TU Ilmenau, Fakultät IA
Institut TI, FG Komplexitätstheorie und Effiziente Algorithmen
Priv.-Doz. Dr. rer. nat. habil. K.-H. Niggl

Wiederholungsklausur „Komplexitätstheorie“ SS08

22. Juli 2008

NICHT MIT BLEISTIFT ODER ROTSTIFT SCHREIBEN!

Heften Sie die Blätter bei Abgabe zusammen, und tragen Sie auf jedem Blatt Ihren Namen, Vornamen, Studiennummer und Matrikel ein.

Name, Vorname:
Studiennummer und Matrikel:

Code

abgegeben: **2 Aufgabenblätter**
... **eigene Blätter**

Einsichtnahme

Datum, Unterschrift

Aufgabe	1	2	3	4	5	6	7	Ges.
erreichbare Punktzahl	30	12	16	9	11	12	10	100
erreichte Punktzahl								

Es sind keine Hilfsmittel, insbesondere Taschenrechner oder Mobiltelefone, zugelassen.

Aufgabe 1 (Fragenkatalog) [30 Punkte]

Bitte kreuzen Sie für jede der folgenden 30 Fragen entweder „JA“ oder „NEIN“ an.
Bewertung: Ist C die Anzahl der richtigen Antworten, so errechnet sich die Anzahl P der erzielten Punkte aus $P := \frac{3}{2} \cdot \max\{0, C - 10\}$. Insbesondere: Kein Kreuz wird als falsche Antwort gewertet.

(a) Welche der nachstehenden Aussagen folgt/folgen aus dem Satz von Ben-Or?

JA NEIN
[ ] [ ] Für jedes RRAM-Programm M für ELEMENT UNIQUENESS gilt für genügend große n: $T_M(n) \geq 0.38 \cdot \log n! - 0.68 \cdot n$
[ ] [ ] Für jedes RRAM-Programm M zur Berechnung der konvexen Hülle von n Punkten in der Ebene gilt für genügend große n: $T_M(n) \geq 0.16 \cdot \log(\#\mathrm{EU}_n) - O(n)$
[ ] [ ] Für jedes RRAM-Programm M für das Sortieren von n reellen Zahlen gilt für genügend große n: $T_M(n) \geq 0.38 \cdot n \cdot \log n - O(n)$

(b) Welche der folgenden Aussagen ist/sind korrekt, wenn $P \leq_N P'$ mit O-Konstante c gilt?

JA NEIN
[ ] [ ] Wenn P' einen Polynomialzeitalgorithmus besitzt, dann auch P.
[ ] [ ] Aus $T_P(n) \geq d \cdot n \cdot \log n - O(n)$ folgt $T_{P'}(n) < \frac{d}{c} \cdot n \cdot \log n - O(n)$.
[ ] [ ] Aus jedem RRAM-Programm M' für P' kann man ein RRAM-Programm M für P mit $T_{M'}(c \cdot n) \geq T_M(n) - O(n)$ konstruieren.

(c) Welche der folgenden Aussagen ist/sind korrekt, wenn $A \leq_T B$ gilt?

JA NEIN
[ ] [ ] Es existiert ein Polynom r, sodaß zu jeder Turingmaschine N für B eine Turingmaschine M für A mit $T_M(n) \leq T_N(r(n))$ existiert.
[ ] [ ] Wenn B effizient lösbar ist, dann ist auch A effizient lösbar.
[ ] [ ] Aus $B \leq_p C$ folgt $A \leq_T C$.

(d) Welche der folgenden Aussagen ist/sind für **alle** Sprachen L korrekt?

JA NEIN

[ ] [ ] Wenn PARTITION $\leq_p L$ gilt, dann gilt $L \in \mathsf{NP}$.
[ ] [ ] Wenn L von keiner $O(2^n)$-zeitbeschränkten TM akzeptiert wird, so gilt $L \notin \mathsf{NP}$.
[ ] [ ] Wenn $L \leq_p$ SAT und CLIQUE $\leq_p L$ gilt, dann gilt $L \in \mathsf{NPC}$.

(e) Welche der folgenden Aussagen ist/sind korrekt?

JA NEIN

[ ] [ ] Wenn 3-COLORING $\in \mathsf{P}$ gilt, dann gilt $\mathsf{P}=\mathsf{NP}$.
[ ] [ ] Wenn $\mathsf{P}=\mathsf{NP}$ gilt, dann gilt SAT $\leq_p \{\mathrm{a, b, l, blablabla}\}$.
[ ] [ ] Wenn $\mathsf{P} \neq \mathsf{NP}$ gilt, dann gilt nicht 3-COLORING $\leq_p$ k-COLORING für $k \geq 4$.

(f) Welche der folgenden Aussagen ist/sind für **alle** Sprachen $L \subseteq \Sigma^*$ mit $\# \notin \Sigma$ korrekt?

JA NEIN

[ ] [ ] Aus $L \in \mathsf{NP}$ folgt $L=\{x \in \Sigma^* \mid \exists y \in \{0,1\}^{\leq p(|x|)}: x\#y \in L_0\}$, für ein Polynom $p(n)$ und eine Sprache $L_0 \in \mathsf{P}$.
[ ] [ ] Aus $\overline{L}=\{x \in \Sigma^* \mid \exists y \in \{0,1\}^{\leq |x|^2}: x\#y \in L_0\}$ und $L_0 \in \mathsf{P}$ folgt $L \in \mathsf{co\text{-}NP}$.
[ ] [ ] Aus $\overline{L}=\{x \in \Sigma^* \mid \forall y \in \{0,1\}^{\leq |x|^2}: x\#y \notin L_0\}$ und $L_0 \in \mathsf{P}$ folgt $L \leq_p$ DHC.

(g) Welche der folgenden Aussagen ist/sind korrekt?

JA NEIN

[ ] [ ] MAX-RUCKSACK ist r-approximierbar via Algorithmus Greedy Rucksack, für ein $r>2$.
[ ] [ ] MAX-RUCKSACK ist 2-approximierbar via Algorithmus Maximum Greedy Rucksack.
[ ] [ ] MIN-BINPACKING ist asymptotisch $\frac{3}{2}$-approximierbar via Algo. First Fit Decreasing.

(h) Welche der folgenden Aussagen ist/sind korrekt?

JA NEIN

[ ] [ ] Wenn MIN-BINPACKING 1.3-approximierbar ist, dann gilt $\mathsf{P}=\mathsf{NP}$.
[ ] [ ] Wenn MIN-TSP einen Approximationsalgorithmus besitzt, dann gilt $\mathsf{NP}=\mathsf{co\text{-}NP}$.
[ ] [ ] Wenn MIN-PARTITION kein polynomielles Approximationsschema besitzt, dann gilt $\mathsf{P}=\mathsf{NP}$.

(i) Welche der folgenden Aussagen ist/sind für **alle** NPO-Probleme $\mathcal{P}$ korrekt?

JA NEIN

[ ] [ ] Aus $\mathcal{P}_D \notin \mathsf{NPC}$ folgt $\mathcal{P}_D \notin \mathsf{NP}$.
[ ] [ ] Aus $\mathcal{P}_C \leq_T \mathcal{P}_D$ folgt $\mathcal{P}_D =_T \mathcal{P}_E =_T \mathcal{P}_C$.
[ ] [ ] Aus $\mathcal{P}_C \not\leq_T \mathcal{P}_D$ folgt, daß $\mathcal{P}_D$ nicht NP-schwer ist.

(j) Welche der folgenden Aussagen ist/sind korrekt?

JA NEIN

[ ] [ ] Es gilt $L \in \mathsf{co\text{-}NPC}$ genau dann, wenn $\overline{\text{3-SAT}} \leq_p \overline{L}$ und $L \leq_p$ CLIQUE gilt.
[ ] [ ] Es gilt $\mathsf{NP} \neq \mathsf{co\text{-}NP}$ genau dann, wenn $\overline{\text{TSP}} \notin \mathsf{NP}$ gilt.
[ ] [ ] Wenn eine Sprache L durch keine polynomiell-platzbeschränkte Turingmaschine akzeptiert wird, dann gilt $L \notin \mathsf{PH}$.

Aufgabe 2 (Definitionen von Begriffen aus der Vorlesung) [$12 = 2+2+3+2+2+1$ Punkte]

Im folgenden sei $\mathcal{P} = (D, O, SOL, m, goal)$ ein beliebiges NPO-Problem. Definieren Sie

(a) den Begriff $L \leq_p L'$, für $L \subseteq \Sigma^*$ und $L' \subseteq \Delta^*$.

(b) den Begriff L *ist NP-vollständig.*

(c) den Begriff $A \leq_T B$ (*Problem* A *ist turingreduzierbar auf Problem* B).

(d) den Begriff $\mathcal{A}$ *ist Approximationsalgorithmus für* $\mathcal{P}$.

(e) die *Güte* $r_{\mathcal{A}}(x)$ eines Approximationsalgorithmus $\mathcal{A}$ für $\mathcal{P}$ bzgl. Input $x \in D$.

(f) die Sprachklasse co-NP.

Aufgabe 3 (Sätze bzw. Konstruktionen aus der Vorlesung) [$16 = (5+4) + (2+5)$ Punkte]

(a) Sei $\varphi = C_1 \wedge \ldots \wedge C_r$ eine beliebige 3-KNF-Formel mit Variablen $x_1, \ldots, x_m$. Geben Sie die gemäß Vorlesung definierten Komponenten des Tupels $z_\varphi = (b, a_1, \ldots, a_m, b_1, \ldots, b_m, c_1, \ldots, c_r, d_1, \ldots, d_r)$ von *Oktalzahlstrings der Länge* $r + m$ mit der Eigenschaft

$$\varphi \text{ ist erfüllbar } \iff z_\varphi \in \text{RUCKSACK}^* \text{ (d.h. } \exists \text{ Auswahl aus } \vec{a}, \vec{b}, \vec{c}, \vec{d}, \text{ die sich zu } b \text{ addiert)}$$

an und beweisen Sie davon die Implikation „$\Longleftarrow$“.

(b) Sei $x = (b, a_1, \ldots, a_m) \in \mathbb{N}^{m+1}$ beliebig mit $m \geq 1$. Geben Sie den gemäß Vorlesung definierten Wert $f(x)$ für eine Funktion $f \in \mathsf{FP}$ mit

$$x \in \text{RUCKSACK}^* \iff f(x) \in \text{PARTITION}$$

an und beweisen Sie davon die Implikation „$\Longleftarrow$“.

Aufgabe 4 (Turingreduzierbarkeit) [9 Punkte]

Eine Matrix $A = (a_{ij}) \in \{0,1\}^{m \times n}$ heiße *zulässig*, falls jede Spalte von A mindestens eine Eins enthält.

Das Entscheidungsproblem SET-COVER besteht, informal beschrieben, in folgender Aufgabe: Gegeben eine Matrix $A = (a_{ij}) \in \{0,1\}^{m \times n}$, entscheide, ob eine *Zeilenauswahl* $I \subseteq \{1, \ldots, m\}$ existiert, für die die *Streichungsmatrix* $A(I) \in \{0,1\}^{|I| \times n}$, die aus M durch Streichung aller Zeilen $M[k]$ mit $k \in \{1, \ldots, m\} \backslash I$ entsteht, zulässig ist — ob also $\forall j \in \{1, \ldots, n\}\ \exists i \in I \colon a_{ij} = 1$ gilt.

SET-COVER besitzt die nachfolgenden zwei Varianten.

Konstruktionsvariante SET-COVER$_C$**:** Zu gegebener zulässiger Matrix A berechne eine (bzgl. der Größe) minimale Zeilenauswahl $I \subseteq \{1, \ldots, m\}$, für die $A(I)$ zulässig ist.

Auswertungsvariante SET-COVER$_E$**:** Zu gegebener zulässiger Matrix A berechne die Größe $m^*(A)$ einer minimalen Zeilenauswahl $I \subseteq \{1, \ldots, m\}$, für die $A(I)$ zulässig ist.

Zeigen Sie: SET-COVER$_C$ $\leq_T$ SET-COVER$_E$

Aufgabe 5 (Reduktionsmethode für NPC-Sprachen) [11 = 7 + 4 Punkte]

Das Problem MULTIPROCESSOR-SCHEDULING (MPS) besteht, informal beschrieben, in folgender Aufgabe: Gegeben $n \geq 1$ Aufgaben mit *Bearbeitungszeiten* $t_1, \ldots, t_n$ sowie $m \geq 1$ Prozessoren und eine *Deadline* d, entscheide, ob eine *Verteilung* $r_1, \ldots, r_n \in \{1, \ldots, m\}$ der n Aufgaben auf die m Prozessoren existiert, für die alle Aufgaben in Zeit d durchführbar sind, d.h. es gilt:

$$(*) \qquad \forall h \in \{1, \ldots, m\}\colon \sum_{r_i = h} t_i \leq d$$

Formal erhalten wir die Sprache – wie üblich steht $(n_1, \ldots, n_l)$ für das Wort $\mathrm{bin}(n_1)\#\ldots\#\mathrm{bin}(n_l)$:

$$\text{MPS} = \{(t_1, \ldots, t_n, m, d) \in \mathbb{N}^{n+2} \mid m, n \geq 1,\ \exists \text{ Verteilung } r_1, \ldots, r_n \in \{1, \ldots, m\} \text{ mit } (*)\}$$

Zeigen Sie: MPS ist NP-vollständig.

Aufgabe 6 (Reduktionsmethode für NPC-Sprachen) [12 = 6 + 6 Punkte]

Man betrachte die erfüllende Belegung v mit $v(x_1) = v(x_3) = v(x_4) = 1$, $v(x_2) = 0$ für die 3-KNF-Formel

$$\varphi = (\overline{x_1} \vee x_2 \vee x_4) \wedge (x_1 \vee x_1 \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3} \vee x_4).$$

(a) Geben Sie den gemäß Vorlesung zu φ konstruierten gerichteten Graphen G_φ mit

$$\varphi \text{ ist erfüllbar} \iff G_\varphi \text{ besitzt einen Hamiltonkreis}$$

in graphischer Darstellung an, mit Kennzeichnung der relevanten Komponenten, wie zum Beispiel die Bauteile K_i sowie die x_j- und $\overline{x_j}$-Wege.

(b) Kennzeichnen Sie in G_φ (mit geeigneter Farbe) den gemäß Vorlesung bestimmten Hamiltonkreis, der zur erfüllenden Belegung v von φ gehört.

Aufgabe 7 (Approximationsalgorithmen) [10 Punkte]

Betrachten Sie den folgenden Approximationsalgorithmus LF (Last Fit) für MIN-BINPACKING.

> **Eingabe:** $(a_1, \ldots, a_n, b) \in \mathbb{Q}^{n+1}$ mit $n \geq 1$ und $0 < a_i \leq b$ für $i = 1, \ldots, n$
> **Ausgabe:** Verteilung $r_1, \ldots, r_n \in \{1, \ldots, n\}$ der n Objekte in Bins mit Kapazität b
> **for** $i := 1, 2, \ldots, n$ **do**
> $r_i :=$ Nummer des letzten nichtleeren Bins, in das „Objekt“ a_i noch paßt, falls ein solches Bin existiert, andernfalls sei r_i die kleinste Nummer eines leeren Bins.

Sei $m_{\mathsf{LF}}(x)$ die Anzahl der Bins, die der Algorithmus LF auf Eingabe x benutzt, und $m^*(x)$ die Anzahl der Bins einer optimalen Verteilung für x.

Zeigen Sie: $\frac{m_{\mathsf{LF}}(x)}{m^*(x)} < 2$ für alle Eingaben x.

Viel Erfolg!