KTH Computer Science and Communication

Computational Complexity: Problem Set 2

Due: Tuesday November 3, 2015, at 23:59 AoE. Submit your solutions as a PDF file by e-mail to `jakobn at kth dot se` with the subject line `Problem set 2:` ⟨`your full name`⟩. Name the PDF file `PS2_`⟨`YourFullName`⟩`.pdf` (with your name coded in ASCII without national characters), and also 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. Please try to be precise and to the point in your solutions and refrain from vague statements. *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 stated on the course webpage always apply.

Collaboration: Discussions of ideas in groups of two people are allowed—and indeed, encouraged—but you should write down your own solution individually and understand all aspects of it fully. You should also acknowledge any collaboration. State at the beginning of the problem set 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 easy 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 which can be found in chapters of Arora-Barak covered in the course, should be fair game, though, unless you are specifically asked to show something that we claimed without proof in class. It is hard to pin down 100% formal rules on what all this means—when in doubt, ask the lecturer.

About the problems: Some of the problems are meant to be quite challenging and you are not necessarily expected to solve all of them. A total score of around 75 points should be enough for grade E, 105 points for grade D, 135 points for grade C, 165 points for grade B, and 195 points for grade A on this problem set. Any corrections or clarifications will be given at `piazza.com/kth.se/fall2015/dd2445/` and any revised versions will be posted on the course webpage `www.csc.kth.se/DD2445/kplx15/`.

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) 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.
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.

3 (20 p) Show that $\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP}$.

4 (20 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 SpaceBoundedTM is PSPACE-complete from first principles (i.e., prove that SpaceBoundedTM is in PSPACE and that any other language in PSPACE reduces to it).

5 (20 p) We proved in class that there are oracles relative to which P and NP are equal by defining the language $\textsc{ExpCom} = \left\{\langle M, x, 1^n\rangle \middle| M \text{ accepts } x \text{ within } 2^n \text{ steps}\right\}$ and showing that $\mathsf{P}^{\textsc{ExpCom}} = \mathsf{NP}^{\textsc{ExpCom}} = \mathsf{EXP}$. In this problem we want to understand how important (or unimportant) the exact details in the definition of ExpCom is for this result to hold.

5a Let $\textsc{ExpCom}' = \left\{\langle M, x, 1^n\rangle \middle| M \text{ accepts } x \text{ within } n \text{ steps}\right\}$. Does it hold that $\mathsf{P}^{\textsc{ExpCom}'} = \mathsf{NP}^{\textsc{ExpCom}'} = \mathsf{EXP}$ hold? Modify the argument we gave in class to establish these equalities or explain where the proof fails.

5b Let $\textsc{ExpCom}'' = \left\{\langle M, x, n\rangle \middle| M \text{ accepts } x \text{ within } 2^n \text{ steps}\right\}$ (where n in the input is a number given in binary). Does it hold that $\mathsf{P}^{\textsc{ExpCom}''} = \mathsf{NP}^{\textsc{ExpCom}''} = \mathsf{EXP}$? Adapt the proof given in class or explain where it fails.

6 (20 p) Let us say that a function $f : \{0,1\}^* \to \{0,1\}^*$ is *write-once logspace computable* if f can be computed by a Turing machine M that uses $\mathrm{O}(\log n)$ space on its work tapes and whose output tape is *write-once*. By a write-once tape we mean a tape where at every time step M can either keep its head at the same position on the tape or write a symbol to it and move one location to the right, but M can never read from the tape or move left. The used cells on the write-once tape are not counted towards the space bound on M.

Prove that f is write-once logspace computable if and only if it is *implicitly logspace computable* as defined in class.

7 (30 p) When proving $\textsc{Parity} \notin \mathsf{AC}^0$, the starting point is a bounded-depth polynomial-size circuit C that supposedly computed the parity of its input bits. Before proving the actual lower bound, this circuit C can then be preprocessed to get an equivalent circuit C' such that:

1. All gates in C' have fan-out 1 (i.e., it is what is known as a *formula*, with a DAG structure that is a tree).
2. All NOT ($\neg$) gates are at the input level of C' (i.e., they only apply to variables).
3. The AND ($\wedge$) and OR ($\vee$) gates alternate, so that at each level of C' all gates are either AND or OR.
4. The bottom level has AND gates of some small, bounded fan-in (for the purposes of this problem, let us say some global constant K).

Show how this preprocessing can be done without increasing the circuit depth by more than a constant and the size more than polynomially (so that C' is also a bounded-depth polynomial-size circuit computing the parity of its input bits). If C is a circuit of size S and depth d, what size and depth do you get for C'?

8 (40 p) In this problem we want to study some connections between decision trees, CNF formulas, and DNF formulas.

8a Suppose that a Boolean function f can be represented as a decision tree in depth d. Show that f can also be represented as a d-CNF formula and as a d-DNF formula.

8b 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 has a decision tree of depth at most $k\ell$.

9 (50 p) Show that $\mathsf{NP} \neq \mathsf{SPACE}(n^2)$.
Hint: Use padding.

10 (60 p) Your task in this problem is to produce a complete, self-contained proof of (the vanilla version of) Ladner's theorem that we sketched in class. The goal is (at least) twofold:

- To have you work out the proof in detail and make sure you understand it.
- To train your skills in mathematical writing.

When you write the proof, you can freely consult the lecture notes as well as the relevant material in Arora-Barak, but you need to fill in all missing details. Also, the resulting write-up should stand on its own without referring to the lecture notes, Arora-Barak, or any other source.

Your write-up should be accessible to a student who has studied and fully understood the material at the level of *DD1352 Algorithms, Data Structures, and Complexity* but has not seen any more computational complexity than that (i.e., not more than the first three lectures of the current course, but you do not need to explain again the material in these lectures).

You are free to structure your proof as you like, except that all of the ingredients listed below should be explicitly addressed somewhere in your proof. (You can take care of them in whatever order you find appropriate, however. Please do not refer to the labelled subproblems in your write-up, since it should be a stand-alone text, but make sure your peer reviewer can find without problems where in your solution the different items are dealt with.)

10a Define

$$\textsc{Sat}_P = \left\{ \psi 01^{n^{P(n)}} \;\middle|\; \psi \in \textsc{CnfSat} \text{ and } n = |\psi| \right\}$$

as the language of satisfiable CNF formulas padded by a suitable number of ones at the end as determined by the polynomial-time computable function P.

10b Prove that if $P(n) = \mathrm{O}(1)$, then $\textsc{Sat}_P$ is NP-complete.

10c Prove that if $P(n) = \Omega(n/\log n)$, then $\textsc{Sat}_P \in \mathsf{P}$.

10d Argue that if we want $\textsc{Sat}_P$ not to lie in P but also not to be NP-complete, then P has to be unbounded but grow asymptotically more slowly than $n/\log n$.

10e Give a complete description of the algorithm computing $H(n)$ (as in the lecture notes) and prove that H is well-defined in that the algorithm terminates and computes some specific function.

10f Prove that not only does the algorithm terminate, but it can be made to run in time polynomial in n. (Note that there are a number of issues needing clarification here, such as, for instance, how to solve instances of CnfSat efficiently enough.)

10g Prove that $\text{Sat}_H \in \mathsf{P}$ if and only if $H(n) = \mathrm{O}(1)$.

10h Prove that if $\text{Sat}_H \notin \mathsf{P}$, then $H(n) \to \infty$ as $n \to \infty$.

10i Assuming that $\mathsf{P} \neq \mathsf{NP}$, prove that Sat_H does not lie in P but also cannot be NP-complete.