Carnegie Mellon University Instructor: Venkatesan Guruswami
15-252: More GITCS Term: Spring 2020

Problem Set 13

Due date: Sunday, May 3

Instructions

- You are allowed to collaborate with one other student taking the class, or do it solo.
- Collaboration is defined as discussion of the lecture material and solution approaches to the problems. Please note that *you are not allowed to share any written material and you must write up solutions on your own.* You must clearly acknowledge your collaborator in the write-up of your solutions.
- Solutions must be typeset in LaTeX and emailed to the TA.
- You should not search for solutions on the web. More generally, you should try and solve the problems without consulting any reference material other than what we cover in class and any provided notes.
- Please start working on the problem set early. Though it is short, the problem(s) might take some time to solve.

1. In the lecture, we have seen how to construct pseudorandom bits from average-case hard functions. In fact, the converse also holds: the existence of pseudorandom generators for a complexity class C implies the existence of average case hard functions for C.

   For the sake of simplicity, consider Turing Machines as the adversaries for this problem.

   Suppose that a probability distribution D on $\{0,1\}^n$ is a (C,ϵ)-pseudorandom distribution i.e. no Turing Machine running in time $C(n)$ can distinguish between D and the uniform distribution on n bits to accuracy better than ϵ. Prove that there exists no Turing Machine A that runs in time $C(n)$ that predicts nth bit of the distribution D from the first $n-1$ bits with accuracy better than $\frac{1}{2}+\epsilon$: that is, there is no Turing machine $A : \{0,1\}^{n-1} \to \{0,1\}$ running in time $C(n)$ such that

$$\Pr_{x\sim D}[A(x_1, x_2, \ldots, x_{n-1}) = x_n] > \frac{1}{2} + \epsilon$$

2. In this exercise, we will prove that least significant bit $\mathrm{lsb}(x)$ is not a hardcore predicate for the exponentiation function $g^x \mod p$. Let $p > 3$ be a prime number. Let $Z_p^* = \{1, 2, \ldots, p-1\}$. There is an integer $g \in Z_p^*$ such that $Z_p^* = \{g, g^2, \ldots, g^{p-1}\}$ (here, all the multiplications are modulo p.) Prove that there is an algorithm that runs in time $\mathrm{poly}(\log p)$ that given g^x for some $x \in Z_p^*$ outputs correctly whether x is even or odd.