Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: ______ Kennwort: ______ Arbeitsplatz-Nr.: ______	**Herbst 2013**	**46113**

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen
— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Theoretische Informatik**

Anzahl der gestellten Themen (Aufgaben): **2**

Anzahl der Druckseiten dieser Vorlage: **8**

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1:

Reguläre Sprachen I

Sei $\Sigma = \{a, b\}$ und L_1 die Sprache bestehend aus allen Wörtern gerader Länge über dem Alphabet Σ. Sei L_2 die Sprache bestehend aus allen Wörtern über Σ, in denen b genau zweimal vorkommt.

a) Geben Sie jeweils einen regulären Ausdruck für L_1 und L_2 an.

b) Eine Grammatik heißt *rechtslinear*, wenn es für jede Produktion Nichtterminale A, B und ein Terminalsymbol x gibt, so dass die Produktion die Form $A \to xB$ oder $A \to \epsilon$ hat. Geben Sie eine rechtslineare Grammatik für die Sprache $L_1 \cap L_2$ an.

c) Geben Sie einen endlichen Automaten (gegebenenfalls mit ε-Transitionen) an, der die Sprache $L(a(b^*|c)^*(a|b))$ erkennt.

Fortsetzung nächste Seite!

Aufgabe 2:

Reguläre Sprachen II

Sei $\Sigma = \{0, 1\}$. Wir definieren die Inversion $\overline{u}$ von Wörtern $u \in \Sigma^*$ wie folgt:

$$\begin{aligned}\overline{\varepsilon} &:= \varepsilon\\ \overline{0v} &:= 1\overline{v} \text{ für } v \in \Sigma^*\\ \overline{1v} &:= 0\overline{v} \text{ für } v \in \Sigma^*\end{aligned}$$

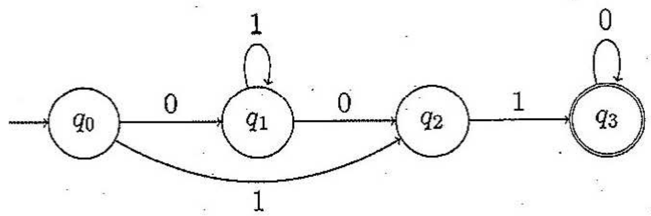
Für eine Sprache $L \subseteq \Sigma^*$ definieren wir

$$\mathsf{inverse\_from}(L) := \{a_1 \cdots a_n \mid \exists k. \in \mathbb{N} a_1 \cdots a_k \overline{a_{k+1} \cdots a_n} \in L \land 0 \leq k \leq n\}$$

die Sprache, die durch Invertieren der Wörter aus L ab beliebigen Positionen hervorgeht.

Beispiel: Ist $L = \{000, 11\}$, so ist $\mathsf{inverse\_from}(L) = \{000, 001, 011, 111, 11, 10, 00\}$.

a) Sei L die Sprache, die durch den folgenden endlichen Automaten akzeptiert wird. Geben Sie einen endlichen Automaten (gegebenenfalls mit ε-Transitionen) für $\mathsf{inverse\_from}(L)$ an.
Hinweis: Starten Sie mit einem endlichen Automaten für L und fügen Sie zusätzliche Zustände und Übergänge ein.



b) Zeigen Sie: Ist $L \subseteq \Sigma^*$ regulär, so ist $\mathsf{inverse\_from}(L)$ ebenfalls eine reguläre Sprache.
Hinweis: Verallgemeinern Sie Ihre Konstruktion von Teilaufgabe 1.
Erläutern Sie Ihre Konstruktionsidee!

Fortsetzung nächste Seite!

Aufgabe 3:

Kontextfreie Sprachen

a) Geben Sie eine kontextfreie Grammatik für die folgende Sprache an:

$$L = \{a^n b^m c^\ell b^{2n} \mid 0 \leq n, 0 \leq m < \ell\}$$

b) Sei $G_1 = (N_1, \Sigma_1, P_1, S)$ mit $N_1 = \{S, T\}$, $\Sigma_1 = \{a, b, c, d\}$ und P_1 bestehe aus den folgenden Produktionen:

$$S \to TaS \mid T \qquad T \to bSc \mid d$$

i) Zeigen Sie $dabdc \in L(G_1)$.

ii) Bringen Sie die Grammatik G_1 in Chomsky-Normalform.

Fortsetzung nächste Seite!

Aufgabe 4:

Berechenbarkeit und Entscheidbarkeit

Sei $\Sigma = \{0,1\}$. Wir nehmen an, dass jedes $w \in \Sigma^*$ eine Turingmaschine kodiert und bezeichnen diese TM mit M_w. Die von M_w berechnete Funktion bezeichnen wir mit φ_w.

a) Welche der folgenden Mengen sind entscheidbar? Begründen Sie kurz.

i) $L_1 := \{w \in \Sigma^* \mid \text{es gibt eine Eingabe, für die } M_w \text{ das leere Wort ausgibt}\}$

ii) $L_2 := \{w \in \Sigma^* \mid \exists n \in \mathbb{N} \text{ mit } w = 0^n 1^n\}$

iii) $L_3 := \{w \in \Sigma^* \mid \forall x \in \Sigma^*. x \in L(0^*1^*) \leftrightarrow \varphi_w(x) = 1\}$

b) Ist die folgende Funktion berechenbar? Begründen Sie kurz.

$$h(w) = \begin{cases} 1 & \text{wenn } \varphi_w(01) = 10 \\ 0 & \text{sonst} \end{cases}$$

c) Konstruieren Sie eine Turingmaschine M mit Eingabealphabet Σ, die jedes Vorkommen von 1 im Eingabewort verdoppelt. Das heißt, M soll die folgende Funktion $f : \Sigma^* \to \Sigma^*$ berechnen:

$$f(\varepsilon) = \varepsilon \qquad f(1w) = 11f(w) \qquad f(0w) = 0f(w)$$

Beschreiben Sie auch Ihre Konstruktionsidee!
Beispiel: Für die Eingabe 01101 soll M die Ausgabe 01111011 liefern.

Fortsetzung nächste Seite!

Aufgabe 5:

Komplexitätstheorie

Wir betrachten das Problem, in einer Schulklasse Gruppenarbeit durchzuführen. Es gibt mehrere *Arbeitsthemen*; jedem *Schüler* wird ein Thema zugeordnet und alle Schüler mit dem gleichen Thema arbeiten in einer Gruppe. Allerdings weiß man, dass wenn gewisse *Teilgruppen* von Schülern vollständig einem Arbeitsthema zugeordnet sind, sie sich gegenseitig und somit die ganze Gruppe stören. Man möchte durch eine geeignete Zuordnung von Schülern zu Arbeitsthemen sicherstellen, dass keine Störungen auftreten.

Formal betrachtet definieren wir das Problem GRUPPENARBEIT wie folgt:

Gegeben: $(S, \mathcal{T}, A)$:
S ... Menge von Schülern
$\mathcal{T}$ ... störende Teilgruppen, $\mathcal{T} \subseteq \mathcal{P}(S)$
A ... Menge der Arbeitsthemen

Problem: Existiert eine Zuordnung $g : S \to A$, sodass aus jeder störenden Teilgruppe mindestens zwei Schüler unterschiedlichen Arbeitsthemen zugeordnet sind:

$$\forall T \in \mathcal{T} : \exists t_1 \in T : \exists t_2 \in T : g(t_1) \neq g(t_2)$$

a) Begründen Sie: GRUPPENARBEIT liegt in NP.

b) Geben Sie eine polynomielle Reduktion von 3-COL auf GRUPPENARBEIT an. Begründen Sie kurz die polynomielle Laufzeit und die Korrektheit und Vollständigkeit der Reduktion.
Hinweis: 3-COL ist definiert als das Problem, für einen gegebenen Graphen (V, E) zu entscheiden, ob die Knoten V so mit drei Farben eingefärbt werden können, dass keine durch Kanten in E benachbarte Knoten derselben Farbe zugeordnet sind.

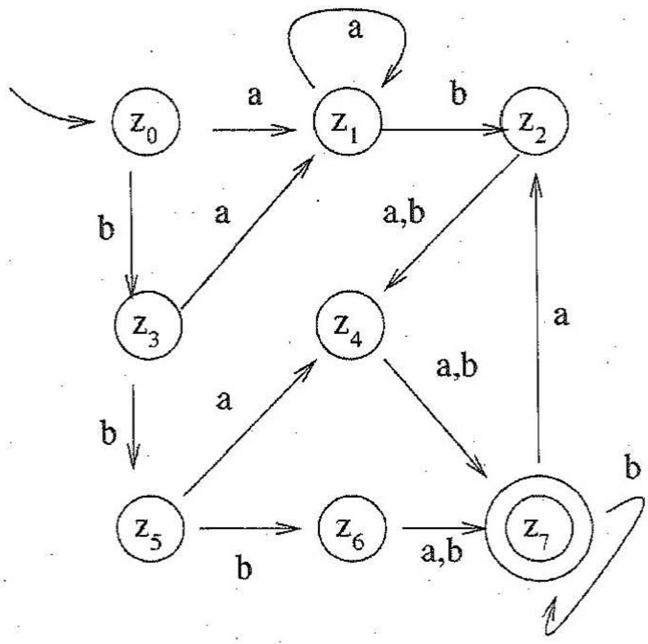
c) Zeigen Sie: GRUPPENARBEIT ist NP-vollständig.
Hinweis: Sie dürfen verwenden, dass 3-COL NP-hart ist.

Thema Nr. 2

(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Teilaufgabe I: Minimierung und Äquivalenzen Gegeben ist folgender deterministische endliche Automat (DEA) A über dem Alphabet $\Sigma = \{a, b\}$.



a) Konstruieren Sie den zugehörigen Minimalautomaten und dokumentieren Sie Ihre Schritte auf geeignete Weise.

b) Geben Sie $L(A)$ als regulären Ausdruck an.

c) Geben Sie reguläre Ausdrücke für die Äquivalenzklassen der Myhill-Nerode Äquivalenzrelation $\sim_{L(A)}$ an. Erinnerung: Diese Klassen entsprechen den Zuständen des Minimalautomaten.

d) Sei $B = (Z, \Sigma, \delta, z_0, E)$ der von Ihnen konstruierte Minimalautomat. Wir definieren die Beobachtungsgleichheit $\simeq_B$ auf Σ^* durch $u \simeq_B v \Longleftrightarrow \forall z \in Z{:}\delta(z,u) = \delta(z,v)$.

Sollten Sie die vorhergehenden Aufgaben nicht gelöst haben, so nehmen Sie für B einen DEA Ihrer Wahl mit $L(B) = (a \mid b)b^*a(a \mid b)b^*$. Bitte vermerken Sie dann, dass Sie diese Option gewählt haben und geben Sie Ihren DEA B explizit an.

- Begründen Sie, dass $\simeq_B$ eine Äquivalenzrelation ist.
- Geben Sie zwei verschiedene Wörter an, die in derselben $\simeq_B$-Klasse liegen.
- Geben Sie zwei Wörter an, die in unterschiedlichen Klassen liegen.
- Begründen Sie, dass $\simeq_B$ nur endlich viele Klassen hat.

Fortsetzung nächste Seite!

Teilaufgabe II: Berechenbarkeit Bekanntlich ist das Halteproblem H_0 definiert als die Menge aller Codes von Turingmaschinen, die bei leerer Eingabe halten. I.Z.: $H_0 = \{e \mid M_e(\epsilon) \textit{ hält}\}$.

a) Begründen Sie, dass H_0 rekursiv aufzählbar (= semi-entscheidbar) ist, das Komplement $\overline{H_0} = \mathbb{N} \setminus H_0$ aber nicht.

b) Eine Turingmaschine M kann man effektiv zu einer anderen Maschine $tr(M)$ umbauen, sodass gilt: $tr(M)$ hält bei Eingabe $t \in \mathbb{N}$, genau dann, wenn M bei leerer Eingabe mindestens t Schritte rechnet. Begründen Sie dies.

c) Es sei $Tot = \{e \mid M_e \text{ hält für alle Eingaben } x \in \mathbb{N}\}$ das *Totalitätsproblem*.
Welche der folgenden Aussagen sind richtig? Begründen Sie jeweils Ihre Antwort!

- Die Konstruktion tr liefert eine Reduktion $\overline{H}_0 \leq Tot$.
- Die Konstruktion tr liefert eine Reduktion $Tot \leq \overline{H}_0$.
- Die Konstruktion tr liefert eine Reduktion $Tot \leq H_0$.
- Das Problem Tot ist semi-entscheidbar.
- Das Problem Tot ist nicht rekursiv aufzählbar.

Teilaufgabe III: Komplexität Beim Problem MAX2SAT ist eine Menge von Zweierklauseln und eine Zahl $k \in \mathbb{N}$ gegeben. Gefragt ist, ob eine Variablenbelegung existiert, die mindestens k von den Zweierklauseln erfüllt. Erinnerung: Zweierklausel: Disjunktion zweier Literale; Literal: negierte oder nichtnegierte aussagenlogische Variable.

Ist zum Beispiel $\mathcal{C} = \{X \vee \neg Y, Y \vee \neg Z, Z \vee \neg X, X \vee Y, \neg X \vee \neg Y\}$, so ist $\mathcal{C}, k = 4$ eine positive Instanz von MAX2SAT, $\mathcal{C}, k = 5$ aber eine negative Instanz.

Beim Problem CLIQUE hingegen ist ein ungerichteter Graph $G = (V, E)$ und $j \in \mathbb{N}$ gegeben und es ist gefragt, ob es in G mindestens j paarweise verbundene Knoten gibt (eine Clique der Größe j). Das Problem CLIQUE ist bekanntlich NP-vollständig.

Hat man einen ungerichteten Graphen $G = (V, E)$ gegeben, so konstruiert man eine Menge von Zweierklauseln wie folgt: Variablen sind die Knoten von G. Für je zwei Knoten v, v' mit $(v, v') \notin E$ gibt man eine Klausel $\neg v \vee \neg v'$ aus und außerdem für jedes $v \in V$ die Klausel $v \vee v$. Insgesamt also $|V| + d$ Klauseln, wobei d die Zahl der Nicht-Kanten in G ist.

a) Begründen Sie folgendes: Hat G eine Clique der Größe j, so kann man mindestens $d + j$ viele dieser Klauseln simultan erfüllen.

Es gilt auch die Umkehrung dieser Aussage (können irgend $d + j$ der Klauseln simultan erfüllt werden, so hat G eine Clique der Größe j), das dürfen Sie im folgenden ohne Begründung voraussetzen.

b) Was kann aus dieser Konstruktion nun gefolgert werden? Begründen Sie jeweils Ihre Antwort:

- Diese Konstruktion liefert eine Reduktion CLIQUE $\leq_p$ MAX2SAT.
- Diese Konstruktion liefert eine Reduktion MAX2SAT $\leq_p$ CLIQUE.
- Die Konstruktion liefert ein Entscheidungsverfahren für MAX2SAT mit polynomieller Laufzeit.
- MAX2SAT ist NP-vollständig.