Complexity exam

June 24, 2022

This exam consists of four problems.

Problem 1 (15 points)

1. Merge sort splits the input array in two parts, and applies itself recursively to these two parts. If we modify merge sort to split the array into three parts, will this result in a better asymptotic runtime? Briefly explain why it does, or doesn't.
2. Bob is assigned a scheduling problem by his boss that he thinks is too difficult to solve efficiently. Having followed a complexity course, Bob shows the problem reduces to SAT, and argues that it therefore cannot be solved efficiently. Are you convinced? Briefly explain your position.
3. Two methods, A and B, are used within a small music streaming company to index files. Insertion of a new file takes $T_{\mathrm{A}}(n) = n$ microseconds under A, and $T_{\mathrm{B}}(n) = 100 \cdot \sqrt{n}$ microseconds under B, where n is the number of files already present.
    (a) Which method is asymptotically faster?
    (b) Which method would you recommend to the company for indexing their employment contracts?
    (c) Which method would you recommend for indexing their music catalogue?

Problem 2 (40 points) Let $T\colon \mathbb{N} \longrightarrow [0, \infty)$ be given with

$$T(n) \;=\; T(\lceil n/2 \rceil) \;+\; T(\lceil n/4 \rceil) \qquad \text{for all } n \in \{2, 3, \ldots\}.$$

(Note: to get the full marks for this problem, you should deal with ceilings and base cases of induction proofs rigorously.)

1. Show that $T(n) = \mathcal{O}(n)$.

From this point onward, you may assume that T is monotone, i.e., $n \leq m \implies T(n) \leq T(m)$ for all $n, m \in \mathbb{N}$.

2. Show using induction that $T(n) \leq S(n)$ for all $n \in \mathbb{N}$, where the function $S\colon \mathbb{N} \to [0, \infty)$ is recursively defined by, for all $n \in \mathbb{N}$,

$$S(n) \;=\; \begin{cases} T(n) & \text{if } n < 2 \\ 3S(\lceil n/4 \rceil) & \text{if } n \geq 2. \end{cases}$$

3. Show that $T(n) = \mathcal{O}(n^{\log_4(3)})$.
4. Show that $T(n) = \mathcal{O}(n^{\ln(\varphi)})$, where $\varphi = \frac{1}{2} + \frac{1}{2}\sqrt{5}$. (You may use the fact that $a_k = \Theta(\varphi^k)$, where a_k is the k-th Fibonacci number.)

Problem 3 (25 points) In this exercise, we look at the Knapsack problem.

1. Find three subsets I of $\{1, 2, 3, 4, 5\}$ such that

$$\sum_{i \in I} w_i \leq 5 \qquad \text{and} \qquad \sum_{i \in I} v_i \geq 12,$$

where $w_1 = w_2 = w_3 = 1$, $w_4 = w_5 = 2$, $v_1 = v_2 = 3$, $v_3 = 4$, and $v_4 = v_5 = 5$.

2. Show that the following decision problem, known as the 0–1-Knapsack problem, is **NP**-complete:

> Given $k, V, W, v_1, \ldots, v_k, w_1, \ldots, w_k \in \mathbb{N}$, is there a subset I of $\{1, \ldots, k\}$ with $\sum_{i \in I} w_i \leq W$ and $\sum_{i \in I} v_i \geq V$?

Problem 4 (20 points) Consider the following situation. We have a finite set S of students and a finite set C of courses, and a relation $R(s, c)$ that indicates whether student s follows course c. Courses can be scheduled in three time slots: 10:45–12:30, 13:45–15:30 and 15:45–17:30.

A **schedule** assigns to every course a time slot, and a schedule is called **good** if no student has a scheduling conflict.

Show that the following problem is **NP**-complete.

> Given S, C and R as before, is there a good schedule?

Hint: use 3Color.