CMPUT 675 - Winter 2019
Assignment #4 - Due March 21 (Thursday) by 12:30pm

All **Exercises** have the same weight, regardless of their difficulty. As you have a bit less time to solve this one after the due date of the previous assignment (and since I released this one day later than usual), you are allowed to skip **one** exercise freely with no penalty. If you answer more than what is required, I will drop the ones with the lowest marks when computing your mark for this assignment.

It is highly recommended that you typeset your solutions in LaTeX. I still want hard copies of your solution, so submit a printout if you do typeset it. As always, if any question is not clear then please feel free to ask me for clarification.

Pages: 4

Exercise 1: Sampling Arbitrary Values Using Unbiased Coins

One detail we have glossed over a few times is the ability to sample random values from certain distributions when we only have access to random coin flips. For example, in the proof of **IP** = **PSPACE** we uniformly sampled a random value $a \sim \{0, 1, \ldots, p-1\}$ where p was some prime. But how can this be done with random coin flips?

- Argue, briefly, that if we sample $a \sim \{0, 1, 2, \ldots, 2^k - 1\}$ where k is the largest integer satisfying $2^k \leq p$ then the interactive proof protocol for #SAT is still sound enough. That is, with this modification we can still get the verifier to reject every prover in the "no" case with probability at least 2/3 for large enough inputs.

  Obviously this extends to the protocol for TQBF, but it is enough to check it just for #SAT for this exercise.

- In some contexts, such a trick is not possible. Show the following "workaround". For any $n \geq 1$ and any $\epsilon > 0$, describe a way to flip poly($\log n, \log \frac{1}{\epsilon}$) random bits to sample from $\{0, 1, 2, \ldots, n-1\} \cup \{\text{FAIL}\}$ so that:

  - All $x \in \{0, 1, 2, \ldots, n-1\}$ have the exact same probability of being sample.
  - $\mathbf{Pr}[\text{FAIL is sampled}] \leq \epsilon$.

  Hint: Start by trying to get $\mathbf{Pr}[\text{FAIL is sampled}] \leq 1/2$.

- Now suppose you have access to a sequence of bits $b_1, b_2, b_3, \ldots$ (each $b_i \in \{0, 1\}$) describing the binary expansion of a real value $x \in [0, 1]$. That is, $x = \sum_{i \geq 1} \frac{b_i}{2^i}$. By "have access", I mean you can compute b_i on the fly for any i. Don't worry about the complexity of computing such b_i, but it is possible to do this with many well-known irrational constants like $1/\pi, 1/e$ and $1/\sqrt{2}$.

- Show for any $k \geq 1$ you can sample some $Y \in \{0,1\}$ by flipping only k random coins so $x \leq \mathbf{Pr}[Y = 1] \leq x + 2^{-k}$.
- Describe how to sample some $Y \in \{0,1\}$ by flipping at most 2 coins *in expectation* so $\mathbf{Pr}[Y = 1] = x$.

Exercise 2: A Gap for E3SAT

For a SAT instance ϕ, let unsat(ϕ) be the minimum-fraction of clauses of ϕ that can be left unsatisfied by some variable assignment. In particular, ϕ is satisfiable if and only if unsat(ϕ) $= 0$.

We saw how the PCP theorem implies the following for some constant $q \geq 3$ and some constant $\rho > 0$. For any $L \in \mathbf{NP}$, there is a polynomial-time reduction f from L to qSAT such that for any $x \in \{0,1\}^*$, if $x \in L$ then unsat(ϕ) $= 0$ and if $x \notin L$ then unsat(ϕ) $\geq \rho$.

Your job is to describe how to convert an instance ϕ of qSAT to an instance ψ of E3SAT in polynomial time such that:

- If ϕ is satisfiable, then so is ψ.
- If unsat(ϕ) $\geq \rho$ then unsat(ψ) $\geq \rho'$ where $\rho' > 0$ is a constant that depends only on ρ and q.

Thus, there is some constant $\rho' > 0$ such that we cannot approximate E3SAT with a factor greater than $1 - \rho'$.

Hint: A standard reduction from qSAT to E3SAT works, your main task is to show some "constant gap" is preserved.

Exercise 3: A Different PCP

We discussed the big *PCP Theorem*: that $\mathbf{PCP}(O(\log n), O(1)) = \mathbf{NP}$.

Swap the parameters: prove $\mathbf{PCP}(O(1), O(\log n)) = \mathbf{P}$.

Exercise 4: A Characterization of Linear Functions

A function $f : \{0,1\}^n \to \{0,1\}$ was said to be *linear* if $f(x+y) = f(x) + f(y)$ for any $x, y \in \{0,1\}^n$ where addition is modulo 2. Prove that a function f is linear if and only if there is some $u \in \{0,1\}^n$ such that $f(x) = x \circ u$ for every $x \in \{0,1\}^n$. Here, we let $x \circ u = \sum_{i=1}^{n} x_i \cdot u_i \pmod 2$.

Obviously this would follow from the far more general "linearity test" result that is eventually proven in Chapter 22, but your proof should be "elementary" and not use the Fourier analysis tool from Chapter 22.

Exercise 5: Dominating Sets

Let $G = (V; E)$ be a graph. A set S is a **dominating set** if every $u \in V - S$ has a neighbour in S. That is, if $u \notin S$ then $v \in S$ for some edge $uv \in E$.

For a quantity $\alpha \geq 1$, an α-approximation for the minimum dominating set problem is a polynomial-time algorithm A such that given a graph G, $A(G)$ outputs a dominating set S of G with size at most α times the size of a minimum dominating set of G.

Show there is some constant $\alpha > 1$ such that there is no α-approximation for the minimum dominating set problem unless $\mathbf{P} = \mathbf{NP}$.

Exercise 6: Computationally Secure Encryption

Essentially question 9.3 from the book. We mentioned this in class, but I said it was up to you to *carefully* verify it is true.

Suppose we have a pseudorandom generator G of stretch $\ell(n)$. Show the following encryption scheme (E, D) with keys of length $K(n)$ satisfying $\ell(K(n)) = n$ is computationally secure.

Given a message x, E will sample $k \in \{0,1\}^{K(|x|)}$ uniformly at random and encrypt x using the key $G(k)$ as a one-time pad. That is, $E_k(x)$ is the ciphertext y with $|y| = |x|$ where $y_i = x_i \oplus G(k)_i$ (addition mod 2) for all $1 \leq i \leq |y|$.

Note: Naturally, D decrypts the exact same way as E encrypts because adding $G(k)$ to y yields x itself.

Exercise 7: Iterating Permutations

Question 9.11 from the book.

Help complete the proof of how we get pseudorandom generators from one-way permutations. Let $f : \{0,1\}^* \to \{0,1\}^*$ be a one-way permutation: a one-to-one, one-way function with $|f(x)| = |x|$ for each x.

For an integer $k \geq 0$ define f^k inductively by $f^0(x) = x$ and $f^k = f(f^{k-1}(x))$ for $k \geq 1$. Then for any polynomial $p(n)$ define the function $f^p : \{0,1\}^* \to \{0,1\}^*$ as $f^{p(|x|)}(x)$.

Show f^p is a one-way permutation for any polynomial $p(n)$.

Exercise 8: Reconstructing Vectors

Question 9.13 from the book.

Let $x \in \{0,1\}^m$ be some unknown vector. Suppose we sample random $r^1, \ldots, r^m \sim \{0,1\}^m$ independently and all $x \circ r^i$ values (our usual dot product mod 2) are revealed to us for all i. Describe an efficient, deterministic algorithm that tries to reconstruct x from these dot products and show it succeeds with probability at least some absolute constant $c > 0$ that is independent of m. The probability being over the random choices of the r^i.

Hint: The book's hint mentions we need to show a certain determinant is not zero. My hint is to show the following: if you view the r^i vectors as lying in the vector space $(\mathbb{Z}/2\mathbb{Z})^m$, then each vector r^i is independent from the previously-sampled vectors $r^1, \ldots, r^{i-1}$ with very good probability.