Complexity exam

with solutions

June 24, 2022

This exam consists of four problems.

Grading. Let $p \in \{0, 1, \ldots, 100\}$ denote the number of points earned. Then the 'raw' grade $b \in [1, 10]$ for this exam is $b = 1.5 + p \cdot 8.5/100$. The actual, rounded grade g including the bonus for the weekly assignments is computed as follows.

> Letting $a \in [0, 10]$ denote the average of the six weekly assignments, we have $g = \mathrm{round}(b')$, where
>
> $$b' = \begin{cases} \min(10,\ b + a/10) & \text{if } b > 5 \\ b & \text{if } b \leq 5 \end{cases}$$
>
> and $\mathrm{round}(b')$ is the number (or the greatest of the two numbers) in $\{1.0,\ 1.5,\ \ldots,\ 9.5,\ 10.0\,\}\backslash\{5.5\}$ nearest to b'.

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?

Solution. 1. The runtime of merge sort can be computed using the Master theorem: since merge sort recurses twice, $a = 2$, on inputs of essentially half the size, $b = 2$, whose results take linear time to combine, $f(n) = \Theta(n)$, the runtime will be $\Theta(n^{\log_b(a)} \log(n))$. Splitting the array into three parts changes a and b to 3, but leaves $\log_b(a) = 1$ and $\Theta(f)$ unchanged. (Merging three sorted arrays can be done in linear time by comparing the heads of the lists and popping the least until the arrays are exhausted.)

Thus, by splitting the input array into three parts, no asymptotic speedup is gained.

2. No, that the scheduling problem, let us denote it by SCHED, reduces to SAT, that is, SCHED $\leq_P$ SAT, does not imply that SCHED is NP-complete, or difficult in any sense (for any $\mathbf{P}$-problem reduces to SAT.)

If instead Bob would have established a reduction in the other direction, SAT $\leq_P$ SCHED, then this would have implied that SCHED is NP-hard, and therefore in general intractable (though specific instances might still be solvable, perhaps using a SAT-solver via Bob's original reduction.)

3. (a) Method B is asymptotically faster.

(b) Although method B is asymptotically faster, method A is faster when $n < 10000$, for then $T_{\mathrm{A}}(n) = n \text{ ms} = \sqrt{n} \cdot \sqrt{n} \text{ ms} < \sqrt{10000} \cdot \sqrt{n} \text{ ms} = 100 \cdot \sqrt{n} \text{ ms} = T_{\mathrm{B}}(n)$. Since the number of employment contracts for our small company will likely not exceed 10000, method A is from an efficiency standpoint more appropriate.

Grading. Full points are awarded for the reasoning presented here, or any other convincing argument. One might, for example, argue that the reduction in maintenance cost when using just one method at the company (method B) outweighs any gains from using method A instead of method B for employment contracts.

(c) Since a music catalogue is likely to contain more than 10000 files, method B is to be recommended.

■

Grading. **5 points** are given for every part.

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

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

(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., $n \le m \implies T(n) \le T(m)$ for all $n, m \in \mathbb{N}$.

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

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

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.)

Solution. 1. Choose $C > 0$ such that $T(n) \le Cn$ for all $n \in \{1,2,\dots,7\}$ (for example, by defining $C := \max_{n\in\{1,\dots,7\}} T(n)/n$.)

We will prove by strong induction that $T(n) \le Cn$ for all $n \in \{1,2,\dots\}$. Let $n \in \{1,2,\dots\}$ with — the induction hypothesis — $T(m) \le Cm$ for all $m \in \{1,2,\dots,n-1\}$ be given; we must show that $T(n) \le Cn$. If $n < 8$, then $T(n) \le Cn$ by choice of C. On the other hand, if $n \ge 8$, then:

$$\begin{aligned} T(n) &= T(\lceil n/2\rceil) \,+\, T(\lceil n/4\rceil) \\ &\le C(\lceil n/2\rceil \,+\, \lceil n/4\rceil) && \text{by the induction hypothesis} \\ &\le C(n/2+1 \,+\, n/4+1) && \text{since } \lceil x\rceil \le x+1 \\ &= C(3n/4+2) \\ &= Cn \,-\, C(n/4-2) \\ &\le Cn && \text{because } n \ge 8. \end{aligned}$$

Whence $T(n) \le Cn$ for all $n \in \{1,2,\dots\}$. Thus $T \in \mathcal{O}(n)$.

Common errors. (a) Implicitly using the incorrect estimate $\lceil x\rceil \le x$ (instead of e.g. $\lceil x\rceil \le x+1$.)

(b) Making statements such as "for $n = 2$, we have $T(2) = 2T(1) = \mathcal{O}(n)$" where the n in $\mathcal{O}(n)$ is a concrete number instead of a variable.

(c) Trying to prove that $T(n) \le Cn$ holds for all $n \in \mathbb{N}$ instead of for sufficiently large n.

(d) Trying to deduce $T(n+1) \le C(n+1)$ from $T(n) \le Cn$.

(e) Not being clear about the exact induction hypothesis.

(f) Forgetting to address all base cases. For example, not realising for which n the induction step works, and for which n the estimate must be established manually.

(g) Applying the recurrence relation for T to $n = 0$ or $n = 1$.

(h) Giving an intuitive argument instead of a rigorous proof. E.g., only drawing a recursion tree and claiming that $T(n) = \mathcal{O}(n)$ follows.

(i) Assuming that $T(1) = 1$ without justification.

(j) Assuming that T is monotone without justification.

2. We prove that $T(n) \leq S(n)$ using strong induction. Let $N \in \mathbb{N}$ with $T(n) \leq S(n)$ for all $n < N$ be given; we must show that $T(N) \leq S(N)$.

   If $N < 2$, then $T(N) = S(N)$, by definition of S, so suppose that $N \geq 2$. Then we have

$$\begin{aligned} T(N) &= T(\lceil N/2 \rceil) + T(\lceil N/4 \rceil) \\ &= T(\lceil \lceil N/2 \rceil /2 \rceil) + T(\lceil \lceil N/2 \rceil /4 \rceil) + T(\lceil N/4 \rceil) \end{aligned}$$

   by applying the recurrence relation for T twice.

$$= 2T(\lceil N/4 \rceil) + T(\lceil N/8 \rceil)$$

   using the formula $\lceil x/b \rceil = \lceil \lceil x \rceil /b \rceil$ for $x \in \mathbb{R}$ and $b \in \{1, 2, \dots\}$ (established in Exercise 1 of exercise set #2.)

$$\leq 3T(\lceil N/4 \rceil)$$

   since $\lceil N/8 \rceil \leq \lceil N/4 \rceil$, and T is monotone.

$$\leq 3S(\lceil N/4 \rceil) = S(N)$$

   by the induction hypothesis (since $\lceil N/4 \rceil < N$.)

   Whence $S(n) \leq T(n)$ for all $n \in \mathbb{N}$.

   Common errors. (a) Not addressing all base cases.

   (b) Using an estimates such as $3T(\lceil n/4 \rceil) \geq T(n)$ and $3T(\lceil n/4 \rceil) \geq T(\lceil n/2 \rceil) + T(\lceil n/4 \rceil)$ without explanation.

   (c) Trying to deduce $T(n+1) \leq S(n+1)$ from $T(n) \leq S(n)$ instead of using strong induction.

3. The recurrence relation of S is of the form $aS(\lceil n/b \rceil) + f(n)$, where $a = 3$, $b = 4$, and $f(n) = 0 = \mathcal{O}(n^0)$. Since $\log_4(3) > 0$, case I of the Master theorem applies to S, and yields $S(n) = \Theta(n^{\log_4(3)})$. Since $T(n) \leq S(n) = \mathcal{O}(n^{\log_4(3)})$, we get $T(n) = \mathcal{O}(n^{\log_4(3)})$.

   Common errors. (a) Stating that $S(n) = \mathcal{O}(n^{\log_4(3)})$ without explanation.

(b) When using the Master theorem, not mentioning which of the three cases applies.

(c) Statements such as "$T(n) = \mathcal{O}(...)$ for $n \geq 2$". (The "$n \geq 2$" makes no sense.)

4. Let $a_0, a_1, \ldots$ denote the Fibonacci numbers, so $a_0 = 0$, $a_1 = 1$, and $a_{k+2} = a_{k+1} + a_k$. Then by repeatedly applying the recurrence relation for T, one finds that, for all $n \in \mathbb{N}$ and $k \in \mathbb{N}$ with $\lceil n/2^{k-1} \rceil \geq 2$, we have

$$T(n) \;=\; a_{k+1} T(\lceil n/2^k \rceil) \;+\; a_k T(\lceil n/2^{k+1} \rceil). \tag{1}$$

Indeed, for $k = 0$, (1) holds, because $T(n) = 1 \cdot T(n) + 0 \cdot T(\lceil n/2 \rceil)$.

Now suppose that for some $k \in \mathbb{N}$ (1) holds for all $n \in \mathbb{N}$ with $\lceil n/2^{k-1} \rceil \geq 2$ (the 'induction hypothesis'), and let $n \in \mathbb{N}$ with $\lceil n/2^k \rceil \geq 2$ be given; we claim that (1) holds too for n and $k+1$. To see this, note that

$$T(n) \;=\; a_{k+1} T(\lceil n/2^k \rceil) \;+\; a_k T(\lceil n/2^{k+1} \rceil),$$

by the induction hypothesis, which applies since $\lceil n/2^{k-1} \rceil \geq \lceil n/2^k \rceil \geq 2$.

$$= \; (a_k + a_{k+1}) T(\lceil n/2^{k+1} \rceil) \;+\; a_{k+1} T(\lceil n/2^{k+2} \rceil),$$

by substituting $T(\lceil n/2^k \rceil) = T(\lceil \lceil n/2^k \rceil /2 \rceil) + T(\lceil \lceil n/2^k \rceil /4 \rceil) = T(\lceil n/2^{k+1} \rceil) + T(\lceil n/2^{k+2} \rceil)$, using here that $\lceil n/2^k \rceil \geq 2$.

$$= \; a_{k+2} T(\lceil n/2^{k+1} \rceil) \;+\; a_{k+1} T(\lceil n/2^{k+2} \rceil),$$

by the recurrence relation for a_k, and so (1) holds for $k+1$ too.

Given $n \in \mathbb{N}$ with $n \geq 2$, define $k(n) := \lfloor \ln(n) \rfloor$. We have

$$\lceil n/2^{k(n)-1} \rceil \geq n/2^{\lfloor \ln n \rfloor - 1} \geq n/2^{\ln n - 1} = 2,$$

so (1) applies to $k(n)$ and n. Moreover, we have

$$\lceil n/2^{k(n)+1} \rceil \;\leq\; \lceil n/2^{k(n)} \rceil \;\leq\; n/2^{\lfloor \ln n \rfloor} + 1 \;\leq\; n/2^{\ln n - 1} + 1 = 3,$$

so $T(\lceil n/2^{k(n)} \rceil)$ and $T(\lceil n/2^{k(n)+1} \rceil)$ are in $\mathcal{O}(1)$.

Note that from the fact that $a_k = \Theta(\varphi^k)$ one easily deduces that $a_{k(n)} = \Theta(\varphi^{k(n)})$, because $k \colon \{2, 3, \ldots\} \to \mathbb{N}$ has the property that for every $K \in \mathbb{N}$ there is $N \in \{2, 3, \ldots\}$ such that for all $n \geq N$, $k(n) \geq K$.

Whence $a_{k(n)} = \Theta(\varphi^{k(n)}) = \Theta(\varphi^{\lfloor \ln(n) \rfloor}) = \Theta(\varphi^{\ln(n)})$. Since similarly, $a_{k(n)+1}$ and $a_{k(n)+2}$ are in $\Theta(\varphi^{\ln(n)})$, and by the previous observation that $T(\lceil n/2^{k(n)} \rceil)$ and $T(\lceil n/2^{k(n)+1} \rceil)$ are in $\mathcal{O}(1)$, Equation (1) yields that $T(n) = \mathcal{O}(\varphi^{\ln(n)})$.

Since $\varphi^{\ln(n)} = 2^{\ln(n)\ln(\varphi)} = n^{\ln(\varphi)}$, we get $T(n) = \mathcal{O}(n^{\ln(\varphi)})$. ■

Common errors. (a) Only considering n of the form 2^k where $k \in \mathbb{N}$.

Grading. **10 points** are given per part.

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

Solution. Three subsets that satisfy the requirements of 1 are

$$\{1, 4, 5\}, \{2, 4, 5\}, \{3, 4, 5\}$$

To show that this problem is **NP**-complete, we first show that it's in **NP**. A certificate here is a set I, so it is a list of natural numbers. We can verify a solution as follows:

1. Check whether every element in I is in $\{0, \dots, k\}$
2. Add all the v_i where $i \in I$ and check whether their sum is at least V
3. Add all the w_i where $i \in I$ and check whether their sum is at most W

Each of these steps can be done in polynomial time, and thus verification can be done in polynomial time.

To show that it's **NP**-hard, we show how to reduce SubsSum to this problem. Suppose, that $S \subseteq \mathbb{N}$ is a finite subset of natural numbers and let $t \in \mathbb{N}$. Write $S = \{s_1, \dots, s_k\}$, and define

- $w_i = s_i$
- $v_i = s_i$
- $W = t$
- $V = t$

Note that this construction is done in polynomial time.

Suppose we have a subset I of S such that

$$\sum_{i \in I} w_i \leq W \quad \text{and} \quad \sum_{i \in I} v_i \geq V$$

If we recall the definitions, then we observe

$$\sum_{i \in I} s_i \leq t \quad \text{and} \quad \sum_{i \in I} s_i \geq t$$

Hence, we have $\sum_{i \in I} s_i = t$, and thus this is a solution of the subset problem.

Suppose, we have a subset I of S such that

$$\sum_{i \in I} s_i = t$$

Hence,

$$\sum_{i \in I} s_i \leq t \quad \text{and} \quad \sum_{i \in I} s_i \geq t$$

If we recall the definitions, then we get

$$\sum_{i \in I} v_i \leq V \quad \text{and} \quad \sum_{i \in I} w_i \geq W$$

Hence, we have the desired reduction. ■

Grading. Points are given as follows

- If the subsets given in 1 satisfy the requirements, **5 points** are given.
- **5 points** are given for the proof that it is NP.
- **10 points** are given for the correct function for the reduction
- **5 points** are given for proof that it is indeed a reduction

At most 10 points are given if the Knapsack problem is used for the reduction. The Knapsack problem was not one of the problems from the lectures/assignments. If the reduction is done in the wrong way around (for example, showing that the problem reduces to SAT), then **at most 10 points** are given. **3 points** are deducted if there's not enough detail.

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.

Solution. A certificate to this problem is a schedule, so pairs of courses and time slots. To verify whether it is good, we check for every student whether they have a scheduling conflict. This can be done in polynomial time, so it is in **NP**.

To show that it is **NP**-hard, we show that 3Color reduces to this problem. Given a graph G, define

- C is the set of vertices of G

- S is the set of edges of G
- We say that $R(s, c)$ for $s \in S$ and $c \in C$ if the node c lies on the edge s.

Time slots represent colours, for example 10:45-12:30 is red, 13:45-15:30 is blue, and 15:45-17:30 is purple. If we have a 3-colouring of G, we get a good schedule for the C, S, and R described above. The time slot assigned to a course is the color assigned to the corresponding vertex. In addition, if we have a good schedule for the C, S, and R described above, we get a 3-colouring of G. The colours assigned to a vertex are the time slots assigned to the corresponding course. Hence, we constructed the desired reduction. ■

Grading. Points are given as follows

- **5 points** are given for the proof that it is NP.
- **10 points** are given for the correct function for the reduction
- **5 points** are given for proof that it is indeed a reduction