Complexity Resit Exam

with solutions

July 13, 2022

This exam consists of four problems. Put your answers on the lined paper. Your solutions are judged not only on correctness, but also clarity. Good luck!

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 $b = 1.25 + p \cdot 8.75/100$. 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 $g = \text{round}(b')$, where
>
> $$b' = \begin{cases} \min(10,\ b + a/10) & \text{if } b > 5 \\ b & \text{if } b \leq 5 \end{cases}$$
>
> and $\text{round}(b')$ 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 (20 points)

1. Two students are quibbling over the complexity of integer multiplication: the one says it's $\mathcal{O}(n^2)$, while the other claims it's $\mathcal{O}(n^{\log_2(3)})$. Who is right?

   Solution. They are both right, because integer multiplication can be performed in $\mathcal{O}(n^{\log_2(3)})$ time (using Karatsuba's algorithm), and thus in $\mathcal{O}(n^2)$ time too.

   Common errors:

   (a) Claiming only one of the two students is correct (perhaps by either forgetting the existence of Karatsuba's algorithm, or by overlooking the difference between Θ and $\mathcal{O}$.)

   (b) Offering no explanation.

   (c) Drawing no conclusion (but, for example, only mentioning the complexity classes of the standard and Karatsuba's algorithm.)

2. Suppose you are consulted by a company that wishes to improve an algorithm operating on large datasets that currently has running time

$$T(n) \;=\; 2\,T(\lceil 5/8 \cdot n \rceil) \;+\; 256n. \tag{1}$$

After some consideration, you find two mutually exclusive methods, "A" and "B", to improve the algorithm's runtime:

A. reduces the factor 256 in (1) to 32.

B. reduces the factor $5/8$ in (1) to $3/8$.

Which method yields the best asymptotic runtime? Which method will you pursue?

Solution. Writing T_A and T_B for the runtime after application of "A" and "B", respectively, we have $T_A(n) = \Theta(n^{\log_{8/5}(2)})$ and $T_B(b) = \Theta(n)$, by cases I and III of the Master Theorem, respectively, because $\log_{8/3}(2) < \log_2(2) \equiv 1 < \log_{8/5}(2)$.

Whence method B yields the best asymptotic runtime.

Given the limited information here I would pursue method B, because for large enough inputs method B will be faster than method A, (and there is no indication that 'large datasets' would not be able to reach this cross-over point.)

Common errors:

(a) Claiming that A has the better asymptotic complexity.

(b) Informal reasoning (about e.g. the 'number of steps'.)

(c) Missing explanation.

(d) Computing T_A and T_B, but drawing no conclusion.

(e) Unclear use of the Master Theorem. (The details matter: if methods A and B would both have fallen in case III, then A would have been the better choice.)

3. Is the variation on SAT, where the formulae are built from variables using the logical connectives $\wedge$ and $\vee$ (but not $\neg$) **NP**-complete? Briefly explain why.

Solution. This variation of SAT is in **P** (because a formula built from just variables and the binary logical connectives $\wedge$ and $\vee$ is always satisfiable, by assigning true to all variables,) and thus **NP**-complete iff **NP** = **P**.

(If we interpret $\wedge$ and $\vee$ not necessarily as binary operations, and thus our formulae may include false, 0, being the empty conjunction, then no longer are all formulae satisfiable, but whether one is can still be computed in polynomial time, as follows. Note that if a formula is satisfiable with some assignment of truth values to variables, then changing all assignments to true does not alter the truth value of the formula — $\wedge$ and $\vee$ unlike $\neg$ are 'monotone' — and thus a formula is satisfiable iff it is by the assignment of true to all its variables. Checking this can be done in polynomial time, as we already know from the proof that SAT is in **NP**.)

Common errors:

(a) Claiming that the problem is **NP**-complete (without proving that **NP** = **P**.)

(b) Incorrect explanation.

4. Many problems can be encoded as integer linear programs, such as finding the maximum flow in a flow network. Someone claims that this means that we have a reduction from integer linear programming to the max-flow problem, and that therefore the max-flow problem is NP-complete. Are they correct?

Solution. No, their reasoning is incorrect, because that a problem A reduces to an **NP**-complete problem does not imply that A is **NP**-complete as well, only that A is in **NP**.

Common errors:

(a) Confusing the direction of the reduction, and as a result claiming, for example, that max-flow is **NP**-hard.

(b) Claiming that **NP** is not shown to be in **NP**.

(c) No or insufficient explanation.

Grading. **5 points** per part.

Problem 2 (30 points) Write down a concrete asymptotic solution — for example, $T(n) = \Theta(n \lg(n))$ — for each of the following recurrence relations.

1. $T(n) = 25\,T(\lceil n/5 \rceil) + n^3$ for $n \geq 2$
2. $T(n) = 125\,T(\lceil n/5 \rceil) + n^3$ for $n \geq 2$
3. $T(n) = 625\,T(\lceil n/5 \rceil) + n^3$ for $n \geq 2$
4. $T(n) = 9\,T(\lceil n/3 \rceil + 2) + n$ for $n \geq 5$
5. $T(n) = T(\lceil n/2 \rceil) + 3\,T(\lfloor n/7 \rfloor) + n$ for $n \geq 2$
6. $T(n) = \sum_{k=1}^{\infty} T(\lfloor n/3^k \rfloor)$ for $n \geq 3$, and $T(0) = 0$

(You may assume that $T(n) > 0$ for all $n > 0$.)

Solution. 1. Note that $\log_5(25) = 2 < 3$, so we get $T(n) = \Theta(n^3)$, by case III of the Master Theorem. (We have already seen that n^3 obeys the regularity condition, for example, in exercise 2 of exercise set #2.)

   Common error: not mentioning regularity.

2. Since $\log_5(125) = 3$, we get $T(n) = \Theta(n^3 \log(n))$, by case II of the Master Theorem.

3. $T(n) = \Theta(n^4)$, by case I of the Master Theorem since $\log_5(625) = 4 > 3$.

   Common errors for 1–3:

   (a) Not noting the case of the Master Theorem or the value of $\log_b(a)$.

4. Defining $S\colon \mathbb{N} \to \mathbb{R}$ by $S(n) = T(n+3)$ for all $n \in \mathbb{N}$, we get

$$\begin{aligned} S(n) &= T(n+3) \\ &= 9T(\lceil (n+3)/3 \rceil + 2) + n + 3 \\ &= 9T(\lceil n/3 + 1 \rceil + 2) + n + 3 \\ &= 9T(\lceil n/3 \rceil + 1 + 2) + n + 3 \\ &= 9S(\lceil n/3 \rceil) + n + 3. \end{aligned}$$

   Whence $S(n) = \Theta(n^2)$ by case I of the Master theorem since $\log_3(9) = 2 > 1$ and $n + 3 = \Omega(n)$. Thus $T(n) = S(n-3) = \Theta((n-3)^2) = \Theta(n^2)$.

   Common error: attempting to apply the Master Theorem directly to T, which is not possible.

5. We'll show that $T(n) = \Theta(n)$ To begin, since $T(n) \geq n$ for all $n \in \mathbb{N}$, we have $T(n) = \Omega(n)$.

   For the other direction, pick $C \geq 24$ such that $T(n) \leq Cn$ for all $n \in \{1, \ldots, 23\}$ (such as $C := \max\{\, 24,\ T(1),\ T(2)/2,\ \ldots,\ T(23)/23\,\}$.) We'll prove that $T(n) \leq Cn$ for all $n \geq 1$, by strong induction. So let $n \in \mathbb{N}$ with $T(m) \leq Cm$ for all $m < n$ with $m \geq 1$ be given. If $n = 0$, there's nothing to prove. If $0 < n < 24$, then $T(n) \leq Cn$ by choice of C. If $n > 24$, we have:

$$\begin{aligned} T(n) &= T(\lceil n/2 \rceil) + 3T(\lfloor n/7 \rfloor) + n \qquad \text{since } n \geq 1 \\ &\leq C\lceil n/2 \rceil + 3C\lfloor n/7 \rfloor + n \end{aligned}$$

   by the I.H., which can be applied since $0 < \lceil n/2 \rceil, \lfloor n/7 \rfloor < n$,

$$\leq C(n/2 + 1) + 3Cn/7 + n$$

   since $\lceil n/2 \rceil \leq n/2 + 1$ and $\lfloor n/7 \rfloor \leq n/7$,

$$\begin{aligned} &= Cn(13/14 + 1/n + 1/C) \\ &\leq Cn, \end{aligned}$$

   because $n \geq 24$ and $C \geq 24$ entail that $1/n + 1/C \leq 1/14$. Whence $T(n) = \mathcal{O}(n)$.

   Common errors:

(a) Not dealing with the base cases correctly, or at all.

(b) Trying to apply the Master Theorem to T, which is not possible.

6. Note that for $n \geq 3$ we have

$$
\begin{aligned}
T(n) &= \textstyle\sum_{k=1}^{\infty} T(\lfloor n/3^k \rfloor) \\
&= T(\lfloor n/3 \rfloor) + \textstyle\sum_{k=2}^{\infty} T(\lfloor n/3^k \rfloor) \\
&= T(\lfloor n/3 \rfloor) + \textstyle\sum_{k=1}^{\infty} T(\lfloor (n/3)/3^k \rfloor) \\
&= T(\lfloor n/3 \rfloor) + \textstyle\sum_{k=1}^{\infty} T(\lfloor \lfloor n/3 \rfloor /3^k \rfloor) \\
&= T(\lfloor n/3 \rfloor) + T(\lfloor n/3 \rfloor) \\
&= 2T(\lfloor n/3 \rfloor),
\end{aligned}
$$

and so $T(n) = \Theta(n^{\log_3(2)})$ by case I of the Master Theorem.

Common error: claiming that $T(n) = 0$ for all n.

Common errors for 1–6:

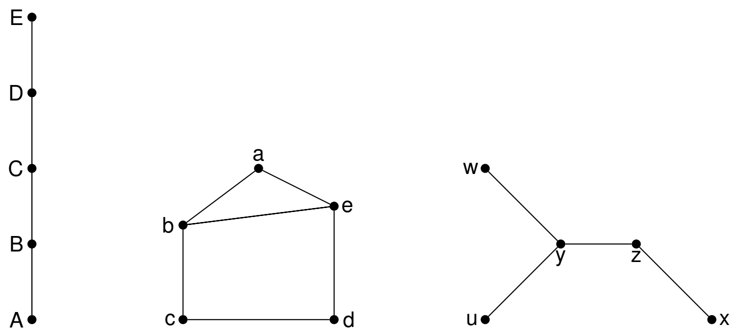
1. Giving an asymptotic solutions without explanation.

2. Giving an asymptotic bound such as $\mathcal{O}(n^2)$ instead of a concrete solution (using Θ.)

Grading. **5 points** per part.

Problem 3 (25 points) A **subgraph** of G is a graph G' whose vertices and edges are subsets of the vertices and edges of G respectively. We say that two graphs G_1 and G_2 are **isomorphic** if there is a bijective function $f : G_1 \to G_2$ such that we have an edge from x to y if and only if we have an edge from $f(x)$ to $f(y)$.

1. Is G_1 isomorphic to a subgraph of G_2? And of G_3?



G_1 G_2 G_3

2. Show that the following problem, known as Subgraph, is **NP**-complete:

   Given finite graphs G_1 and G_2, is G_1 isomorphic to a subgraph of G_2?

Solution. Note that Subgraph is in **NP**. A certificate is a map between the two graphs and we can check in polynomial time is an isomorphism to some subgraph of G_2.

To show that Subgraph is **NP**-hard, we show that Ham reduces to this problem. Let G be a graph with n vertices. Define

- $G_2 = G$
- G_1 is the graph with vertices $\{1, \ldots, n\}$ and edges from i to $i+1$ for $i < n$.

Note that this can be construct in polynomial time relative to G.

Suppose, that G has a Hamiltonian path $v_1, \ldots, v_n$. Note that this path gives rise to a subgraph P of G. This subgraph is isomorphic to G: assign the vertex 1 to the vertex v_1. This assignment is surjective, because it hits every vertex in the Hamiltonian path. Since vertices are crossed precisely once in Hamiltonian paths, this is also injective.

Now suppose that G_1 is isomorphic to some subgraph of G. Denote this isomorphism by f. Then the image of G_1 under this isomorphism is a Hamiltonian path in G. Since f is injective and has n nodes, all nodes in G are in the image. In addition, there must be an edge between all those nodes because of how G_1 was construct.

All in all, we can conclude that Ham $\leq_P$ Subgraph, and thus Subgraph is **NP**-hard.

Grading. Points are given as follows

- **5 points** are given for correctly identifying which graphs are isomorphic
- **5 points** are given for the proof that it is NP.
- **5 points** are given for the correct function for the reduction
- **5 points** are given for proof that it is indeed a reduction
- **5 points** are given for explaining why the reduction is polynomial

Problem 4 (25 points) In this problem, we look at coloring problems. More specifically, we define the problem kColor for every natural number k to be:

Given a finite graph G, does G have a k-coloring?

If $k = 3$, then this is the problem 3Color which we discussed during the lectures. (Note: in this exercise you may use the fact that 3Color is **NP**-complete.)

1. Is the problem 2Color in **P**? Briefly explain why.
2. Give a graph with 5 nodes and give a 4-coloring of that graph.
3. Show that for every k with $k \geq 3$ the problem kColor is **NP**-complete.

Solution. We can determine whether a graph G has a 2-coloring using breadth first search. Starting from an arbitrary vertex v in G, we perform breadth first search starting at v. Nodes on an even layer get the color blue while nodes on an odd layer get the color red. This algorithm runs in polynomial time, and thus 2Color is in **P**.

Next we show that kColor is **NP**-complete. Note that kColor is in **NP**: the certificate is just an assignment of colors to vertices. We can check in polynomial time whether this actually is a coloring: we just look for every edge whether the endpoints get the same color.

To show that kColor is **NP**-hard, we show that 3Color $\leq_P$ kColor. We show two ways of doing that.

Approach 1: There are two cases: either $k = 3$ or $k > 3$. If $k = 3$, then we are just looking at 3Color of which we already know that it's **NP**-complete. If $k > 3$, we can perform the following construction:

1. Start with a graph G
2. Let $m = k - 3$.
3. Add m new vertices $v_1, \ldots, v_m$ to G
4. Connect each v_i to every other vertex in G (including the other v_j)
5. This gives a graph G'

Note that a 3-coloring of G is the same as a k-coloring of G'. Also note that G' can be constructed in polynomial time.

If we have a 3-coloring of G, then we obtain a k-coloring of G' by assigning every v_i a unique color. Since we added $k - 3$ nodes, this is possible.

Suppose, that we have a k-coloring of G'. Each v_i added to G' gets a color and because they are connected to every other vertex in G', every v_i gets a different color. Since we added $k - 3$, there are 3 colors remaining for G, and thus we have a 3-coloring of G.

Approach 2: Using induction, we prove that for all k we have kColor $\leq_P$ $k+1$Color. From this, we can conclude that 3Color $\leq_P$ kColor for all $k \geq 3$.

Let k be a natural number and let G be a graph. We construct a graph G' from G by adding a new vertex v and we connect that vertex with every vertex in G. Note that G' can be constructed in polynomial time from G. By contraction, a coloring of G' must assign the vertex v a different color than every other vertex in G'.

If G has a k-coloring, then we obtain a $(k + 1)$-coloring of G' by assigning the vertex v the remaining color. If G' has a $(k + 1)$-coloring, then we obtain a k-coloring of G by leaving out the vertex v.

Grading. Points are given as follows

- **5 points** are given for the proof that 2Color is in **P**
- **5 points** are given for giving a graph that has 5 nodes and a 4-coloring.
- **5 points** are given for the reason why kColor is in **NP**.
- **10 points** are given for the correct reduction.