Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für Theoretische Informatik
Prof. Dr. Angelika Steger

27. April 2011

Algorithmen und Komplexität (D-MATH)
Musterklausur 2

Kandidat/in:

Name: ....

Vorname: ....

Stud.-Nr.: ....

Ich bezeuge mit meiner Unterschrift, dass ich die Prüfung unter regulären Bedingungen ablegen konnte und dass ich die unten stehenden allgemeinen Bemerkungen gelesen und verstanden habe.

Unterschrift: ....

Allgemeine Bemerkungen und Hinweise:

- Diese Prüfung besteht neben diesem doppelseitigen Deckblatt aus einem beidseitig bedruckten Aufgabenblatt mit insgesamt 6 Aufgaben.
- Als einziges Hilfsmittel sind 10 beidseitig handschriftlich beschriebene A4-Blätter erlaubt.
- Falls Sie während der Prüfung in irgendeiner Weise gestört oder beeinträchtigt werden, melden Sie dies sofort der Aufsichtsperson. Spätere Klagen werden nicht akzeptiert.
- **Schreiben Sie nicht mit Bleistift! Abgaben, die mit Bleistift geschrieben sind, werden nicht bewertet.**
- Alle Mobiltelefone müssen vollständig ausgeschaltet sein.
- **Alle Antworten müssen für den Korrektor verständlich begründet werden. Schreiben Sie die wesentlichen Lösungsgedanken in klaren Sätzen oder Stichworten hin. Unverständliche oder nicht begründete Antworten werden nicht bewertet.**
- Pro Aufgabe ist höchstens eine gültige Version eines Lösungsversuchs zulässig. Streichen Sie ungültige Lösungsversuche klar durch.
- Sie dürfen alle Aufgaben in beliebiger Reihenfolge lösen. Konzentrieren Sie sich jeweils auf eine Aufgabe, aber teilen Sie sich Ihre Zeit ein.
- Abschreiben und sonstige Versuche des Betrugs führen zum sofortigem Ausschluss von der Prüfung und können rechtliche Folgen haben.
- Die Prüfung dauert zwei Stunden. Falls Sie vorzeitig fertig werden sollten, melden Sie sich durch Handaufhalten bei einer der Aufsichtspersonen und verlassen Sie still den Raum. **In den letzten 20 Minuten der Prüfung kann der Raum nicht mehr verlassen werden.**
- **Vergessen Sie nicht, dieses Deckblatt zu unterschreiben, und beschriften Sie jedes abgegebene Blatt mit Ihrem Namen. Das Aufgabenblatt ist mit abzugeben.**

Viel Erfolg!

	Erreichte Punktzahl (maximal)	Visum
1	(20)	
2	(15)	
3	(10)	
4	(20)	
5	(15)	
6	(20)	
Σ	(100)	

Aufgabe 1 (total 20 Punkte)

Welche der folgenden Aussagen treffen zu? Begründen Sie Ihre Antwort.

(a) QuickSort hat eine Worst-Case-Laufzeit von $\mathcal{O}(n \log n)$.

(b) Die Kreisfreiheit in gerichteten Graphen kann in Zeit $\mathcal{O}(|V| + |E|)$ überprüft werden.

(c) Für alle $n \in \mathbb{N}$ kann man n^n in $\mathcal{O}(\log n)$ Schritten im Einheitskostenmass berechnen.

(d) Für alle $n \in \mathbb{N}$ kann man n^n in $\mathcal{O}(\log n)$ Schritten im logarithmischen Kostenmass berechnen.

(e) Jeder Graph $G = (V, E)$ mit $|E| = |V| - 1$ vielen Kanten ist ein Baum.

(f) Für ein Entscheidungsproblem A gilt: $A \in \mathcal{P} \Rightarrow (A \in \text{co-}\mathcal{NP} \ \wedge \ A \in \mathcal{NP})$

(g) $2^{20} \cdot n^2 = \mathcal{O}(n^2)$.

(h) $n \log(n) = o(n \cdot \sqrt{n})$.

(i) $e^{\sin(n)n} = \Omega(1)$.

(j) $\sum_{i=1}^{n} \frac{n^i}{i!} = \mathcal{O}(e^n)$.

(10 · 2 = 20 Punkte)

Aufgabe 2 (total 15 Punkte)

Ein (2,5)-Baum ist ein externer Suchbaum, bei dem alle Blätter dieselbe Tiefe haben. Jeder innere Knoten in einem (2,5)-Baum hat mindestens 2, jedoch maximal 5 Kinder.

Eine Insert-Operation besteht aus zwei Teilen. Im ersten Teil wird ein geeigneter Pfad von der Wurzel zu einem Blatt durchlaufen und der neue Knoten als zusätzlicher Knoten am Vaterknoten dieses Blattes eingefügt. Dadurch kann die Gradbedingung verletzt werden und es muss in einem zweiten Teil der Insert-Operation diese wieder hergestellt werden.

Die Kosten dieses zweiten Teils wollen wir im Folgenden als Rebalancierungskosten bezeichen. (Als Kostenmass betrachten wir, wie oft ein Knoten den Vater wechselt.)

Auch bei einer Delete-Operation wird ähnlich vorgegangen: zunächst wird der Knoten im Baum lokalisiert, dann ageschnitten und verletzte Gradbedingungen gegebenenfalls wieder hergestellt. Die Rebalancierungskosten einer Delete-Operation definieren wir daher analog.

(a) Beweisen Sie folgenden Satz mit Hilfe der amortisierten Analyse:
Startet man mit einem leeren (2,5)-Baum und fügt in diesen n beliebige Elemente ein, so ist die Gesamtzahl der Rebalancierungskosten höchstens $2n$.

(10 Punkte)

(b) Beweisen Sie folgenden Satz mit Hilfe der amortisierten Analyse:
Startet man mit einem leeren (2,5)-Baum und führt auf diesem n Hinzufügungen bzw. Streichungen von Blättern aus, so ist die Gesamtzahl der Rebalancierungskosten höchstens $2n$.

(5 Punkte)

Aufgabe 3 (total 10 Punkte)

Es seien 2 sortierte Arrays $[a_1, \ldots, a_n]$, sowie $[b_1, \ldots, b_n]$ gegeben. Wir nehmen an dass die Elemente a_i und b_i paarweise verschieden sind.

(a) Zeigen Sie, dass man für ein festes Element a_i in Zeit $\mathcal{O}(1)$ entscheiden kann ob dieses das k-grösste aller $2n$ Elemente ist oder nicht.

(5 Punkte)

(b) Entwerfen sie einen Algorithmus, der das k-grösste[1] Element aller $2n$ Elemente in $\mathcal{O}(\log n)$ Schritten findet.

(5 Punkte)

[1] das 1.-grösste ist das Maximum, das 2.-grösste ist das nächstkleinere, usw.

Aufgabe 4 (total 20 Punkte)

Ein Palindrom ist eine Zeichenkette, welche sich von hinten nach vorne gelesen nicht ändert. "ANNA" oder "LAGERREGAL" sind bekannte Beispiele für Palindrome. Natürlich ist es einfach zu überprüfen ob eine gegebene Zeichenkette ein Palindrom ist, aber ist es auch einfach ein Palindrom zu bauen?

Die folgende rekursive Funktion, aufgerufen mit $i = 1$ und $j = n$, berechnet die minimale Anzahl von Zeichen die man in eine gegebene Zeichenkette $S = \langle s_1, s_2, \ldots, s_n \rangle$ einfügen muss um aus dieser ein Palindrom zu bauen.

$$pal(S, i, j) = \begin{cases} 0, & \text{if } i = j, \\ pal(S, i+1, j-1), & \text{if } s_i = s_j, \\ \min(1 + pal(S, i+1, j),\ 1 + pal(S, i, j-1)), & \text{if } s_i \neq s_j. \end{cases}$$

(a) Zeigen Sie, dass $pal(S, 1, n)$ exponentielle Laufzeit in n haben kann.

(3 Punkte)

(b) Schreiben Sie ein dynamisches Programm, welches in quadratischer Zeit die minimale Anzahl einzufügender Zeichen berechnet um aus einer gegebenen Zeichenkette S ein Palindrom zu bauen. Wie viel Speicherplatz benötigt Ihr dynamisches Programm?

(12 Punkte)

(c) Modifizieren/erweitern Sie Ihren Algorithmus so, dass dieser nicht nur die benötigte Anzahl von Zeichen berechnet, sondern auch direkt das entsprechende Palindrom generiert und ausgibt. Wie ändert sich die Laufzeit bzw. der Platzbedarf Ihres Algorithmus?

(5 Punkte)

Aufgabe 5

(total 15 Punkte)

In dieser Aufgabe betrachten wir das folgende Problem: Gegeben sei eine Menge der Grösse n von (x, y)-Koordinaten, welche Positionen in einem öffentlichen Park angeben, an denen Telefonzellen aufgebaut werden sollen. Entwerfen Sie einen Algorithmus, der die Länge der Kabel minimiert, welche benötigt werden um diese Positionen zu einem Netzwerk zu verbinden. Hierzu kann zwischen zwei beliebigen Telefonzellen A und B ein Kabel stets so verlegt werden, dass dessen Länge der euklidischen Distanz zwischen A und B entspricht.

(a) Entwerfen Sie einen Algorithmus, der aus der Vorlesung bekannte Algorithmen als Unterprogramme verwendet darf, und zum Finden einer minimalen Lösung nicht länger als $\mathcal{O}(n^2)$ lange braucht. Wie lange braucht Ihr Algorithmus?

(5 Punkte)

Wir erweitern nun die Problemstellung aus Teilaufgabe (b). Wieder müssen die gegebenen Positionen miteinander vernetzt werden. Jetzt nehmen wir aber an, dass wir k Paare kabelloser Telefonrouter zur Verfügung haben. Stellt man ein solches Paar an zwei Positionen A und B auf, so übernehmen diese die gesamte Kommunikation zwischen diesen beiden Punkten (ohne dabei Kosten zu verursachen).

(b) Entwerfen Sie einen Algorithmus, der die Länge der benötigten Kabel minimiert wenn $k = 1$ ist, man also nur ein Routerpaar aufstellen kann. Ihr Algorithmus muss zum einen bestimmen wo das Routerpaar aufgestellt werden soll und zum anderen welche Paare von Telefonzellen durch ein Kabel verbunden werden müssen.
Beweisen sie die Korrektheit Ihres Algorithmus. Beweisen Sie, dass Ihr Algorithmus nicht länger als $\mathcal{O}(n^2)$ lange benötigt um eine minimale Lösung zu finden.

(6 Punkte)

(c) Modifizieren Sie Ihren Algorithmus so, dass er auch für $k > 1$ in Zeit $\mathcal{O}(n^2)$ eine minimale Lösung findet. Beweisen Sie erneut Laufzeit und Korrektheit.

(4 Punkte)

Aufgabe 6

(total 20 Punkte)

Das Problem ZERO-ONE LINEAR INEQUALITIES ist wie folgt definiert:

Input: Ein System von m linearen Ungleichungen.

$$a_{j_1}z_1 + \ldots + a_{j_n}z_n \geq b_j,$$

wobei $a_{j_i}, b_j \in \mathbb{Z}$ für $1 \leq j \leq m$ und $1 \leq i \leq n$.

Output: JA, wenn eine Lösung für das Ungleichungssystem existiert, sodass $z_i \in \{0, 1\}$ for $i = 1, \ldots, n$; NEIN, sonst.

Zeige, dass das Problem ZERO-ONE LINEAR INEQUALITIES $\mathcal{NP}$-vollständig ist. Dazu darf angenommen werden, dass die folgenden drei Probleme $\mathcal{NP}$-vollständig sind:

1. RUCKSACKPROBLEM

   Input: Eine Kapazität $B \in \mathbb{N}$, ein $K \in \mathbb{N}$ und n Objekte mit Gewichten $w_1, \ldots, w_n \in \mathbb{N}$ und Profiten $p_1, \ldots, p_n \in \mathbb{N}$.

   Output: JA, wenn es eine Teilmenge $I \subseteq [n]$ mit $\sum_{i \in I} p_i \geq K$ und $\sum_{i \in I} w_i \leq B$ gibt.

2. SAT

   Input: Eine boolesche Formel F in KNF.

   Output: JA, wenn es für F eine erfüllende Belegung gibt; NEIN, sonst.

3. MENGENÜBERDECKUNG

   Input: Ein Mengensystem über einer endlichen Grundmenge M, also $T_1, \ldots, T_k \subseteq M$, sowie eine Zahl $n \leq k$.

   Output: JA, wenn man n Mengen $T_{i_1}, \ldots, T_{i_n}$ so wählen kann, dass $\bigcup_{j=1}^{n} T_{i_j} = M$; NEIN, sonst.

(5 + 15 = 20 Punkte)