Approximation Algorithms (02360521) - HW1

Submission deadline: 23.12.2025

Guidelines:

- The homework can be submitted either in singles or in pairs.
- You are required to provide complete and formal proofs of your claims.
- Assignments should be typed and may not exceed 20 pages (using font size at least 11).
- You may consult others as necessary, but write up your own solutions independently - plagiarism of any kind is not allowed.

1. **15pt.** Design a 2-approximation algorithm for the problem of finding a minimum cardinaliy maximal matching in an undirected graph.
2. **20pt.** Consider the greedy algorithm for Vertex Cover which at every step adds to the cover the vertex which covers the largest number of uncovered edges. Prove that the algorithm does not achieve approximation ratio better than $\Omega(\log n)$, where n is the number of vertices in the input graph.
3. In class we saw an H_n-factor greedy algorithm for the minimum weighted set cover problem.

   (a) **10pt.** Give a tight example in which $w(I) = H_n \cdot OPT - \epsilon$ for all $\epsilon > 0$. (I is the cover that the greedy algorithm we saw in class outputs).

   Consider the following variant of (unweighted) Set Cover: Given a set of elements E, m subsets of the elements $S_1, \ldots, S_m \subseteq E$ (such that $\bigcup_j S_j = E$) and an integer $k \leq m$, select k subsets whose union has maximum cardinality.

   (b) **5pt.** Show that this variant of the *Set Cover* problem is NP-hard.

   (c) **20pt.** Give a $\left(1 - \frac{1}{e}\right)$-approximation algorithm for this problem.
   Hint: recall that $\left(1 - \frac{1}{n}\right)^n < e^{-1}$ for every $n \in \mathbb{N}$.
   Note: for maximization problems, we say that an algorithm is α-approximation for $\alpha < 1$ if the value of the solution returned by the algorithm is at least α times the value of the optimal solution.
4. In the *minimum-cost Steiner tree* problem we are given an undirected graph $G = (V, E)$ with non-negative costs c_{ij} for all edges $(i, j) \in E$. The set of vertices is partitioned into *terminals* R and *non-terminals* $V \setminus R$. The goal is to find a minimum-cost tree containing all the terminals.

(a) **20pt.** Suppose initially that G is complete and the edge costs obey the triangle inequality, i.e., $c_{ij} \leq c_{ik} + c_{kj}$ for all $i, j, k \in V$. Consider computing a minimum spanning tree of $G[R]$ ($G[R]$ is the subgraph of G induced by the terminals R). Prove that this gives a 2-approximation algorithm for the minimum-cost Steiner tree problem.

(b) **10pt.** Now suppose that edge costs do not obey the triangle inequality, and that the input graph G is connected but not necessarily complete. Let $\widetilde{c}_{ij}$ be the distance of the shortest path from i to j in G. Consider running the algorithm above on the complete graph $\widetilde{G}$ on V with edge costs $\widetilde{c}_{ij}$, to obtain a tree $\widetilde{T}$.[1] To compute a tree T in G, for each edge (i, j) in $\widetilde{T}$ we add to T all edges in a shortest path from i to j in G. If the resulting graph T has a cycle, arbitrarily remove a single edge from the cycle, and repeat the process until T is acyclic. Show that this is still a 2-approximation algorithm for the minimum-cost Steiner tree problem on the original (incomplete) input graph G.

[1] $\widetilde{G}$ is sometimes called the metric completion of G.