

Reguläre Klausur zur Vorlesung Formale Sprachen und Komplexität

Die Bearbeitungszeit beträgt **120 Minuten**. Hilfsmittel sind nicht erlaubt, auch das Mitführen ausgeschalteter elektronischer Geräte (inklusive Smartwatches aber exklusive normaler Uhren) wird als Täuschungsversuch gewertet. Schreiben Sie Ihren vollständigen Namen und Ihre Matrikelnummer deutlich lesbar auf dieses Deckblatt, sowie Ihren Namen in die Kopfzeile auf jedem Blatt der Klausurangabe. Geben Sie alle Blätter ab. Lassen Sie diese zusammengeheftet. Verwenden Sie nur **dokumentenechte Stifte** und **weder** die Farbe **rot noch grün**.

Kontrollieren Sie, ob Sie alle Aufgabenblätter erhalten haben. Aufgabenstellungen befinden sich auf den **Seiten 1–11**. Sie dürfen die Rückseiten für Nebenrechnungen nutzen. Falls Sie die Rückseiten für Antworten nutzen, so markieren Sie klar, was zu welcher Aufgabe gehört und geben Sie in der entsprechenden Aufgabe an, wo alle Teile Ihrer Antwort zu finden sind. Streichen Sie alles durch, was nicht korrigiert werden soll.

Es gibt 5 unterschiedlich gewichtete Aufgaben zu insgesamt 100 Punkten. Mit 50 Punkten haben Sie sicher bestanden. Die Teilaufgaben können unabhängig voneinander bearbeitet werden.

Mit Ihrer Unterschrift bestätigen Sie, dass Sie zu Beginn der Klausur in ausreichend guter gesundheitlicher Verfassung sind und diese Klausurprüfung verbindlich annehmen.

Nachname (in GROSSBUCHSTABEN):

Vorname (in GROSSBUCHSTABEN):

Matrikelnummer:

Studiengang:

Hiermit erkläre ich die Richtigkeit der obigen Angaben:

Unterschrift

Die folgende Tabelle nicht ausfüllen:

Aufgabe	1	2	3	4	5	Σ
Punkte	30	18	20	22	10	100
Erreicht						

Aufgabe 1 (Reguläre Sprachen):**(30 Punkte)**

- a) Sei L_1 die Sprache über dem Alphabet $\{a, b, c\}$, die vom regulären Ausdruck

$$(a|b|c)^*(ab|ba)(cb|bc)(a|b|c)^*$$

erzeugt wird. Geben Sie den Zustandsgraphen eines nichtdeterministischen endlichen Automaten (ohne ε -Übergänge) an, der L_1 akzeptiert.

Fortsetzung von Aufgabe 1:

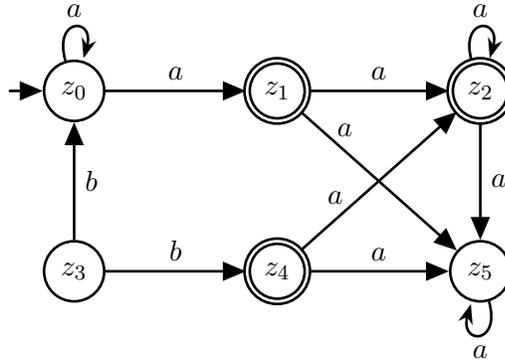
b) Sei L_2 die Sprache über dem Alphabet $\{a, b\}$, die vom regulären Ausdruck

$$(a|b)^*aaa(a|b)^*$$

erzeugt wird. Geben Sie den Zustandsgraphen eines deterministischen endlichen Automaten an, der L_2 akzeptiert. Bitte zeichnen Sie alle Übergänge und Zustände sorgfältig, inklusive des Müllzustands (falls vorhanden).

Fortsetzung von Aufgabe 1:

- c) Der nichtdeterministische endliche Automat M_1 über dem Alphabet $\{a, b\}$ ist durch den folgenden Zustandsgraphen gegeben:



Berechnen Sie einen deterministischen endlichen Automaten M'_1 aus M_1 mit der Potenzmengenkonstruktion. Es ist ausreichend, die erreichbaren Zustände von M'_1 anzugeben. Sie müssen keine Begründung angeben, nur das Ergebnis.

Aufgabe 2 (Nicht reguläre Sprachen):**(18 Punkte)**

a) Zeigen Sie mit dem Satz von Myhill und Nerode, dass die Sprache

$$L_1 = \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

über dem Alphabet $\{a, b, c\}$ nicht regulär ist. Geben Sie für jede Äquivalenzklasse $[u_i]_{\sim_{L_1}}$, die Sie im Beweis nutzen, ein Suffix w_i an, sodass $u_i w_i \in L_1$ aber $u_j w_i \notin L_1$ für jedes $j \neq i$.

Fortsetzung von Aufgabe 2:

b) Beweisen Sie mithilfe der Abschlusseigenschaften der regulären Sprachen, dass die Sprache

$$L_2 = \overline{\{a^i b^i c^i d^j \mid i, j \in \mathbb{N}\}}$$

über dem Alphabet $\{a, b, c, d\}$ nicht regulär ist. \bar{L} bezeichnet dabei das Komplement einer Sprache L . Sie dürfen annehmen, dass die Sprache $L_1 = \{a^i b^i c^i \mid i \in \mathbb{N}\}$ aus Teilfrage a) nicht regulär ist.

Zur Erinnerung: Die regulären Sprachen sind unter Vereinigung, Schnitt, Komplement, Produkt und Kleeneschem Abschluss abgeschlossen.

Aufgabe 3 (Kontextfreie Sprachen):**(20 Punkte)**

- a) Die Sprache L_1 über dem Alphabet $\Sigma_1 = \{a, b, c, d, e\}$ sei definiert als

$$L_1 = \{a^m b^n c d^n e^m c \mid m, n \in \mathbb{N}\}$$

Geben Sie eine kontextfreie Grammatik G_1 , die L_1 erzeugt, als 4-Tupel an. Die Grammatik darf keine ε -Produktionen enthalten. Erläutern Sie, warum G_1 die Sprache L_1 erzeugt. Beschreiben Sie beispielsweise, welche „Aufgabe“ die einzelnen Nichtterminale bei der Erzeugung übernehmen.

Geben Sie zusätzlich eine Linksableitung für das Wort $abcddec$ für Ihre Grammatik an.

Fortsetzung von Aufgabe 3:

b) Sei $G_2 = (V_2, \Sigma_2, P_2, S)$ mit $V_2 = \{S, A, B\}$, $\Sigma_2 = \{a, b\}$ und

$$P_2 = \{S \rightarrow AB \mid BA, A \rightarrow AA \mid AB \mid a, B \rightarrow BA \mid a \mid b\}$$

eine kontextfreie Grammatik in Chomsky-Normalform.

Entscheiden Sie, ob $bbabab \in L(G_2)$ ist, indem Sie den Algorithmus von Cocke, Younger und Kasami (CYK-Algorithmus) ausführen. Geben Sie die dabei entstehende Tabelle und die Ausgabe des Algorithmus an.

Aufgabe 4 (Berechenbarkeit und Entscheidbarkeit):**(22 Punkte)**

- a) Geben Sie eine primitiv rekursive Funktion f an, die $n \mapsto n!$ berechnet. Zum Beispiel soll $f(4) = 4! = 24$ gelten. Sie können die Tatsachen nutzen, dass

$$0! = 1 \quad \text{und} \quad n! = n \cdot (n-1)! \quad \text{für } n > 0$$

gelten. Erläutern Sie, warum f primitiv rekursiv ist. Sie dürfen dabei annehmen, dass Multiplikation (\cdot) primitiv rekursiv ist.

Zur Erinnerung:

Eine Funktion $f : \mathbb{N}^k \rightarrow \mathbb{N}$ ist *primitiv rekursiv*, wenn sie der folgenden Definition genügt:

- Jede *konstante Funktion* $f(x_1, \dots, x_k) = c$ mit $c \in \mathbb{N}$ ist primitiv rekursiv.
- Die *Projektionsfunktionen* $\pi_i^k(x_1, \dots, x_k) = x_i$ sind primitiv rekursiv.
- Die *Nachfolgerfunktion* $\text{succ}(x) = x + 1$ ist primitiv rekursiv.
- *Komposition*: Wenn die Funktionen $g : \mathbb{N}^m \rightarrow \mathbb{N}$ und $h_i : \mathbb{N}^k \rightarrow \mathbb{N}$ für $i = 1, \dots, m$ primitiv rekursiv sind, dann ist auch f mit $f(x_1, \dots, x_k) = g(h_1(x_1, \dots, x_k), \dots, h_m(x_1, \dots, x_k))$ primitiv rekursiv.
- *Rekursion*: Wenn $g : \mathbb{N}^{k-1} \rightarrow \mathbb{N}$ und $h : \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ primitiv rekursiv sind, dann ist auch folgende Funktion primitiv rekursiv:

$$f(x_1, \dots, x_k) = \begin{cases} g(x_2, \dots, x_k) & \text{falls } x_1 = 0 \\ h(f(x_1 - 1, x_2, \dots, x_k), x_1 - 1, x_2, \dots, x_k) & \text{sonst} \end{cases}$$

Fortsetzung von Aufgabe 4:

- b) Das Postsche Korrespondenzproblem (PCP) ist bekanntlich unentscheidbar. Gegeben sei die folgende Variante des Problems:

Sei Σ ein Alphabet mit $|\Sigma| > 1$. Eine Instanz des *gespiegelten Postschen Korrespondenzproblems* (GPCP) besteht aus einer endlichen Folge $(x_1, y_1), \dots, (x_k, y_k)$ von Wortpaaren mit $x_i, y_i \in \Sigma^+$. Das Entscheidungsproblem ist die Frage, ob es eine Folge von Indizes i_1, \dots, i_m mit $i_j \in \{1, \dots, k\}$ und $m > 0$ gibt, sodass $x_{i_1} \cdots x_{i_m} = \overline{y_{i_1} \cdots y_{i_m}}$ gilt, wobei $\overline{a_1 \cdots a_n}$ als $a_n \cdots a_1$ definiert ist.

Der folgende Beweis ist falsch. Lokalisieren Sie den Fehler und begründen Sie, wieso der Beweis falsch ist.

Satz: GPCP ist unentscheidbar.

Beweis: Wir zeigen $\text{PCP} \leq \text{GPCP}$.

Sei $f((x_1, y_1), \dots, (x_k, y_k)) = (x_1, \overline{y_1}), \dots, (x_k, \overline{y_k})$.

Die Funktion f ist offensichtlich total und berechenbar.

Korrektheit:

$$(x_1, y_1), \dots, (x_k, y_k) \in \text{PCP}$$

g.d.w. es gibt i_1, \dots, i_m , sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$

g.d.w. es gibt i_1, \dots, i_m , sodass $x_{i_1} \cdots x_{i_m} = \overline{\overline{y_{i_1} \cdots y_{i_m}}}$

g.d.w. es gibt i_1, \dots, i_m , sodass $x_{i_1} \cdots x_{i_m} = \overline{y_{i_1} \cdots y_{i_m}}$

g.d.w. $(x_1, \overline{y_1}), \dots, (x_k, \overline{y_k}) \in \text{GPCP}$.

Wenn GPCP entscheidbar wäre, könnten wir obige Konstruktion verwenden, um PCP zu entscheiden. Dies wäre dann ein Widerspruch zur Unentscheidbarkeit von PCP. Daher muss auch GPCP unentscheidbar sein.

Fortsetzung von Aufgabe 4:

c) Der Satz von Rice besagt:

Sei \mathcal{R} die Klasse aller turingberechenbaren Funktionen. Sei \mathcal{S} eine nichtleere echte Teilmenge von \mathcal{R} . Dann ist folgende Sprache unentscheidbar:

$$C(\mathcal{S}) = \{w \mid \text{die von der deterministischen Turingmaschine } M_w \text{ berechnete Funktion liegt in } \mathcal{S}\}$$

wobei M_w die Turingmaschine mit der Gödelnummer w bezeichnet.

Wenden Sie für jede der beiden Aussagen unten den Satz von Rice an, um sie zu beweisen, oder erklären Sie, warum der Satz nicht anwendbar ist.

(i) Es ist unentscheidbar, ob eine gegebene deterministische Turingmaschine für keine Eingabe $i \in \mathbb{N}$ die Zahl i berechnet.

(ii) Es ist unentscheidbar, ob eine gegebene deterministische Turingmaschine für alle Eingaben erst nach 1000 Schritten anhält.

Aufgabe 5 (Komplexität):**(10 Punkte)**

In der Vorlesung wurde das VERTEX-COVER-Problem vorgestellt:

gegeben: ein ungerichteter Graph $G = (V, E)$ und eine Zahl $k \in \mathbb{N}$
gefragt: Besitzt G eine überdeckende Knotenmenge der Größe höchstens k ?

Für einen Graphen $G = (V, E)$ ist $V' \subseteq V$ eine *überdeckende Knotenmenge*, wenn jede Kante aus E mindestens einen Knoten in V' hat, d. h. für alle Knoten $u, v \in V$: $\{u, v\} \in E$ impliziert $u \in V' \vee v \in V'$.

Sie dürfen als bekannt annehmen, dass VERTEX-COVER \mathcal{NP} -vollständig ist.

Nun führen wir folgendes HYPERCOVER-Problem ein:

gegeben: eine endliche Grundmenge X ,
eine Menge von Mengen $\mathbf{M} \subseteq \mathcal{P}(X)$ und
eine Zahl $\ell \in \mathbb{N}$
gefragt: Gibt es eine Menge $X' \subseteq X$, sodass $|X'| \leq \ell$ und jede Menge aus \mathbf{M} mindestens ein Element in X' hat, d. h. für jede Menge $M \in \mathbf{M}$ gilt $M \cap X' \neq \emptyset$?

Beweisen Sie mithilfe einer Polynomialzeit-Reduktion, dass HYPERCOVER \mathcal{NP} -schwer ist.

