TU Berlin
Institut für Mathematik
Prof. David Williamson
Jannik Matuschke

Approximation Algorithms (ADM III)
Winter term 2010

Problem Set 3

(due date: *November 24, 2010*)

Exercise 3.1 **5 points**

In the *minimum-cost Steiner tree problem*, we are given as input a complete, undirected graph $G = (V, E)$ with nonnegative costs $c_{ij} \geq 0$ for all edges $(i, j) \in E$. The set of vertices is partitioned into *terminals* T and *nonterminals* (or *Steiner vertices*) $V - T$. The goal is to find a minimum-cost tree containing all terminals.

(a) Suppose initially that the edge costs obey the triangle inequality; that is, $c_{ij} \leq c_{ik} + c_{kj}$ for all $i, j, k \in V$. Let $G[T]$ be the graph induced on the set of terminals; that is, $G[T]$ contains the vertices in T and all edges from G that have both endpoints in T. Consider computing a minimum spanning tree in $G[T]$. Show that this gives a 2-approximation algorithm for the minimum-cost Steiner tree problem.

(b) Now we suppose that edge costs do not obey the triangle inequality, and that the input graph G is connected but not necessarily complete. Let c'_{ij} be the cost of the shortest path from i to j in G using input edge costs c. Consider running the algorithm above in the complete graph G' on V with edge costs c' to obtain a tree T'. To compute a tree T in the original graph G, for each edge $(i, j) \in T'$, we add to T all edges in a shortest path from i to j in G using input edge costs c. Show that this is still a 2-approximation algorithm for the minimum-cost Steiner tree problem on the original (incomplete) input graph G. G' is sometimes called the *metric completion* of G.

Exercise 3.2 **5 points**

In the *edge-disjoint paths* problem in directed graphs, we are given as input a directed graph $G = (V, A)$ and k source-sink pairs $s_i, t_i \in V$. The goal of the problem is to find edge-disjoint paths so that as many source-sink pairs as possible have a path from s_i to t_i. More formally, let $S \subseteq 1, \ldots, k$. We want to find S and paths P_i for all $i \in S$ such that $|S|$ is as large as possible and for any $i, j \in S$, $i \neq j$, P_i and P_j are edge-disjoint ($P_i \cap P_j = \emptyset$).

Consider the following greedy algorithm for the problem. Let ℓ be the maximum of $\sqrt{m}$ and the diameter of the graph (where $m = |A|$ is the number of input arcs and the diameter is $\max_{i,j \in V} d_{ij}$ with d_{ij} being the length of a shortest i-j-path). For i from 1 to k, we check to see if there exists a s_i-t_i path of length at most ℓ in the graph. If there is such a path P_i, we add i to S and remove the arcs of P_i from the graph.

Show that this greedy algorithm is an $\Omega(1/\ell)$-approximation algorithm for the edge-disjoint paths problem in directed graphs.

Exercise 3.3 **5 points**

Suppose that an undirected graph G has a Hamiltonian path. Give a polynomial-time algorithm to find a path of length at least $\Omega(\log n/(\log\log n))$.

Exercise 3.4 **5 points**

Consider the following greedy algorithm for the knapsack problem. We initially sort all the items in order of nonincreasing ratio of value to size so that $v_1/s_1 \geq v_2/s_2 \geq \cdots \geq v_n/s_n$. Let i^* be the index of an item of maximum value so that $v_{i^*} = \max_{i \in I} v_i$. The greedy algorithm puts items in the knapsack in index order until the next item no longer fits; that is, it finds k such that $\sum_{i=1}^{k} s_i \leq B$ but $\sum_{i=1}^{k+1} s_i > B$. The algorithm returns either $1, \ldots, k$ or i^*, whichever has greater value. Prove that this algorithm is a 1/2-approximation algorithm for the knapsack problem.