

Computationally Hard Problems – Fall 2020 Assignment 2

Date: 08.09.2020, **Due date:** 14.09.2020, 21:00

The following exercises are **not** mandatory:

Exercise 2.1: Show how to convert a decision algorithm A_d for the following problem into a polynomial-time optimization algorithm A_o . The input to the decision problem consists of an undirected graph $G = (V, E)$ and a positive integer k . The decision algorithm A_d answers YES if there is a set $V' \subseteq V$ of cardinality k such that $\forall v, w \in V': \{v, w\} \notin E$. This problem is called INDEPENDENTSET since there are no edges between the vertices in V' .

We now want to solve the optimization problem, that is, we are only given the undirected graph G and want to find in it an independent set V' of maximum cardinality.

- a) Describe an algorithm A_o which solves the optimization problem as described above. The running time of A_o has to be polynomial in the input size and A_o may make calls to A_d . Recall that such a call counts as one step.
- b) Argue that your algorithm is correct.
- c) Show the polynomial running time of the algorithm.

End of Exercise 1

Exercise 2.2: Consider the following decision problem.

Problem: [CLIQUEORNOT]

Input: An undirected graph $G = (V, E)$.

Output: YES if G contains a clique and NO otherwise.

Discuss in how far this decision problem is useful for finding cliques of maximal size in G .

End of Exercise 2

Continued on next page.

Exercise 2.3: Consider the following problem.

Problem: [MINIMUMTESTSET]

Input: You want to test an electronic circuit. The circuit can have n different errors, e_1, \dots, e_n . You can perform m different tests T_1, \dots, T_m . Every test can detect some, but not necessarily all, errors.

Output for the optimizing version: A minimum set of tests that can detect all possible errors.

1. Formulate the above problem in an appropriate way as a decision problem. Show that an algorithm for your decision problem can be used in order to solve the above optimization problem. The resulting optimization algorithm should have polynomial running time when a call of the decision algorithm is counted as one step. Argue that your algorithm is correct and show the running time of the algorithm.
2. We suggest the following algorithm for selecting a minimum test set: Select a test T_i which can detect a maximum number of errors. Continue by selecting tests which can detect a maximum number of errors from those that previously selected tests could not detect.

Prove or disprove that the output of this algorithm is a minimum test set.

End of Exercise 3

Exercise 2.4: Prove Proposition 2.14 in the lecture notes, which is also repeated below, and determine the constants c_p and c_e . You may assume that the running time for input size n is n^a for a positive constant $a > 0$ in the case of polynomial running time, and b^n for a positive constant $b > 1$ in the case of exponential running time.

Proposition 2.14: Suppose an algorithm has been implemented on some machine. With the current hardware, this can solve problems up to size n in a fixed time t_0 . Now suppose that the speed of the hardware is doubled. If the running time of the algorithm is polynomial then there is a constant $c_p > 1$ such that one can now solve problems of size $c_p \cdot n$ in time t_0 . If the running time of the algorithm is exponential then there is a constant $c_e > 0$ such that one can now solve problems of size $c_e + n$ in time t_0 .

End of Exercise 4

Continued on next page.

The following exercise is **mandatory**:

Exercise 2.5: We consider the following problem.

Problem: [REFUTATION] Given is a disjunctive form consisting of k monomials m_1, \dots, m_k over n boolean variables x_1, \dots, x_n (as on Exercise Sheet 1). The task is to decide if there is a truth assignment to the variables such that the truth value of the disjunctive form is *false*.

You are given a decision algorithm A_d that solves this problem, i. e., for each instance to REFUTATION, A_d answers YES if there is an assignment that makes the truth value of the disjunctive form *false*; otherwise it answers NO.

- a) Describe an algorithm A_o which solves the optimization problem, i. e., which finds a truth assignment making the disjunctive form *false* if one exists. The running time has to be polynomial in the input size and A_o may make calls to A_d . Such calls count as one basic computational step.
- b) Argue that your algorithm is correct.
- c) Prove that the running time of the algorithm is bounded from above by a polynomial. Any polynomial is sufficient; you need not look for a polynomial of minimal degree. Recall that a call to A_d counts one step.

Note: The input to REFUTATION is a disjunctive form of monomials over n boolean variables, nothing else. In particular, a legal input to the problem cannot specify specific settings of variables. For example, you cannot simply say “set x_1 to true” and then call the decision algorithm.

End of Exercise 5
