Computability and Complexity: Problem Set 2

Due: Friday March 15 at 23:59 AoE.

Submission: Please submit your solutions via *Absalon* as a PDF file. State your name and e-mail address close to the top of the first page. Solutions should be written in LaTeX or some other math-aware typesetting system with reasonable margins on all sides (at least 2.5 cm). Please try to be precise and to the point in your solutions and refrain from vague statements. Make sure to explain your reasoning. *Write so that a fellow student of yours can read, understand, and verify your solutions.* In addition to what is stated below, the general rules for problem sets stated on the course webpage always apply.

Collaboration: Discussions of ideas in groups of two to three people are allowed—and indeed, encouraged—but you should always write up your solutions completely on your own, from start to finish, and you should understand all aspects of them fully. It is not allowed to compose draft solutions together and then continue editing individually, or to share any text, formulas, or pseudocode. Also, no such material may be downloaded from or generated via the internet to be used in draft or final solutions. Submitted solutions will be checked for plagiarism. You should also clearly acknowledge any collaboration. State close to the top of the first page of your problem set solutions if you have been collaborating with someone and if so with whom. *Note that collaboration is on a per problem set basis, so you should not discuss different problems on the same problem set with different people.*

Reference material: Some of the problems are "classic" and hence it might be possible to find solutions on the Internet, in textbooks or in research papers. It is not allowed to use such material in any way unless explicitly stated otherwise. Anything said during the lectures or in the lecture notes, or any material found in Arora-Barak, should be fair game, though, unless you are specifically asked to show something that we claimed without proof in class. All definitions should be as given in class or in Arora-Barak and cannot be substituted by versions from other sources. It is hard to pin down 100% watertight, formal rules on what all of this means—when in doubt, ask the main instructor.

Grading: A total score of 60 points will be enough for grade 02, 90 points for grade 4, 120 points for grade 7, 150 points for grade 10, and 180 points for grade 12 on this problem set. Any revised versions of the problem set with clarifications and/or corrections will be posted on the course webpage `jakobnordstrom.se/teaching/CoCo24/`.

Questions: Please do not hesitate to ask the instructors or TA if any problem statement is unclear, but please make sure to send private messages when using Absalon—sometimes specific enough questions could give away the solution to your fellow students, and we want all of you to benefit from working on, and learning from, the problems. Good luck!

1 (10 p) We say that a language $L \subseteq \{0,1\}^*$ is *sparse* if there is a polynomial p such that it holds for every $n \in \mathbb{N}^+$ that $\left|L \cap \{0,1\}^n\right| \leq p(n)$. Show that if L is sparse, then $L \in \mathsf{P/poly}$.

2 (10 p) Let $\bigoplus_n : \{0,1\}^n \to \{0,1\}$ be the function that computes the XOR of all its input bits. Show that $\bigoplus_n$ has a Boolean circuit of size $O(n)$.

3 (30 p) Show that the function that takes as input two n-bit numbers in binary and outputs their product is computable in $O(n)$ space.

4 (40 p) Show that the language $L \subseteq \{0,1,\#\}^*$ described below can be decided by a deterministic Turing Machine running in $O(\log\log n)$ space. (We can also encode this language L as a language of strings only made up of 0s and 1s, but it is simpler with a slightly bigger alphabet.) For any $i \geq 1$, let $b(i)$ denote the unique binary expansion of i (using $\lceil \log_2(i+1) \rceil$ bits). With this notation, we define L to be the language

$$L = \{b(1)\#b(2)\#b(3)\#\cdots\#b(2^m - 1) \mid m \in \mathbb{N}\} \; .$$

Remark: It turns out that this language is optimal in the following sense: The only languages that can be decided in $o(\log\log n)$ space are the ones that can be decided in $O(1)$ space!

5 (30 p) Consider the language

$$\textsc{SpaceBoundedTM} = \left\{\langle M, x, 1^n \rangle \,\middle|\, M \text{ accepts } x \text{ in space } n\right\}$$

where M is a deterministic Turing machine and 1^n denotes a string of ones of length n (as usual). Prove that $\textsc{SpaceBoundedTM}$ is PSPACE-complete from first principles (i.e., prove that $\textsc{SpaceBoundedTM}$ is in PSPACE and that any other language in PSPACE reduces to it).

(In this problem, we assume that all Turing machines have a fixed configuration in terms of alphabet and number of tapes, and that a universal TM for space-bounded computation as in Exercise 4.1 in Arora-Barak can be assumed without proof.)

6 (40 p) Show that Σ_2SAT (defined below) is complete for Σ_2^p under polynomial-time reductions.

Recall (example 5.6 in the textbook or the notes) that this problem is defined as follows. Σ_2SAT consists of all true totally quantified Boolean formulas of the form

$$\exists y_1 \exists y_2 \cdots \exists y_n \forall z_1 \forall z_2 \cdots \forall z_m F(y_1, \ldots, y_n, z_1, \ldots, z_m) \; ,$$

where F is a Boolean formula (not necessarily in CNF).

7 (40 p) In our lectures on Boolean circuits we defined $\mathsf{DTIME}\big(T(n)\big)/a(n)$ as the class of languages decided by Turing machines M running in time $\mathrm{O}\big(T(n)\big)$ that also get a specific advice string $\alpha_n \in \{0,1\}^{a(n)}$ for inputs of size n. We then proved (or at least outlined a proof) that $\mathsf{P/poly} = \bigcup_{c,d \in \mathbb{N}^+} \mathsf{DTIME}\big(n^c\big)/n^d$.

Is it possible to change the definition so that not only the advice string α_n depends on the size of the input, but so that we can also pick different Turing machines M_n for different input sizes (while still maintaining that all running times be bounded by a common polynomial $p(n)$), and prove that $\mathsf{P/poly}$ is equal to the set of languages decided by such sequences of Turing machines $\{M_n\}_{n \in \mathbb{N}^+}$ with advice strings $\{\alpha_n\}_{n \in \mathbb{N}^+}$? Work out the details to show that this alternative definition is just as fine, or give a clear mathematical argument why it seems problematic.

8 (60 p) Show that $\mathsf{EXP} \not\subseteq \mathsf{SIZE}(n^k)$ for any fixed $k \in \mathbb{N}^+$. That is, show that for each fixed k, there is a language that can be decided in exponential time, but not by a circuit family of size $\mathrm{O}(n^k)$.

Hint: On each input length n, use a kind of diagonalization on all circuits of size at most n^k. (Also, just to avoid confusion, we note that $\mathsf{P/poly} = \bigcup_{k \in \mathbb{N}^+} \mathsf{SIZE}(n^k)$, and we do not know that $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$, but the point here is that we are fixing k.)