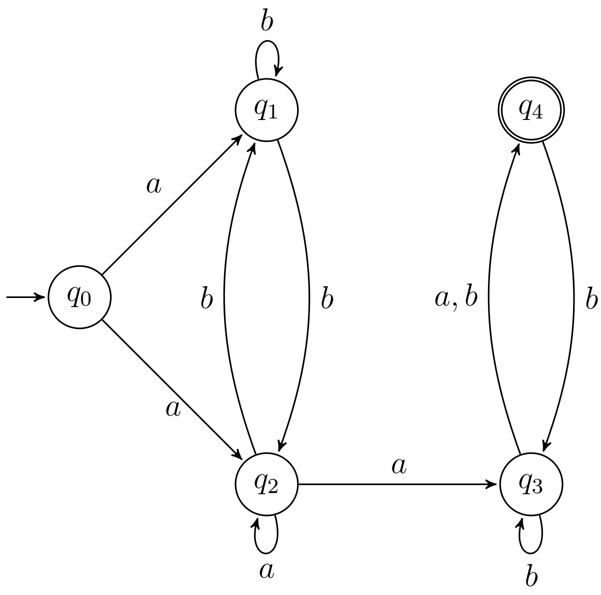
Entscheiden Sie mit Hilfe des Cocke–Younger–Kasami-Algorithmus, ob das Wort *aaabba* von folgender Grammatik erzeugt wird:

$$\begin{aligned} S &\to AB \mid AC, \\ A &\to AA \mid a, \\ B &\to SB \mid a, \\ C &\to CC \mid b. \end{aligned}$$

3. Determinisierung

10 Punkte

Determinisieren Sie folgenden NFA über $\Sigma = \{a, b\}$ unter Verwendung der Potenzmengen-Konstruktion:



4. Myhill-Nerode

$\boxed{7 + 3 = 10 \text{ Punkte}}$

Gegeben sei die Sprache:

$$L = \{w \in \{a, b, c\}^* \mid w \text{ enthält } aaa \text{ oder } c\}.$$

a) Geben Sie alle Klassen der Nerode-Rechtskongruenz an.

b) Konstruieren Sie den Äquivalenzklassen-Automaten A_L.

5. Pumping-Lemma

10 Punkte

Es sei $\Sigma = \{a, b, c\}$ und $w \in \Sigma^*$. Wir bezeichnen mit $|w|_a$ (analog $|w|_b$) die Anzahl der a's in w.

Zeigen Sie mit Hilfe des Pumping-Lemmas, dass die Sprache

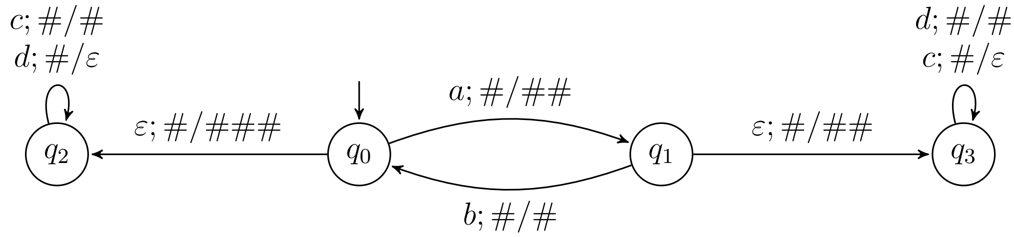
$$L = \{w \in \Sigma^* \mid |w|_a + 1 < |w|_b\}$$

nicht regulär ist.

6. Tripelkonstruktion

$\boxed{\text{10 Punkte}}$

Gegeben sei der folgende PDA P über dem Alphabet $\Sigma = \{a, b, c, d\}$ mit initialem Stack-Symbol $\#$. Ferner akzeptiert P mit leerem Stack.



Benutzen Sie das Verfahren aus der Vorlesung, um eine kontextfreie Grammatik G mit $L(G) = L(P)$ zu konstruieren.

7. Pushdown-Automaten

$\boxed{7 + 3 = 10 \text{ Punkte}}$

Sei $n \in \mathbb{N}$ eine natürliche Zahl. Mit $\mathsf{bin}_l(n)$ (resp. $\mathsf{bin}_m(n)$) bezeichnen wir die Binärdarstellung von n mit least-significant-bit-first (resp. most-sigificant-bit-first). Zum Beispiel ist $\mathsf{bin}_l(19) = 11001$ und $\mathsf{bin}_m(19) = 10011$. Betrachten Sie folgende Sprache über dem Alphabet $\Sigma = \{0, 1, \#\}$.

$$\mathcal{L} = \{\mathsf{bin}_l(n)\#\mathsf{bin}_m(n+1) \in \Sigma^* \mid n \in \mathbb{N}\}$$

a) Zeigen Sie, dass $\mathcal{L}$ eine kontextfreie Sprache ist.

b) Betrachten Sie die nicht-kontextfreie Sprache $\mathcal{L}'$, die wie folgt definiert ist.

$$\mathcal{L}' = \{\mathsf{bin}_l(n)\#\mathsf{bin}_l(n+1) \in \Sigma^* \mid n \in \mathbb{N}\}$$

Begründen Sie warum man (intuitiv) mit einer Konstruktion wie in a) nicht zeigen kann, dass $\mathcal{L}'$ kontextfrei ist.

8. Fragen zu Sprachen

$\boxed{2 + 2 + 2 + 2 + 2 = 10 \text{ Punkte}}$

Beantworten Sie die folgenden Fragen. Begründen Sie ihre Antwort mit einem kurzen Beweis oder einem Gegenbeispiel.

a) Es sei L eine reguläre Sprache über Σ und $f : \Sigma^* \to \Sigma^*$ eine Funktion über Wörtern. Ist die Sprache $f(L) = \{f(w) \mid w \in L\}$ dann auch immer regulär?

b) Sei $M = \{42n + 5 \mid n \in \mathbb{N}\}$. Ist dann $L = \{a^k \mid k \in M\}$ regulär?

c) Es seien L eine kontextfreie Sprache und $n \in \mathbb{N} \setminus \{0\}$. Ist dann die Sprache $L^n = \{w^n \mid w \in L\}$ auch kontextfrei?

d) Es seien L_1, L_2 kontextfreie Sprachen und $L_1 \subseteq L \subseteq L_2$. Ist dann L auch kontextfrei?

e) Ist $(\mathbb{N}, \leq)$ ($\leq$ ist wie üblich auf den natürlichen Zahlen definiert) ein vollständiger Verband?

9. Swap-Sprache

10 Punkte

Es sei Σ ein endliches Alphabet mit $\# \notin \Sigma$ und L eine reguläre Sprache über Σ.

Zeigen Sie, dass dann die Sprache

$$L_{\mathsf{swap}} = \{w_2.\#.w_1 \in (\Sigma \cup \{\#\})^* \mid w_1.w_2 \in L\}$$

auch regulär ist. Erklären Sie dazu, wie man aus einem DFA für L einen NFA für L_{swap} konstruiert.

10. Counter-Automaten

3+7 = 10 Punkte

Ein Counter-Automat über einem Alphabet Σ ist ein Tupel $A = (Q, \rightarrow, q_0, Q_F)$, wobei

- Q die endliche Menge an Zuständen,
- die Transitionsrelation $\rightarrow \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times \{\mathsf{Inkr}, \mathsf{Dekr}, \mathsf{Test0}\} \times Q$,
- $q_0 \in Q$ der Anfangszustand und
- $Q_F \subseteq Q$ die endliche Menge an Endzuständen ist.

Die Konfigurationen eines Counter-Automaten bestehen aus einem Zustand $q \in Q$ und einem Counter-Wert $n \in \mathbb{N}$.

In der Transitionsrelation steht die Komponente $\{\mathsf{Inkr}, \mathsf{Dekr}, \mathsf{Test0}\}$ für die möglichen Counter-Operationen.

- Inkr erhöht den Wert des Counters um 1.
- Dekr zieht 1 vom Wert des Counters ab. Diese Operation kann nur ausgeführt werden, wenn der Counter mindestens Wert 1 hat. Ist dies während eines Laufes nicht der Fall, kann die Transition zu diesem Zeitpunkt nicht verwendet werden.
- Test0 testet, ob der Wert des Counters 0 ist. Eine Transition mit dieser Counter-Operation kann in einem Lauf nur verwendet werden, wenn der Wert des Counters zu diesem Zeitpunkt 0 ist.

Ein Lauf auf Wort $v \in \Sigma$ in einem Counter-Automaten ist eine Sequenz von Konfigurationen, die im Anfangszustand q_0 mit Counter-Wert 0 beginnt:

$$(q_0, 0) \xrightarrow{w_1} (q_1, i_1) \xrightarrow{w_2} (q_2, i_2) \xrightarrow{w_3} \ldots \xrightarrow{w_n} (q_n, i_n),$$

wobei $w_1, \ldots, w_n \in \Sigma \cup \{\varepsilon\}$ und $v = w_1.w_2 \ldots w_n$. Wir erlauben also auch ε-Transitionen. Der Lauf auf Wort v ist akzeptierend, falls q_n ein Endzustand ist, also $q_n \in Q_F$. Die Sprache eines Counter-Automaten A ist also definiert durch

$$\mathcal{L}(A) = \{v \in \Sigma^* \mid \text{es gibt einen akzeptierenden Lauf von } A \text{ auf } v\}.$$

a) Man kann zeigen, dass Counter-Automaten alle Sprachen akzeptieren, die von endlichen Automaten akzeptiert werden. Zeigen Sie nun, dass Counter-Automaten echt mächtiger sind als endliche Automaten. Geben Sie dazu einen Counter-Automaten an, der die Sprache $L = \{a^n b^n \mid n \in \mathbb{N}\}$ akzeptiert.

b) Zeigen Sie, dass PDAs mindestens so mächtig sind wie Counter-Automaten. Sei dazu A ein gegebener Counter-Automat. Erklären Sie, wie ein PDA M mit $\mathcal{L}(M) = \mathcal{L}(A)$ konstruiert werden kann.