Problem Set 4

Pseudorandomness, Autumn 2023, University of Chicago
Instructor: William Hoza (williamhoza@uchicago.edu)

Submission. Solutions are due **Friday, November 17** at 5pm Central time. Submit your solutions through Canvas. You are encouraged to typeset your solutions in LaTeX. (Consider using Overleaf, a popular online LaTeX editor.) If you prefer, you may submit a photo of handwritten solutions instead.

The collaboration and resource policies below can also be found on the course webpage.

Collaboration. You are encouraged to collaborate with your classmates on problem sets, but you must adhere to the following rules.

- Before discussing a problem with a classmate, you must work on the problem on your own for at least 20 minutes.
- You must ultimately write your solution on your own. While writing your solution, you may not consult any notes that you took during a discussion with another classmate.
- In your write-up, you must list any classmates who contributed to your solution through discussion. The fact that A contributed to B's solution does not necessarily mean that B contributed to A's solution.

Permitted Resources for Full Credit. Beyond discussions with me and discussions with classmates as discussed above, you may use the course texts and any notes that might be posted in the "Course Timeline" section of the course webpage. If you wish to receive full credit on a problem, you may not use any other resources.

Permitted Resources for Partial Credit. If you wish, you may use outside resources (Wikipedia, ChatGPT, Stack Exchange, research papers, etc.) to solve a problem for partial credit. If you decide to go this route, you must make a note of which outside resources you used when you were working on each problem. You must disclose using a resource even if it was ultimately unhelpful for solving the problem. Furthermore, if you consult an outside resource while working on a problem, then you must not discuss that problem with your classmates.

Recall that conditional min-entropy is defined by the formula

$$\widetilde{\mathrm{H}}_\infty[X \mid A] = \log\left(\frac{1}{\mathbb{E}_{a\sim A}[\max_x \Pr[X = x \mid A = a]]}\right).$$

Definition 1 (Average-case extractors)**.** An *average-case* (k, ε)*-extractor* is a function

$$\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

such that for every pair of discrete random variables X, A, if X is distributed over $\{0,1\}^n$ and $\widetilde{\mathrm{H}}_\infty[X \mid A] \geq k$, then

$$\Delta\big((\mathsf{Ext}(X, U_d), A), (U_m, A)\big) \leq \varepsilon,$$

where U_d is a uniform random seed that is independent of (X, A) and U_m is a uniform random m-bit string that is independent of A.

In class, we stated that every extractor is an average-case extractor (with some minor loss in parameters), but we did not prove it. In this problem, you will prove this fact, albeit with slightly different parameters than what we stated in class.

Problem 1 (10 points)**.**

(a) Let X and A be random variables and let $k, \varepsilon > 0$. Show that if $\widetilde{\mathrm{H}}_\infty[X \mid A] \geq k + \log(1/\varepsilon)$, then

$$\Pr_{a\sim A}\Big[\mathrm{H}_\infty\big[(X|A = a)\big] \geq k\Big] \geq 1 - \varepsilon.$$

Here $(X|A = a)$ denotes the conditional distribution of X given $A = a$.

(b) Show that every (k, ε)-extractor is an average-case $(k + \log(1/\varepsilon), 2\varepsilon)$-extractor. *Hint:* For any particular statistical test, bound the error by an expression of the form $\sum_a \Pr[A = a] \cdot e(a)$, then apply part (a).

Ideally, we would like to prove $\mathsf{P} = \mathsf{BPP}$. So far, it is an open problem to merely show that $\mathsf{BPP} \subseteq \mathsf{NP}$! In this problem, you will show that BPP is contained in a certain complexity class Σ_2^{P} that is closely related to NP.

Recall that by definition, a language L is in BPP if there is some randomized polynomial-time algorithm A such that for every input $\sigma \in \{0,1\}^*$,

$$\begin{aligned} \sigma \in L &\implies \Pr[A(\sigma) = 1] \geq 2/3 \\ \sigma \notin L &\implies \Pr[A(\sigma) = 1] \leq 1/3. \end{aligned}$$

Meanwhile, according to one definition, a language L is in NP if there is some deterministic polynomial-time algorithm A and some constant k such that for every $n \in \mathbb{N}$ and every input $\sigma \in \{0,1\}^n$,

$$\sigma \in L \iff \exists x \in \{0,1\}^{n^k},\ A(\sigma, x) = 1.$$

(Recall that the string x is called the *witness* and A is called the *verifier*. For example, if L is the 3-SAT problem, then σ is a 3-CNF, x is an assignment, and $A(\sigma, x)$ checks whether x satisfies σ.) We define Σ_2^{P} by introducing one more quantifier:

Definition 2 (The complexity class Σ_2^{P})**.** Let L be a language. By definition, $L \in \Sigma_2^{\mathsf{P}}$ if there is some deterministic polynomial-time algorithm A and some constant k such that for every $n \in \mathbb{N}$ and every input $\sigma \in \{0,1\}^n$,

$$\sigma \in L \iff \exists x \in \{0,1\}^{n^k},\ \forall y \in \{0,1\}^{n^k},\ A(\sigma, x, y) = 1.$$

Vadhan's text includes a proof that $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$ (see Theorem 3.14). In this problem, we will study an alternative proof that $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}}$ based on seeded randomness extractors.

Problem 2 (10 points)**.** Let $L \in \mathsf{BPP}$ be a language.

(a) Show that for every constant $\alpha > 0$, the language L can be decided by a randomized polynomial-time algorithm A with failure probability at most 2^{-r+r^α}, where r is the number of random bits that the algorithm uses.

(b) Let A be the algorithm from the previous part with (say) $\alpha = 0.5$. (Actually all we need is that that failure probability is strictly smaller than $2^{-\lceil r/2 \rceil}$.) Let σ be an input, and let r be the number of random bits that A uses on input σ. Let $A(\sigma, xy)$ denote the output of A on input σ and randomness xy where $x \in \{0,1\}^{\lfloor r/2 \rfloor}$ and $y \in \{0,1\}^{\lceil r/2 \rceil}$. Show that

$$\begin{aligned} \sigma \in L &\implies \exists x, \forall y, A(\sigma, xy) = 1 \\ \sigma \notin L &\implies \forall x, \exists y, A(\sigma, xy) = 0. \end{aligned}$$

For part (a), you may take as given the following theorem, which we state but do not prove in class.

Theorem 1 (GUV Extractor)**.** *For every $r, k \in \mathbb{N}$ with $k \leq r$, for every $\varepsilon > 0$, there exists an explicit (k, ε)-extractor $\mathsf{Ext}\colon \{0,1\}^r \times \{0,1\}^d \to \{0,1\}^m$ with seed length $d = O(\log(r/\varepsilon))$ and output length $m \geq k/2$.*

Note: In this context, "explicit" means that $\mathsf{Ext}(x, y)$ can be computed in $\mathrm{poly}(r, d)$ time.

In class, we discuss the connection between PRGs and hard Boolean functions. In this problem, you will construct a function h that is "perfectly hard" for bounded-depth decision trees, meaning that a bounded-depth decision tree does no better than a coin toss at computing h. This "explains" why it is possible to construct a PRG that fools bounded-depth decision trees with error zero (namely, a k-wise uniform generator).

Problem 3 (10 points)**.** Show that for every positive integer n, there exists an explicit[1] function $h\colon \{0,1\}^n \to \{0,1\}$ such that for every decision tree f of depth $n-1$, we have

$$\Pr_{x \sim U_n}[f(x) = h(x)] = \frac{1}{2}. \tag{1}$$

Hint: First figure out a good guess for what h should be. (It's a simple and familiar function.) Then prove (1). There are multiple approaches to proving (1). One approach, which is not necessarily the simplest but which is satisfying in other ways, is to look at the quantity $\mathbb{E}_x[(-1)^{f(x)} \cdot (-1)^{h(x)}]$, which is called the "correlation" between f and h. Expand $(-1)^{f(x)}$ as a sum over all leaves, and use the fact that $\mathbb{E}[A \cdot B] = \mathbb{E}[A] \cdot \mathbb{E}[B]$ whenever A and B are independent.

[1] I.e., $h(x)$ can be computed in $\mathrm{poly}(n)$ time given x.