Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Harald Räcke
Chintan Shah

Summer Semester 2013
Lösungen der Klausur
31st July, 2013

Effiziente Algorithmen und Datenstrukturen II

Last Name	First Name	Matrikel No.
....	....	....

Hall	Seat No.	Signature
....	....	....

General Information for the Examination

- Write your name and matrikel no. on all extra supplementaries (Blätter) provided.
- Please keep your identity card readily available.
- Do not use pencils. Do not write in red or green ink.
- You are not allowed to use any device other than your pens and a single sided handwritten A4 sized paper (with your name clearly written on top).
- Verify that you have received 12 printed sides (check page numbers).
- Attempt all questions. You have 150 minutes to answer the questions.

Left Examination Hall from ...... to ...... / from ...... to ......
Submitted Early at ......
Special Notes:

	A1	A2	A3	A4	A5	A6	A7	A8	Σ	Examiner
Maximum	3	3	4	5	5	10	10	10	50	
1st Correction										
2nd Correction										

Aufgabe 1 (3 Points)

Give formal definitions of a *Basis*, and a *Basic Feasible Solution*.

Aufgabe 2 (3 Points)

(a) Explain why the 2-phase simplex algorithm is needed. (1 point)

(b) Explain the first phase of this algorithm. (2 points)

Aufgabe 3 (4 Points)

Show that there does not exist any FPTAS for the unweighted MAXCUT problem. In the MAXCUT problem, we are given an unweighted graph $G = (V, E)$ and wish to find a partition $U \uplus W$ of V so as to maximize the number of edges having exactly one endpoint in U. It is known that the unweighted MAXCUT problem is strongly NP-Complete.

Aufgabe 4 (5 Points)

Let P be a given feasible, bounded Linear Program. We know how to find the dual D of P. By combining P and D, demonstrate a linear program whose only feasible solution corresponds to the feasible solution optimizing the objective value of P (and similarly for D). The new linear program should have constraints linear in the number of constraints of P and D. Further, ensure that the new linear program is infeasible if either P or D is infeasible.

Aufgabe 5 (5 Points)

Given a directed graph $G = (V, A)$, a special vertex r and a positive cost c_{ij} for each edge $(i, j) \in A$, the minimum-cost arborescence problem is to find a subgraph of minimum cost that contains directed paths from r to all other vertices.

(a) Observe that the following ILP solves the minimum-cost arborescence problem:

$$\begin{array}{lccll} \text{minimize} & \sum\limits_{(i,j)\in A} c_{ij}x_{ij} & & & \\ \text{subject to} & \sum\limits_{i\in S, j\notin S, (i,j)\in A} x_{ij} & \geq & 1 & \forall S \subseteq V, S \ni r \\ & x_{ij} & \in & \{0,1\} & \forall (i,j) \in A \end{array}$$

(b) Show how to efficiently solve the LP obained by relaxing the above ILP.

Aufgabe 6 (10 Points)

We are given a directed graph G on vertex set V, with a nonnegative cost specified for edge $(u \to v)$, for each pair $u, v \in V$. The edge costs satisfy the directed triangle inequality, i.e., for any three vertices u, v, and w, $\text{cost}(u \to v) \leq \text{cost}(u \to w) + \text{cost}(w \to v)$. We need to find a minimum cost cycle visiting every vertex exactly once. Give a $O(\log n)$ approximation algorithm for this problem.

(*Hint*: Use the fact that a minimum cost cycle cover (i.e., disjoint cycles covering all the vertices) can be found in polynomial time. Shrink the cycles and recurse.)

Aufgabe 7 (10 Points)

We are given k stretchable bags $b_1, \ldots, b_k$ and n items $a_1, \ldots, a_n$ with weights $w_1, \ldots, w_n$ and volume $v_1, \ldots, v_n$ respectively, such that $w_i, v_i \leq 1$ and $\sum_{i=1}^{n} w_i = k = \sum_{i=1}^{n} v_i$. We say that a packing of the n items in the k bags is an (α, β)-packing if each bag is filled with weight $\leq \alpha$ and volume $\leq \beta$. Give an efficient algorithm for obtaining a $(3, 3)$-packing.

Lösungsvorschlag

Let

Aufgabe 8 (10 Points)

We are given a graph which is a cycle on n nodes, numbered 0 through $n-1$ clockwise around the cycle. We are also given a set C of calls; each call is a pair (i, j) originating at node i and destined for node j. The call can be routed either clockwise or counter-clockwise around the cycle. The task is to route the calls so as to minimize the *maximum edge load* in the graph. The load L_i on link $(i, (i+1) \mod n)$ is the number of calls routed through $(i, (i+1) \mod n)$, and the maximum edge load is $\max_{0\le i\le n-1} L_i$.

(a) Observe that the following ILP solves the given problem:

$$\begin{array}{lll} \text{minimize} & L & \\ \text{subject to} & x_0^k + x_1^k = 1 & \forall k \in C \\ & \sum_{k\in C} x_{b(k,i)}^k \le L & \forall i \in \{0,\dots,n-1\} \\ & x_0^k \in \{0,1\} & \forall k \in C \\ & x_1^k \in \{0,1\} & \forall k \in C \\ & L \ge 0 & \end{array}$$

where $x_0^k = 1$ indicates that we route the call clockwise, $x_1^k = 1$ indicates that we route the call counter-clockwise, $b(k, i) = 0$ if the clockwise routing uses the link $(i, (i+1) \mod n)$ and $b(k, i) = 1$ if the counter-clockwise routing uses the link $(i, (i+1) \mod n)$.

(b) Relax the above ILP to a linear program and obtain a 2-approximation algorithm for the given problem.

<u>ROUGH WORK</u>