Antonios Antoniadis and Marvin Künnemann **Winter 2018/19**

Exercises for Randomized and Approximation Algorithms

www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter18/rand-apx-algo/

Exercise Sheet 6: Linear Programming, Deterministic Rounding

To be handed in by **November 27th, 2018** via e-mail to André Nusser (CC to Antonios Antoniadis and Marvin Künnemann)

Exercise 1 *(5 Points)* In the lecture we presented a deterministic rounding algorithm for the *metric uncapacitated facility location problem*, and proved it to be a 6-approximation algorithm. In the analysis, we defined the neigborhood of a demand $j \in D$ as $N_j = \{i \in F | c_{ij} \leq 2D_j^*$ and $x_{ij}^* > 0\}$. Can you slightly adapt the definition of neigborhood in order to obtain a 4-approximation algorithm for the problem?

Exercise 2 *(7 Points)* Consider a *ring network* consisting of n vertices that form a cycle and are numbered 0 through $n-1$ clockwise around the cycle. A set of packets C arrives; each packet is a pair (i, j) meaning that it originates at vertex i and has destination on vertex j. The packet can be routed either clockwise or counterclockwise along the cycle, and the objective is to route all the packets so as to minimize the total *load* on the network. The load L_i on an edge $(i, i+1 (\text{mod} n))$ is the number of packets routed through this edge, and the total load is $\max_{1 \leq i \leq n} L_i$.

Give a 2-approximation algorithm for the ring network problem.

Exercise 3 *(15 Points)* Consider the *bounded max-cut problem*, which is defined as the max-cut problem, in which in addition to the graph $G = (V, E)$ and the edge weights $w_e, \forall e \in E$ we are given a parameter $k \leq |V|/2$. Similarily to max-cut we seek an $S \subseteq V$ of maximum weight but with the additional constraint that $|S| = k$.

(i) Prove that the following *nonlinear* integer program models the problem:

$$\begin{array}{ll} \text{maximize} & \sum_{(i,j) \in E} w_{ij}(x_i + x_j - 2x_i x_j) \\ \text{subject to} & \sum_{i \in V} x_i = k, \\ & x_i \in \{0, 1\}, \quad \forall i \in V. \end{array}$$

(3 Points)

(ii) Prove that the following linear program is a relaxation of the problem:

$$
\begin{array}{lrl}
\text{maximize} & \sum_{(i,j)\in E} w_{ij} z_{ij} & \\
\text{subject to} & z_{ij} \le x_i + x_j, & \forall (i,j)\in E \\
& z_{ij} \le 2 - x_i - x_j, & \forall (i,j)\in E \\
& \sum_{i\in V} x_i = k, & \\
& 0 \le z_{ij} \le 1, & \forall (i,j)\in E, \\
& 0 \le x_i \le 1, & \forall i \in V.
\end{array}
$$

(3 Points)

(iii) Let $F(x) = \sum_{(i,j)\in E} w_{ij}(x_i + x_j - 2x_i x_j)$ be the objective function from the nonlinear integer program. Show that for any (x, z) that is a feasible solution to the linear programming relaxation, $F(x) \ge \frac{1}{2}\sum_{(i,j)\in E} w_{ij} z_{ij}$. *(3 Points)*

(iv) Argue that given a fractional solution x, for two arbitrary fractional variables x_i and x_j, it is possible to increase one by $\epsilon > 0$ and decrease the other one by ϵ such that $F(x)$ does not decrease and one of the variables becomes integer. *(3 Points)*

(v) Use the arguments above to devise a $\frac{1}{2}$-approximation algorithm for the problem. *(3 Points)*

Exercise 4 *(13 Points)* We say that an *extreme point* x of a linear program is a feasible solution that cannot be expressed as $\lambda x^1 + (1-\lambda)x^2$ for $0 < \lambda < 1$ and feasible solutions x^1, x^2 distinct from x.

Consider the *vertex cover problem* and the following linear program for it:

$$
\begin{array}{lrl}
\text{minimize} & \sum_{i\in V} w_i x_i & \\
\text{subject to} & x_i + x_j \ge 1, & \forall (i,j)\in E \\
& x_i \ge 0, & i \in V
\end{array}
$$

(i) Prove that every extreme point of the above linear program has $x_i \in \{0, \frac{1}{2}, 1\}$, for all $i \in V$. *(5 Points)*

(ii) Give a 3/2-approximation algorithm for the vertex cover probem when the input graph is planar (A *planar graph* is a graph that can be drawn on the plane so that its edges intersect only at their endpoints). You may use the fact that polynomial-time LP-solvers return extreme points and that there is a polynomial-time algorithm to 4-color any planar graph. (4-coloring a graph, means to assign to each vertex one of four colors, so that for any edge $(i,j) \in$ vertices i and j are assigned different colors.) *(8 Points)*