Advanced Algorithms 2024 Deadline: 28/10/2024, 11:59 pm

Graded Homework 1

Lecturer: Bernhard Haeupler, Johannes Lengler, Maximilian Probst

Note 1: *Solutions must be typeset in LaTeX and uploaded via moodle by 11:59 pm on October 28th 2024 (please use a separate pdf file for each problem). Late submissions will not be graded. If you would like to add a drawing in your solution, you can simply include a picture of a hand-drawn figure in your LaTeX. You can submit only one file for each problem and the drawing should be incorporated in the rest of your explanation.*

Note 2: *This is a theory course and any claim should be substantiated with a proof. When asked to give an algorithm, the algorithm has to have polynomial running time, unless less efficient algorithms are explicitly permitted by the problem statement. You are required to prove the claimed properties of any algorithm you present.*

Note 3: *You can discuss the problems with the other students but you should write your own solutions independently, and you should be able to orally explain your submitted solution to the instructors, if asked. It is strictly prohibited to share any (hand)written or electronic (partial) solutions with fellow students. Moreover, if you discussed a problem with another student, you must list their names on your submitted solution.*

1 Covering Cliques (20 points)

Given an undirected, unweighted graph $G = (V, E)$, a family $\mathcal{S} = \{V_1, \dots, V_\ell\}$ of polynomially many k-subsets of V (i.e., $V_i \subseteq V$ and $|V_i| = k$), and an integer $s \leq \binom{k}{2}$, find a smallest possible set of edges such that $\mathcal{S}$ contains at least s edges within each V_i. In other words, find $F \subseteq E$ of smallest possible cardinality such that for every $V_i \in \mathcal{S}$, $|\binom{V_i}{2} \cap F| \geq s$.

1. Formulate this problem as an integer linear program (6 points).
2. Give an $\mathcal{O}(k^2)$-approximation algorithm to this problem (14 points).

Note: You may assume $|\binom{V_i}{2} \cap E| \geq s$ for all $V_i \in \mathcal{S}$, or equivalently that $F = E$ is a feasible solution.

2 Graph Coloring (20 points)

You are given a graph $G = (V, E)$. Each of the edges is colored with exactly one of the colors blue, red and violet. You want to color the vertices with red and blue, and you receive a score for your coloring as follows:

- For every blue and red edge you get a point if both endpoints are blue or red, respectively.
- For violet edges, you get a point if one endpoint is blue and one is red.

Design a polynomial time randomized algorithm that outputs w.h.p. (i.e. with probability $1-o(1)$) a 3-approximation, i.e., it outputs a solution whose score is at least $\frac{1}{3}$OPT, where OPT is the score of an optimal coloring.

Note: If you design a randomized algorithm which has an *expected* score of at least $\frac{1}{3}$OPT, you get up to 12 points. If you design for every $\varepsilon > 0$ a randomized algorithm which runs in time $\text{poly}(n, 1/\varepsilon)$ and which outputs w.h.p. a $(3+\varepsilon)$-approximation, you get up to 16 points.

3 Item Distribution (30 points)

You are given a **tree** $T = (V, E)$ on n vertices, a set of items I ($|I| \leq n$) with positive integer sizes size(i), and a *neighbour size bound* h.
You can give each vertex of the graph up to one item (and cannot give the same item to multiple vertices). Your goal is to give out **as many items as possible**, such that for every vertex, the sum of sizes of items in its neighbours is at most h. More specifically, let the size of a vertex be 0 if it received no item and size(i) where i is the item it received otherwise. Then, it must hold that for every vertex v, $\sum_{u \in N(v)} \text{size}(u) \leq h$.

1. Suppose that for some constant C, there are at most C distinct sizes among items (that is, there exists a set S of cardinality at most C such that size(i) $\in S$ for all $i \in I$). Give a polynomial-time algorithm that solves the problem exactly in this special case (15 points).

2. Give a PTAS for the problem (15 points).

   Note. Recall that a PTAS for a maximization problem is an algorithm that for any fixed constant $\epsilon > 0$, in polynomial time (where the degree of the polynomial can depend arbitrarily on ϵ), finds a solution of value at least $(1 - \epsilon)$OPT. Here, this means an assignment of at least $(1 - \epsilon)$OPT items to vertices that respects the neighbour size bound.

For part 2, you may assume the result of part 1, even if you did not complete that part.

Note: The neighbourhood $N(v)$ of v does not include v itself.

4 Rental Problem (30 points)

You're in a winter sport area where every day the slopes are all open either only to skiers, or only to snowboarders, and this is determined at the beginning of that day. You would like to do the activity which is possible every day. For that, you need to rent and/or buy skis and a snowboard in a way that minimizes your cost. You always have the option to rent skis or a snowboard for cost 1 on any given day. Alternatively, you could buy skis or a snowboard for a positive real cost $X \geq 1$ (the skis and the snowboard happen to cost the same), and then you will be able to use them until the end of the season. You also have an option of buying both the skis and the snowboard at once for a bundle cost of Y ($X \leq Y \leq 2X$).

1. Show that no deterministic online algorithm for the problem can be $(1 + \sqrt{2} - \varepsilon)$-competitive for any constant $\varepsilon > 0$ (15 points).

2. Show that there is a $(1 + \sqrt{2})$-competitive deterministic online algorithm (15 points).

Note: An online algorithm doesn't know in advance how long the input sequence is. The parameters X and Y are part of the input and known to the online algorithm from the start. A deterministic online algorithm is α-competitive, if for every $1 \leq X \leq Y \leq 2X$ and finite input sequence S of elements from $\{\textit{skis}, \textit{snowboard}\}$, the cost of the algorithm on the sequence is at most $\alpha \cdot \text{OPT}(S)$, where OPT($S$) is the offline optimum for the sequence: the minimum possible cost when the entire sequence is known in advance.