

# Exercise Sheet 1 for Advanced Algorithms, Summer 2025

**Hand In:** Until 2025-05-21 18:00, on ILIAS.

## Exercise Policy

- The exercise problems take up the material from lecture. Working on them deepens understanding and increases proficiency, and I consider them an integral part of the course.
- You will require at least 50% of the total points to take the exam.
- You may collaborate on the problems and hand in your solutions as a group of 2–4 students.
- If you make use of external sources, make sure to clearly mark the used parts as such, including unambiguous references to the original sources. In case of websites, at least add the access date and use permanent links whenever possible (e. g., for Wikipedia). You may cite any publicly accessible sources, but take their credibility into account: you are responsible to verify what you cite.

This applies even more so to generative AI answers. The tasks are designed to be solved without AI tools; don't deprive yourself of the learning experience.

Collaboration between groups on conceptual solutions to the exercises is fine, but each group must work out a detailed submission individually. In these cases, add a quick statement with which groups you joined forces to work on a given problem.

Plagiarism is not tolerated in academia, be it external sources, other groups' submissions, or genAI answers; severe cases can lead to expulsion from university.

**Problem 1**

20 + 30 points

Consider the following problems. Prove that they are NP-complete, respectively.

You may use that the standard problems 3SAT, Hamilton Path, Clique, Knapsack, Subset Sum, Vertex Cover and Traveling Salesperson are NP-complete.

a) **Longest Path:**

**Input:** A graph  $G = (V, E)$  with edge weights  $w : E \rightarrow \mathbb{N}$ , nodes  $s, t \in V$  and  $k \in \mathbb{N}$ .

**Question:** Is there a simple path from  $s$  to  $t$  in  $G$  that has weight – w. r. t.  $w$  – at least  $k$ ?

b) **Scheduling with Missed Deadlines:**

**Input:** A set  $T = \{t_1, \dots, t_n\}$  of jobs, each with duration 1, deadlines  $d : T \rightarrow \mathbb{N}$ , a partial order  $\prec$  on  $T$  and a natural bound  $k \leq n$ .

**Question:** Is there a mapping  $\sigma : T \rightarrow \{1, 2, \dots, n\}$  which schedules the jobs according to the following conditions?

i) Only one job is scheduled per time slot, that is

$$i \neq j \implies \sigma(t_i) \neq \sigma(t_j)$$

for all  $i, j \in [1..n]$ .

ii) Schedule  $\sigma$  is *happens-before* consistent with  $\prec$ , that is

$$t_i \prec t_j \implies \sigma(t_i) < \sigma(t_j)$$

for all  $i, j \in [1..n]$ .

iii) Not too many jobs are late, that is

$$|\{t \in T \mid \sigma(t) > d(t)\}| \leq k.$$

**Advice:** Remember what you know from basic courses! What are the necessary steps?

**Problem 2**

20 + 30 points

Design a solution algorithm for the following search problems using a polynomial number of calls to a decider for the corresponding decision problem, e.g., if the search problem under consideration is SAT, assume there is a procedure `boolean isSAT( $\phi$ )` that checks if a given formula  $\phi$  is satisfiable or not. Use calls to `isSAT` to compute a satisfying variable assignment for  $\phi$ , if  $\phi$  is satisfiable, or output “No-instance” otherwise.

Counting calls to the decider as a single instruction, your algorithm must run in polynomial time.

## a) SAT:

Given a boolean formula  $\phi$  over variables  $x_1, \dots, x_n$  (not in any specific normal form, but given in a reasonable encoding<sup>1</sup> of your choice), compute a satisfying assignment  $v : [n] \rightarrow \{0, 1\}$  of the variables if one exists, or output “not satisfiable”.

## b) Hamilton-Cycle:

Given an undirected (unweighted) graph  $G = (V, E)$  (without self-loops and parallel edges), compute a Hamiltonian cycle (i.e., a simple closed path visiting each vertex in  $V$  exactly once) if one exists, or output “not Hamiltonian” otherwise.

---

<sup>1</sup>i.e., with length of order  $\log n$  times the total number of variable occurrences and operators in  $\phi$ .