Computability and Complexity: Problem Set 3

Due: Tuesday March 28 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 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 score of 100 points is guaranteed to be enough to pass this problem set.
Questions: Please do not hesitate to ask the instructor if any problem statement is unclear, but please make sure to send private messages—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 do not believe $\mathsf{NP} \subseteq \mathsf{P/poly}$), 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) Show that $\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP}$.

3 (30 p) Recall that the language

$$\textsc{CircuitEval} = \bigl\{\langle C, \alpha\rangle \bigm| C \text{ is a circuit that evaluates to 1 on } \alpha\bigr\}$$

is P-complete (we did not prove this in class, but this fact can be used freely in this problem). We want to understand the complexity of the closely related language

$$\textsc{CnfEval} = \bigl\{\langle F, \alpha\rangle \bigm| F \text{ is a CNF formula that evaluates to 1 on } \alpha\bigr\}$$

(in both of the descriptions above we are assuming that the domain of α matches the number of variables in C or F, or else we have a no-instance due to syntax error).

Your task is either to prove that $\textsc{CnfEval}$ is also P-complete, just as $\textsc{CircuitEval}$ is, or else to explain why this seems unlikely and what problems you run into when trying to prove P-completeness.

4 (30 p) We proved in class that the language $\textsc{Path} = \bigl\{\langle G, s, t\rangle \bigm| \exists \text{ path from } s \text{ to } t \text{ in } G\bigr\}$ is NL-complete. We also proved that $\mathsf{NL} = \mathsf{coNL}$, and, in particular, that for the complement language $\overline{\textsc{Path}} = \bigl\{\bigl\langle G', s', t'\bigr\rangle \bigm| \neg\exists \text{ path from } s' \text{ to } t' \text{ in } G'\bigr\}$ it holds that $\overline{\textsc{Path}} \in \mathsf{NL}$.

But this means that there must exist an implicitly logspace computable function that takes a directed graph G' and two vertices $s', t' \in V\bigl(G'\bigr)$ and outputs a directed graph G and two vertices $s, t \in V(G)$ such that there is *some path* from s to t in G if and only if there is *no path* from s' to t' in G'. Describe such a function and how to compute it.

You do not need to decribe every nut and bolt in the construction of G from G', but your description should contain enough details so that you could code it up in principle in your favourite high-level programming language (using well-defined subroutines that we also know can be coded up in principle).

5 (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.

5a 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.

5b 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 $k\ell$.

6 (50 p) Let multiprover interactive protocols be defined as the interactive protocols in Section 8.1 in Arora-Barak, except that there are several provers and that the verifier's messages in each round depends on previous messages from all provers (and on the verifier's private randomness). The messages sent by each prover only depends on the communication with the verifier, however, just as before. Let $\mathsf{MIP}[N]$ denote the set of languages that can be decided by N-multiprover interactive protocols in a polynomial number of rounds (in analogy with $\mathsf{IP} = \mathsf{MIP}[1]$ in Definition 8.6 in Arora-Barak).

Prove that, as claimed in class, only two provers are needed to realize the full power of multiprover interactive protocols. That is, prove that $\mathsf{MIP}[2] = \mathsf{MIP}[\text{poly}]$, where $\mathsf{MIP}[\text{poly}]$-protocols have a number of provers scaling polynomially with the size of the input.

7 (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.

7a (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.

7b (20 p) Assuming that any QBF formula ψ can be rewritten to satisfy the property in Problem 7a, use this to show that $\text{TQBF} \in \mathsf{IP}$ and hence $\mathsf{PSPACE} \subseteq \mathsf{IP}$.

7c (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 7a.

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.