Mastermath and LNMB Course: Discrete Optimization

Solutions for the Exam 30 January 2012

Utrecht University, Buys Ballot Laboratorium, 15:00–18:00

The examination lasts 3 hours. Grading will be done before February 13, 2012. Students interested in checking their results can make an appointment by e-mail (g.schaefer@cwi.nl).

The examination consists of five problems. The maximum number of points to be gained on the different parts are displayed in the following table:

1(a)–(j)	2	3(a)	3(b)	4(a)	4(b)	5	$\sum$
20 (2 each)	15	5	5	15	10	15	**85**

The grade for the exam is obtained by dividing the total number of points by **8.5**. This implies that **47** points are needed to pass.

During the examination only the Lecture Notes of the course without any additional leaflets are allowed to be on your desk and all electronic equipment must be switched off.

Please be short, clear and precise in your answers. If you use results from the Lecture Notes, please provide the respective references.

Good Luck!

Problem 1 (20 points (2 points each))**.** State for each of the claims below whether it is **true** of **false**. Note: You need not justify or prove your answers here.

(a) $17n\log n = \Theta(n^2)$.

(b) Given a spanning tree T of a graph, the number of vertices that have odd degree in T is even.

(c) The symmetric difference $M_1 \triangle M_2$ of two arbitrary matchings M_1 and M_2 exclusively consists of even length cycles and isolated nodes.

(d) If all capacities in the max-flow problem are integers, then there exists a maximum flow that is integral.

(e) Let T be a minimum spanning tree of a graph $G = (V,E)$ with edge cost $c : E \to \mathbb{R}$. By removing an edge $e \in T$ from T, we obtain two trees whose node sets induce a cut $(X,\bar{X})$. Every edge e' that crosses $(X,\bar{X})$ satisfies $c(e') \geq c(e)$.

(f) A pseudoflow that satisfies all capacity constraints is a flow.

(g) If $\Pi_1 \in NP$ and for every problem $\Pi_2 \in NP$, $\Pi_1 \preceq \Pi_2$, then Π_1 is NP-complete.

(h) If $\Pi_1 \in NP$ and $\Pi_2 \preceq \Pi_1$ for some NP-complete problem Π_2, then Π_1 is NP-complete.

(i) The TSP problem in which all edge distances are either 1 or 2 is NP-complete.

(j) Suppose ALG is an α-approximation algorithm for an optimization problem Π whose approximation ratio is tight. Then for every $\varepsilon > 0$ there is no $(\alpha - \varepsilon)$-approximation algorithm for Π (unless $P = NP$).

Solution:

(a) false

(b) true

(c) false

(d) true

(e) true

(f) false

(g) false

(h) true

(i) true

(j) false

Problem 2 (15 points)**.** Consider the *maximum weight indegree-bounded subgraph problem*: We are given a directed graph $G = (V,E)$ with edge weights $w : E \to \mathbb{R}^+$ and degree bounds $b : V \to \mathbb{N}$. The goal is to determine a subset $E' \subseteq E$ of maximum total weight $w(E') = \sum_{e \in E'} w(e)$ such that every node $u \in V$ has indegree at most $b(u)$.

Prove that the greedy algorithm for matroids solves this problem.

Solution: *Define*

$$\mathcal{I} = \{E' \subseteq E \mid \forall v \in V, \text{ the indegree of } v \text{ in the graph } (V,E') \text{ is at most } b(v)\}.$$

Clearly, $\mathcal{I}$ contains all indegree-bounded subgraphs of G with respect to b.

We prove that the pair $(E,\mathcal{I})$ is a matroid. By applying Theorem 3.1 of the Lecture Notes to $(E,\mathcal{I})$ and w, we can then conclude that the greedy algorithm for matroids solves the maximum weight indegree-bounded subgraph problem.

We verify thereto that $(E,\mathcal{I})$ satisfies all properties that define a matroid.

- *E is finite by definition.*
- *$(V,\emptyset)$ is obviously an indegree-bounded subgraph with respect to b.*
- *Let $I \in \mathcal{I}$ and let $J \subset I$. Then (V,I) is an indegree-bounded subgraph with respect to b. Because J is a subset of I, for all nodes $v \in V$ it holds that the indegree of v in (V,J) is at most the indegree of v in (V,I). So (V,J) is an indegree-bounded subgraph with respect to b, and thus $J \in \mathcal{I}$.*
- *Let $I,J \in \mathcal{I}$, $|J| < |I|$. Then (V,I) and (V,J) are both indegree-bounded subgraphs with respect to b. Note that the sum of the indegrees of the vertices in a directed graph equals the number of edges in that graph. Therefore there is a vertex v whose indegree in (V,I) is greater than its indegree in (V,J). So there must be an edge $(u,v) \in I \setminus J$ for some $u \in V$. Adding this edge to J only increments the indegree of v, but still the indegree of v in $(V,J \cup \{(u,v)\})$ is at most the indegree of v in (V,I), which is at most $b(v)$. So $(V,J \cup \{(u,v)\})$ is an indegree-bounded subgraph with respect to b, hence $J \cup \{(u,v)\} \in \mathcal{I}$.*

Problem 3 ($5+5$ points)**.** The *Hitchcock problem* is as follows: We are given a set of m sources $M = \{1,\ldots,m\}$ and a set of n terminals $N = \{1,\ldots,n\}$. Every source $i \in M$ has a supply of $s(i) \in \mathbb{N}$ units and every terminal $j \in N$ has a demand of $d(j) \in \mathbb{N}$ units. We assume that $\sum_{i \in M} s(i) = \sum_{j \in N} d(j)$. The cost to send one unit from source $i \in M$ to terminal $j \in N$ is given by $c(i,j) \in \mathbb{R}^+$. An *allocation* specifies for each pair $(i,j) \in M \times N$ the amount $x(i,j)$ that is sent from i to j. An allocation is *feasible* if it satisfies the supply of every source $i \in M$ and the demand of every terminal $j \in N$. The goal is to compute a feasible allocation x of minimum total cost $\sum_{(i,j) \in M \times N} c(i,j)x(i,j)$.

(a) Formulate the *Hitchcock problem* as an integer linear program and derive the respective LP relaxation.

(b) Show that the set of feasible solutions of this LP is an integral polytope.

Solution:

(a) An integer linear programming formulation for this problem is

$$\begin{array}{lll} \textit{minimize} & \displaystyle\sum_{(i,j)\in M\times N} c(i,j)x(i,j) & \\ \textit{subject to} & \displaystyle\sum_{j\in N} x(i,j) \;\leq\; s(i) & \forall i \in M \\ & \displaystyle\sum_{i\in M} x(i,j) \;\geq\; d(j) & \forall j \in N \\ & x(i,j) \;\in\; \mathbb{N} & \forall (i,j) \in M\times N \end{array} \tag{1}$$

The LP relaxation is

$$\begin{array}{lll} \textit{minimize} & \displaystyle\sum_{(i,j)\in M\times N} c(i,j)x(i,j) & \\ \textit{subject to} & \displaystyle\sum_{j\in N} x(i,j) \;\leq\; s(i) & \forall i \in M \\ & \displaystyle\sum_{i\in M} x(i,j) \;\geq\; d(j) & \forall j \in N \\ & x(i,j) \;\geq\; 0 & \forall (i,j) \in M\times N \end{array} \tag{2}$$

(b) The linear programming relaxation (3) *is equivalent to*

$$\begin{array}{lll} \textit{minimize} & \displaystyle\sum_{(i,j)\in M\times N} c(i,j)x(i,j) & \\ \textit{subject to} & \displaystyle\sum_{j\in N} x(i,j) \;\leq\; s(i) & \forall i \in M \\ & \displaystyle -\sum_{i\in M} x(i,j) \;\leq\; -d(j) & \forall j \in N \\ & x(i,j) \;\geq\; 0 & \forall (i,j) \in M\times N \end{array} \tag{3}$$

Observe that this linear programming formulation is of the form stated in Theorem 8.8 of the Lecture Notes. Also note that the coefficient matrix of this linear program is the incident matrix of a directed graph (which is even complete bipartite). So by Corollary 8.1 of the Lecture Notes, the coefficient matrix is totally unimodular. Therefore, by Theorem 8.8 of the Lecture Notes, the polytope of feasible solutions of the linear program is integral.

Problem 4 ($15+10$ points)**.** In the *longest s,t-path problem* we are given a directed graph $G=(V,E)$ with edge weights $w : E \to \mathbb{R}^+$, a source node $s \in V$ and a target node $t \in V$. The goal is to compute a simple s,t-path P of maximum total weight $w(P) = \sum_{e\in P} w(e)$.

(a) The decision variant of the problem is to determine whether there exists a simple s,t-path of total weight at least K, where K is a given parameter. Show that this problem is NP-complete. (Hint: Use that the *Hamiltonian path problem* is NP-complete: Given an undirected graph G, determine whether G contains a Hamiltonian path.)

(b) Show that the *longest s,t-path problem* in acyclic graphs can be solved in time $O(n+m)$, where n and m refer to the number of nodes and edges of G, respectively.

Solution:

(a) It is clear that the decision problem is in NP: A certificate for an instance of the problem is a subset of the edges that forms an s,t-path of a total weight that exceeds K. Checking whether the total weight of the set of edges exceeds K is obviously possible in polynomial time. Checking whether a set of edges forms a path can be done by verifying whether s and t occur in exactly one of the edges of the edge set, and checking whether all remaining vertices of the graph occur either in 0 or 2 of the edges of the edge set. It is clear that this is a straightforward procedure that can be executed in polynomial time.

We next show that the Hamiltonian path problem is polynomial-time reducible to the longest s,t-path problem. It then follows that the longest s,t-path problem is NP-complete. The reduction works as follows: Let $G=(V,E)$ be an instance of the Hamiltonian path problem. Construct from G the instance $I=(G'=(V',E'),w: E'\to\mathbb{R}^+,K,s,t)$. In this instance, s and t are vertices in $V'\backslash V$, $V'=V\cup\{s,t\}$, and $E'=\{(u,v),(v,u) \mid (u,v)\in E\}\cup\{(s,u),(u,t): u\in V\}$. Moreover w is the function that maps all edges to 1, except those connected to s or t, which are mapped to 0. Lastly, K is set to $|V|-1$.

Obviously, this reduction takes polynomial time: Constructing K is a matter of counting the number of vertices in V, taking $O(n)$ time. Specifying s and t and constructing V' from V takes $O(1)$ time. Constructing E' requires replacing each edge in E by two directed edges, and adding two additional edges to each vertex, so this step takes $O(m+n)$ time.

We prove that G is a yes-instance of the Hamiltonian path problem if and only if I is a yes-instance of the longest s,t-path problem.

($\Leftarrow$) Let $((v_1,v_2),(v_2,v_3),\ldots,(v_{n-1},v_n))$ be a Hamiltonian path for G. Then the simple path $((s,v_1),(v_1,v_2),(v_2,v_3),\ldots,(v_{n-1},v_n),(v_n,t))$ contains $|V|-1$ edges of weight 1, and 2 edges of weight 0, by construction. So it is a simple s,t-path of weight $|V|-1$ for G'. So I is a yes-instance of the longest path problem.

($\Rightarrow$) Suppose that the path $((s,v_1),(v_1,v_2),(v_2,v_3),\ldots,(v_{k-1},v_k),(v_k,t))$ (for some k) is a simple path of weight $|V|-1$. The edges (s,v_1) and (v_k,t) have weight 0, so the path $(v_1,v_2),(v_2,v_3),\ldots,(v_{k-1},v_k)$ must have weight $|V|-1$. Because all edges not connected to s or t have weight 1, it must be that $k=n$. Because $(v_1,v_2),(v_2,v_3),\ldots,(v_{k-1},v_k)$ is a simple directed path of $n-1$ edges in G', by construction it is a simple undirected path of $n-1$ edges in G. In other words, it is a Hamiltonian path in G, so G is a yes-instance of the Hamiltonian path problem.

(b) Define a cost function $c: E\to\mathbb{R}$ as $c=-w$ and consider the graph G with cost function c. Note that (G,c) does not contain any negative cycles because G is acyclic. Now, P is a longest s,t-path in (G,w) iff P is a shortest s,t-path in (G,c). We thus need to compute a shortest s,t-path in (G,c). This can be done in time $O(n+m)$ using the algorithm of Assignment 2 (Problem 1).

Problem 5 (15 points)**.** Consider the *maximum weight acyclic subgraph problem*: We are given a directed graph $G = (V,E)$ with edge weights $w : E \to \mathbb{R}^+$. The goal is to determine a subset $E' \subseteq E$ of maximum total weight $w(E') = \sum_{e \in E'} w(e)$ such that the subgraph $G' = (V,E')$ induced by E' is acyclic.

Derive a 2-approximation algorithm for this problem and show that its approximation ratio is tight. (Hint: Assign a unique number $r(u)$ to every node $u \in V$ of G. An edge $(u,v) \in E$ is a *forward edge* if $r(u) < r(v)$. Prove that the set of all forward edges induces an acyclic subgraph of G.)

Solution: *The algorithm is as follows:*

Input*: A directed graph $G = (V,E)$ with edge weights $w : E \to \mathbb{R}^+$.*
Output*: A subset $E' \subseteq E$ such that $G' = (V,E')$ is acyclic.*

1 *Fix an arbitrary order $u_1, \ldots, u_n$ on the nodes of G and define $r(u_i) = i$.*
2 *Call an edge $(u,v) \in E$ a* forward *edge if $r(u) < r(v)$ and a* backward *edge otherwise. Let F and B be the set of all forward and backward edges, respectively.*
3 *Let $E' = F$ if $w(F) > w(B)$ and $E' = B$ otherwise.*
4 **return** *E'*

We prove that the above algorithm is a 2-approximation for the maximum weight acyclic subgraph problem.

1. *The algorithm has polynomial running time: Defining $r(u)$ for every node $u \in V$ takes $O(n)$ time and identifying all forward and backward edges needs $O(m)$ time. Altogether the algorithm thus needs $O(n+m)$ time.*
2. *The algorithm computes a feasible solution: Suppose for the sake of a contradiction that the subgraph $G = (V,E')$ contains a cycle $C = (v_1, v_2, \ldots, v_k = v_1)$. Because C consists only of forward (or backward) edges, we must have $r(v_1) < r(v_2) < \cdots < r(v_k) = r(v_1)$ (or $r(v_1) > r(v_2) > \cdots > r(v_k) = r(v_1)$), which is a contradiction.*
3. *The algorithm computes a 2-approximate solution: Note that F and B partition the edge set E, i.e., $E = F \cup B$ and $F \cap B = \emptyset$. Also note that $\text{OPT} \leq w(E)$. Because we choose the set of larger weight among F and B, we have*

$$2w(E') \geq w(F) + w(B) = w(E) \geq \text{OPT}.$$

That is, $w(E') \geq \frac{1}{2}\text{OPT}$.

The approximation ratio of is tight as the following example shows: Let $V = \{u_1, u_2, u_3\}$ and assume that this is the order fixed in Step 1 of the algorithm. Further, let $E = \{(u_1,u_3),(u_3,u_2)\}$ and $w(e) = 1$ for both edges. Note that the entire graph is acyclic. Thus the edge set E is the optimal solution with weight $\text{OPT} = w(E) = 2$. The algorithm, however, will output the edge set $E' = \{(u_3,u_2)\}$ with weight $w(E') = 1$.