Computability and Complexity: Problem Set 3

Due: Friday April 5 at 23:59 AoE.
Submission: Please submit your solutions via *Absalon* as a PDF file. State your name and e-mail address at 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. Never just state an an answer, but make sure to also explain your reasoning. *Write so that a fellow student of yours can read, understand, and verify your solutions.* In addition to what is said 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 70 points will be enough for grade 02, 100 points for grade 4, 130 points for grade 7, 160 points for grade 10, and 200 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 (20 p) Under the assumption $\mathsf{NP} \subseteq \mathsf{P/poly}$, describe how to construct a polynomial-size family of circuits $\{C_{m,n}\}_{m,n\in\mathbb{N}^+}$ that take any CNF formula $\phi(x, y) = \phi(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)$ of size m over $2n$ variables and any assignment $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \{0, 1\}^n$ as inputs, and output an assignment $\beta = (\beta_1, \beta_2, \ldots, \beta_n) \in \{0, 1\}^n$ such that it holds that $\phi(\alpha, C_{m,n}(\phi, \alpha)) = \phi(\alpha_1, \alpha_2, \ldots, \alpha_n, \beta_1, \beta_2, \ldots, \beta_n) = 1$ if such a β exists. That is, fill in the details in the construction that we claimed without proof when showing the Karp-Lipton theorem.

Remark: You do not need to provide an exact gate-by-gate specification of the circuits (especially since we believe that $\mathsf{NP} \not\subseteq \mathsf{P/poly}$), but you should describe in reasonable detail what subcircuits you use and how they are glued together. Also, make sure to argue why the size is polynomial.

2 (20 p) This problem demonstrates that "circuits do not really need randomness", or, in other words, that randomness can be replaced by non-uniformity. Your task is to prove the statement that follows below.

Let C be a Boolean circuit with $\vee$-, $\wedge$-, and $\neg$-gates of size $s \geq n$ over the variables $x_1, \ldots, x_n$ and $r_1, \ldots, r_m$. Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function such that for all $x \in \{0,1\}^n$ it holds that if $r \in \{0,1\}^m$ is chosen uniformly at random then

$$\Pr_r[C(x,r) = f(x)] \geq \frac{2}{3} \, .$$

Then there is a Boolean circuit D with $\vee$-, $\wedge$-, and $\neg$-gates of size at most $\mathrm{poly}(sn)$ over the variables $x_1, \ldots, x_n$ such that for all $x \in \{0,1\}^n$ it holds that $D(x) = f(x)$.

3 (20 p) This problem deals with two notions of randomized computation and shows the equivalence between them. If we fix our alphabet to be binary for convenience, then the class RP comprises all languages $L \subseteq \{0,1\}^*$ for which there are a probabilistic Turing machine (PTM) M and a constant $c > 0$ such that for every $x \in \{0,1\}^n$ it holds that

$$\max_R \mathsf{TIME}(M, x, R) \leq c \cdot n^c$$

and

$$x \notin L \Rightarrow \Pr[M(x,R) = 0] = 1 \; ;$$
$$x \in L \Rightarrow \Pr[M(x,R) = 1] \geq \frac{1}{2} \, .$$

The class ZPP comprises all languages $L \subseteq \{0,1\}^*$ for which there are a PTM M and a constant $c > 0$ such that for every $x, R \in \{0,1\}^*$ it holds that

$$\mathsf{TIME}(M, x, R) < \infty \;\;\&\;\; M(x,R) = L(x)$$

and also for every $x \in \{0,1\}^n$ we have that

$$\mathbb{E}_R[\mathsf{TIME}(M, x, R) \leq c \cdot n^c \,].$$

Show that $\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP}$.

4 (30 p) In this problem, we prove a slight improvement of the DeMillo–Lipton–Schwartz–Zippel (DLSZ) lemma. Prove that for every integer $k > 0$, there is a constant $\varepsilon > 0$ such that the following holds: Let $p(x_1, \ldots, x_n)$ for $n > 1$ be an n-variate polynomial of degree d over $\mathbb{Q}$. Assume that (at least) one of the monomials in p contains all variables. Let $S \subseteq \mathbb{Q}$ be of size $|S| = k$ and let $R \in S^n$ be distributed uniformly at random. Then it holds that

$$\Pr_R[p(R) = 0] \leq \frac{d}{k} - \varepsilon \, .$$

5 (60 p) This problem is about connections between hardness and randomness. The solution to the problem shows that given a "hard function" we can use one less bit of randomness. This is a weak form of de-randomization (stronger forms are known). Your task is to prove the statement that follows below.

There is a constant $c > 0$ such that the following holds. Let $g : \{0,1\}^n \to \{0,1\}$ be a function, and let $M(x, R)$ be a PTM with inputs $x \in \{0,1\}^n$ and $R \in \{0,1\}^m$ such that for all $x \in \{0,1\}^n$ it holds that

$$\Pr_R[M(x,R) = g(x)] \geq 0.7$$

and

$$\max_R \mathsf{TIME}(M, x, R) \leq T \ .$$

Let $f : \{0,1\}^{m-1} \to \{0,1\}$ be a function such that for every Boolean circuit C of size at most $c(Tn)^c$ it holds that

$$\Pr_y[C(y) = f(y)] < 0.49 \ ,$$

where y is uniformly distributed in $\{0,1\}^{m-1}$. Then for all $x \in \{0,1\}^n$ it holds that

$$\Pr_y[M(x,(y,f(y))) = g(x)] \geq 0.6 \ .$$

Hint: Assume towards contradiction that the conclusion is false for some $x \in \{0,1\}^n$. For how many values of y do we have $M(x,(y,f(y))) = g(x)$? For how many values of y do we have $M(x,(y,1-f(y))) = g(x)$? And for how many y's do we have

$$M(x,(y,1-f(y))) = g(x) \quad \& \quad M(x,(y,f(y))) = 1 - g(x) \ ?$$

Consider the following (randomized) Boolean circuit C. The input to the circuit is $y \in \{0,1\}^{m-1}$. The computation is:

1. Run M twice to get $b_0 = M(x,(y,0))$ and $b_1 = M(x,(y,1))$.
2. If $b_0 = b_1$, output a uniformly random bit.
3. If $b_0 \neq b_1$, compare both bits with $g(x)$ and output the bit z so that $b_z \neq g(x)$.

What is the size of C? What can you say about $\Pr[C(y) = f(y)]$? How does the circuit C know what $g(x)$ is? How can we make C deterministic (i.e., non-randomized)?

6 (50 p) This problem shows how randomness can help to communicate efficiently. There are three parties: a verifier V, a player A, and another player B. Player A knows a string $x \in \{0,1\}^n$ and player B knows a string $y \in \{0,1\}^n$. Your task is to construct a protocol with properties as described below.

The verifier sends a message $m \in \{0,1\}^t$ for $t = O(\log n)$ bits to both A and B. Player A computes a message $m_A \in \{0,1\}^t$ that depends on x, m and player B computes a message $m_B \in \{0,1\}^t$ that depends on y, m. The players sends their messages back to V. The verifier V computes a bit b that depends on m, m_A and m_B. The property that should hold is that if $x = y$ then $\Pr[b = 1] = 1$, and if $x \neq y$ then $\Pr[b = 1] \leq \frac{1}{3}$.

Hint: Think of x and y as the coefficients of two univariate polynomials $P(z)$ and $Q(z)$, respectively, and work modulo a random large prime. Use the fact that the number of primes $p \in \{1, \ldots, N\}$ is $\Omega(N/\log N)$.

7 (40 p) A *decision tree* T is a binary tree with edges directed from the root to the leaves and with leaves labelled $0/1$, non-leaves labelled by variables x_i, and the two edges out of every non-leaf labelled 0 and 1, respectively. We say that T *represents* the Boolean function $f : \{0,1\}^n \to \{0,1\}$ if for every assignment α it holds that when starting in the root of T and following the edge labelled by $\alpha(x_i)$ out of every non-leaf labelled by x_i we end up in a leaf labelled by the value $f(\alpha)$. The *depth* of a decision tree T is the length of a longest root-to-leaf path in T.

In this problem we want to study some connections between decision trees, CNF formulas, and DNF formulas.

7a Suppose that a Boolean function f can be represented as a decision tree of depth d. Show that f can also be represented as a d-CNF formula and as a d-DNF formula.

7b Suppose that a Boolean function f can be written both as a k-CNF formula and as an ℓ-DNF formula. Show that this implies that f also can be represented as a decision tree of depth at most $k\ell$.

8 (60 p) The goal of this exercise is to give a complete proof that $\mathsf{PSPACE} \subseteq \mathsf{IP}$, strengthening the result $\mathsf{coNP} \subseteq \mathsf{IP}$ that was proven in class.

Given a quantified Boolean formula (QBF) $\psi = \forall x_1 \exists x_2 \forall x_3 \cdots \exists x_n \, \phi(x_1, \ldots, x_n)$, we can use arithmetization as in our proof of $\mathsf{coNP} \subseteq \mathsf{IP}$ to construct a polynomial P_ϕ such that ψ is true if and only if $\prod_{b_1 \in \{0,1\}} \sum_{b_2 \in \{0,1\}} \prod_{b_3 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} P_\phi(b_1, \ldots, b_n) \neq 0$. However, the SumCheck protocol we used to decide the $\#\text{Sat}_D$ problem for CNF formulas no longer works, since each multiplication corresponding to a $\forall$-quantifier can double the degree of the polynomial.

8a (20 p) Suppose that ψ is a QBF formula (not necessarily in *prenex normal form* as described in Definition 4.10 and discussed further below in Arora-Barak) satisfying the following property: if $x_1, \ldots, x_n$ are the variables of ψ sorted in order of first appearance, then for every variable x_i there is at most a single universal quantifier involving x_j for any $j > i$ appearing before the last occurrence of x_i in ψ. Show that in this case, when we run the SumCheck protocol with the modification that we check $s(0) \cdot s(1) = K$ for product operations (i.e., $\forall$-quantifiers), the prover only needs to send polynomials of degree $\mathrm{O}(n)$ since the degree blow-up is at most a constant factor 2.

8b (20 p) Assuming that any QBF formula ψ can be rewritten to satisfy the property in Problem 8a, use this to show that $\text{T}\textsc{qbf} \in \mathsf{IP}$ and hence $\mathsf{PSPACE} \subseteq \mathsf{IP}$.

8c (20 p) Show that any QBF formula ψ of size m can be transformed into a logically equivalent formula ψ' of size $\mathrm{O}(m^2)$ that satisfies the property in Problem 8a.

Hint: Introduce a new variable y_i for any occurrence of x_i that we need to get rid of and encode that x_i and y_i take the same truth value.