6.541/18.405 Problem Set 3

due on **Thursday, May 2, 11:59pm**

Rules: You may discuss homework problems with other students and you may work in groups, but we require that you *try to solve the problems by yourself before discussing them with others.* Think about all the problems on your own and do your best to solve them, before starting a collaboration. If you work in a group, include the names of the other people in the group in your written solution. **Write up your *own* solution to every problem**; don't copy answers from another student or any other source. Cite **all** references that you use in a solution (books, papers, people, websites, etc) at the end of each solution.

We encourage you to use LaTeX, to compose your solutions. The source of this file is also available on Piazza, to get you started!

How to submit: Use Gradescope entry code **2P3PEN**.

Please use a separate page for each problem.

Problem 1: Circuit Lower Bounds from Pseudorandom Generators (3 Points)

Question

We showed that circuit lower bound implied "good" PRGs. Here is a sort-of converse:

Show that if there is a $O(\log(n))$-seed 1/10-pseudorandom generator computable in $2^{O(m)}$ time on m-bit seeds, then there is an $\varepsilon > 0$ such that $\mathbf{DTIME}[2^{O(n)}] \not\subset \mathbf{SIZE}[2^{\varepsilon n}]$.

Answer

TODO: clean this up! Currently messy enough there is a small chance my approach doesn't work.

Say there exists such a pseudorandom generator $g : \{0,1\}* \to \{0,1\}*$ computed by Turing machine G running in 2^{km+l} time. WLOG, say that for all n, $|x| = c\log(n) \implies |g(x)| = n$. For any circuit C of size n,

$$|Pr_{x\in\{0,1\}^{c\log(n)}}[C(g(x)) = 1] - Pr_{x\in\{0,1\}^n}[C(x) = 1]| < 1/10$$

The idea is going to be that in time $2^m \times 2^{km} = 2^{(k+1)m}$, a Turing machine can run g on every length m string, and compute $Pr_{x\in\{0,1\}^{c\log(n)}}[C(g(x)) = 1]$ exactly, but no circuit of size 2^m can do this.

Consider the decision problem $f : \{0,1\}^* \to \{0,1\}$ where

$$f(x) = 1 \iff \exists y \in \{0,1\}^{|x|-1} \text{ s.t. } g(y) = x \qquad (*)$$

Suppose for contradiction that $f \in \mathbf{SIZE}[2^n]$. Let C be the circuit of size 2^n that decides this for a given n. Say $m = cn$ (so $m = c\log(2^n)$). Observe

$$Pr_{y\in\{0,1\}^m}[C(g(y)_{1:m+1}) = 1] = 1$$

Here, $g(y)_{1:m+1}$ is the first $m+1$ bits of $g(y)$.

However,

$$Pr_{x\in\{0,1\}^{2^n}}[C(x_{1:m+1}) = 1] \leq 1/2$$

because only half of the $m+1$ bit strings can be in the range of a function with domain $\{0,1\}^m$. This contradicts $(*)$. Thus $f \notin \mathbf{SIZE}[2^n]$. But certainly $f \in \mathbf{DTIME}[2^{O(n)}]$, because it is possible to loop over all 2^n y values of length n, and for each one check if $x = g(y)$ in time $2^{O(n)}$. This yields an algorithm for deciding f with total runtime $2^n 2^{O(n)} = 2^{O(n)+n} = 2^{O(n)}$.

Problem 2: Randomized Approximate Counting with an NP Oracle (12 pts, 3 for each sub-problem)

Question

We will develop a real-life application of SAT solvers. **Assume P = NP in this question.** Let $H_{n,k}$ be a pairwise independent hash family of functions from $\{0,1\}^n$ to $\{0,1\}^k$.

(a) Prove that there is a constant $p \in (0,1)$ and a constant $\varepsilon > 0$ such that for every k and $S \subseteq \{0,1\}^n$,

- if $|S| \leq 2^{k-2}$, then

$$\Pr_{h \in H_{n,k}}[\text{there is an } x \in S \text{ such that } h(x) = 0^k] < p - \varepsilon,$$

and

- if $|S| \geq 2^{k-1}$, then

$$\Pr_{h \in H_{n,k}}[\text{there is an } x \in S \text{ such that } h(x) = 0^k] > p + \varepsilon.$$

(b) Use part (a) to show that that there is a randomized polynomial-time algorithm that approximates #SAT within a factor of 4. More precisely, there is a randomized polynomial-time algorithm that given any Boolean formula F outputs a number K such that $\#\mathsf{SAT}(F)/2 \leq K \leq 2 \cdot (\#\mathsf{SAT}(F))$.

(c) Show that for any constant $\varepsilon > 0$, there is a randomized polynomial-time algorithm that approximates #SAT within a factor of $1+\varepsilon$. (Hint: Try to modify the given formula F in some natural way that changes the number of SAT assignments, then feed the modification to your algorithm from part (b).)

(d) Show that you can derandomize the algorithm. That is, prove that if **P** = **NP** then for every function $f \in \#\mathbf{P}$, and any constant $\varepsilon > 0$, there is a deterministic polynomial-time algorithm that approximates f within a factor of $1 + \varepsilon$. (Warning: the approximate counting problem is *not* a decision problem, so you cannot just "plug in" **P** = **NP** here...)

Answer

Answer to (a)

Problem 3: Constant Round Arthur-Merlin Collapses (3 points)

Prove that for every fixed positive integer k, $\mathbf{AM}[k] \subseteq \mathbf{AM}[2]$.

Hint: Try error-reduction, to make the probability of error very small.

Problem 4: AM Protocol for Set Lower Bound (6 Points, 2 for each sub-problem)

In this problem, we will develop an **AM** protocol for proving a set lower bound, which is used as a subroutine in the **AM** protocol for graph non-isomorphism. In a set lower bound protocol, the prover needs to prove to the verifier that given a (large) set $S \subseteq \{0,1\}^m$ (where membership in S is efficiently verifiable), S has cardinality at least K, up to a factor of 2. More precisely, given any K,

- if $|S| \geq K$ then the prover can make the verifier accept with high probability;
- if $|S| < K/2$ then the verifier rejects with high probability regardless of what the prover does.

(a) Let $H_{m,k}$ be a pairwise independent hash family of functions from $\{0,1\}^m$ to $\{0,1\}^k$. Use the pairwise independent hash family $H_{m,k}$ to give a 2-round **AM** protocol for the set lower bound problem described above.

(b) Show that there exists an **AM** protocol for set lower bound with perfect completeness.

Hint: Consider the case where the prover uses multiple hash functions $h_1, \ldots, h_n$ so that $\bigcup_{i=1}^{n} h_i(S) = \{0,1\}^k$.

(c) Generalize the idea from part (b) to show that every problem in **MA** has a protocol with perfect completeness. Namely, show that for every language $L \in \mathbf{MA}$, there exists a probabilistic polynomial time verifier V such that

- If $x \in L$, then there exists m such that $\Pr_r[V(x,r,m) = 1] = 1$.
- If $x \notin L$, then for all m, $\Pr_r[V(x,r,m)] \leq 1/3$.

Problem 5: The Limits of PCPs (4 Points, 2 for each sub-problem)

Recall that in class we defined $\mathbf{PCP}_s[r(n), q(n)]$ to be the set of functions with probabilistically checkable proofs having "soundness" s. In general, we can parametrize the "completeness" as well.

Specifically, define $f : \{0,1\}^* \to \{0,1\}$ to be in $\mathbf{PCP}_{c,s}[r(n), q(n)]$ if there is a probabilistic polynomial time algorithm V such that for all x, V uses $O(r(|x|))$ random bits, asks $q(|x|)$ oracle queries to a proof string P non-adaptively, must decide whether accept or reject, and

- $f(x) = 1 \Longrightarrow$ there is a P such that $\Pr[V^P(x) \text{ accepts}] \geq c$.
- $f(x) = 0 \Longrightarrow$ for all P, $\Pr[V^P(x) \text{ accepts}] < s$.

Note that in this generalized version, when $f(x) = 1$, we do not require the verifier to accept with probability 1 on some proof P.

In the PCP lectures, it was proved that $\mathbf{PCP}_{1,1}(\log n, 3) = \mathbf{NP}$. The number 3 here is actually the smallest possible. In this problem, you are asked to show that if we reduce the number of queries to two or one, the classes become $\mathbf{P}$. Prove that:

(a) for every $0 < s \leq c \leq 1$, $\mathbf{PCP}_{c,s}(\log n, 1) = \mathbf{P}$.

(b) for every $0 < s \leq 1$, $\mathbf{PCP}_{1,s}(\log n, 2) = \mathbf{P}$.

Hint: Think about these 1-query and 2-query PCPs from the CSP/inapproximability perspective: what you want to show is that the resulting CSPs are in fact easy to solve.

Extra credit: Prove that for every $0 < s \leq 1$, $\bigcup_{k \geq 1} \mathbf{PCP}_{1,s}(n^k, 2) \subseteq \mathbf{PSPACE}$

Hint: Use the fact that 2SAT is in NL.