CSE 533: PCPs and Hardness of Approximation **Autumn 2005**

Problem Set 1
Due: Wednesday, November 16

Homework policy: Students are encouraged to work on the problems in groups; however, all writeups should be done individually. We suggest that you think about and try to solve all the problems, but it is enough to turn in write-ups for any **six** problems.

---

1. Hardness-of-approximation reductions. Håstad proved a version of the PCP Theorem (as stated in Lecture 1) in which: completeness is $1 - \epsilon$; soundness is $1/2 + \epsilon$; the number of queries C is 3; and, all predicates ψ the verifier uses are of the form "$x_{i_1} + x_{i_2} + x_{i_3} = b \mod 2$", where b is 0 or 1. Here ϵ can be any positive constant. (We will prove this result later in the course.)

a) Assuming P $\neq$ NP, show that the following statement is false: "There is a probabilistically checkable proof for an NP-complete problem in which a) three bits of the proof get queried, b) all predicates are of the form "$x_{i_1} + x_{i_2} + x_{i_3} = b \mod 2$", c) completeness is 1, and d) soundness is 51%." (NB: We only consider PCPs with *nonadaptive* verifiers, as described in Lecture 1.)

b) Let MAX-3LIN be the maximization problem where the input is a set of 3-variable linear equations mod 2 and the goal is to find an assignment satisfying as many equations as possible. Show that for any $\epsilon > 0$, there is no $(1/2 + \epsilon)$-approximation algorithm for MAX-3LIN unless P = NP.

c) Show that there is no $(7/8 + \epsilon)$-approximation algorithm for MAX-E3SAT unless P = NP, as mentioned in class. (Hint: reduce from the hardness of MAX-3LIN.)

d) Let MAX-3MAJ be the optimization problem where the input is a set of constraints over 3 boolean literals each, where each constraint asserts that the majority of its three literals' values is 1. (E.g., a constraint might be "Maj$(x_1, \overline{x}_3, \overline{x}_7) = 1$".) Show that there is no $(2/3 + \epsilon)$ approximation for MAX-3MAJ unless P = NP. (Hint: reduce from the hardness of MAX-3LIN.)

A remark: There is a 2/3-approximation algorithm for MAX-3MAJ due to Zwick; however it is quite nontrivial.

2. 2-Prover 1-Round Games. A "2-Prover 1-Round Game" (2P1R Game) is a kind of proof system for languages L that works as follows: There are two "all-powerful" provers P_1 and P_2 and a polynomial-time verifier V. There are also polynomial-sized "question sets" Q and R and constant-sized "answer sets" A and B. The two provers are allowed to coordinate strategies beforehand, but once the "game" starts *they cannot communicate with each other*. The game consists of an input $x \in \{0,1\}^n$ which the provers and the verifier all see; the provers have to try to convince the verifier that $x \in L$.

On input x, the verifier first does some deterministic computations and then decides on: a) a probability distribution π on $Q \times R$, and b) a (deterministic) predicate φ on $Q \times R \times A \times B$. Next, the verifier uses randomness to draw a pair of "questions" $(q, r) \in Q \times R$ according to π. The verifier sends q to P_1 and r to P_2. The provers, based on x and the question they receive, send back "answers"; P_1 returns some $a \in A$ and P_2 returns some $b \in B$. (Recall that the provers are *not* allowed to communicate.) Finally, V applies φ to (q, r, a, b) and accordingly either accepts or rejects.

a) Show that it doesn't help the provers if they are allowed to use randomness.

b) We will show later in class that for every constant $\epsilon > 0$ there exists a constant-sized alphabet Σ such that GAP-CG$_{1,\epsilon}(\Sigma)$ is NP-hard. Using this fact, show that for every $L \in$ NP and every $\epsilon > 0$ there is a 2P1R Game for L in which the verifier accepts every $x \in L$ with probability 1 and accepts every $x \notin L$ with probability at most ϵ. (Hint: what extra graph property is needed for 2P1R Games?)

3. Error reduction using expanders. Let $G = (V, E)$ be an (n, d, λ)-expander with $\lambda < d$ constants. Let $B \subset V$ be a set of vertices with $|B| = \alpha n$, where $0 < \alpha < 1$. (We think of B as a "bad" set of vertices.) Suppose we pick a uniformly random vertex in G and then perform a t-step random walk in G starting from this vertex. We wish to upper-bound the probability γ that *all* vertices encountered are in B.

a) Let A denote the normalized adjacency matrix of G, and let P denote the matrix corresponding to "projection onto B"; in other words, P is the $n \times n$ diagonal matrix with 1's in the positions corresponding to B. Show that $\gamma = \|PAPAP \cdots APx\|_1$, where x is the vector $(1/n, \ldots, 1/n)$, $\|z\|_1$ denotes $\sum_{i=1}^n |z_i|$, and the matrix product $PAPAP \cdots AP$ has precisely t A's.

b) The "matrix 2-norm" of a matrix C is defined to be $\|C\|_2 := \max_{y \neq 0} \|Cy\|_2 / \|y\|_2$. Show that $\gamma \leq \alpha \|PAPAP \cdots AP\|_2 \leq (\|AP\|_2)^t$.

c) Show that $\|AP\|_2 \leq \sqrt{\alpha^2 + (\lambda/d)^2}$, and conclude $\gamma \leq (\alpha^2 + (\lambda/d)^2)^{k/2}$. (Hint: given arbitrary $y \neq 0$, write $z = Py$ and express $z = z^{\|} + z^{\perp}$ as in Lecture 3…) Bonus: show that in fact $\|PAP\|_2 \leq \lambda/d + \alpha(1 - \lambda/d)$ and show how this can be used to conclude the sharper upper bound $\gamma \leq \alpha(\lambda/d + \alpha(1 - \lambda/d))^t$.

d) Suppose we have an RP algorithm for a problem; i.e., on NO instances the algorithm always says NO and on YES instances the algorithm says YES with probability at least 1/4. Further suppose that the algorithm uses r random bits. Naive serial repetition reduces the error probability to $(3/4)^t$ using rt random bits. Show that the same error probability can be achieved using only $O(r + t)$ random bits.

A remark: With a little more effort, a similar randomness-efficient error amplification can be done for BPP algorithms.

e) Show that there is a PCP for NP with completeness 1 and soundness $1/n$ in which the verifier uses $O(\log n)$ random bits (as opposed to $O(\log^2 n)$) and queries $O(\log n)$ bit positions in the proof.

4. Hardness of Clique, and graph products.

a) Improve the hardness result we showed in class for Clique by proving that for some $\alpha > 0$, there is no $n^{-\alpha}$-approximation algorithm for Clique unless P = NP. (Hint: Apply the FGLSS reduction to the PCP of Problem 3(e).)

We will now explore a different language, namely that of graph products, for boosting hardness-of-approximation results for Clique. For a graph $G = (V, E)$ and integer $k \geq 2$, we define the kth power of G, $G^k = (V', E')$, as follows: The vertex set V' equals V^k, the set of k-tuples of vertices of G. Two distinct vertices $(u_1, u_2, \ldots, u_k)$ and $(v_1, v_2, \ldots, v_k)$ are adjacent in E' if and only if $\{u_1, u_2, \ldots, u_k, v_1, \ldots, v_k\}$ is a clique in G (note that the u_i and v_j do not have to be distinct).

b) Prove that the powering operation defined above satisfies $\omega(G^k) = \omega(G)^k$.

c) Use (b) to prove that if Clique is NP-hard to approximate within some constant factor $\rho < 1$, then

(i) it is NP-hard to approximation within *any* constant factor $\epsilon > 0$, and

(ii) Clique does not admit a polynomial time $2^{-\log^\gamma n}$-approximation algorithm for any $\gamma < 1$ unless NP $\subseteq \bigcup_{c \geq 1} \text{DTIME}(2^{(\log n)^c})$.

d) Suppose for some $\epsilon > 0$ there is a polynomial time algorithm that on input a graph H on n vertices, returns a clique of size at least $\omega(H)/n^{1-\epsilon}$. Prove that for every $a > 0$, there is a randomized polynomial time algorithm that, when given a graph G on N vertices with $\omega(G) \geq a \cdot N$, outputs a clique of size $b_{a,\epsilon} \cdot N$ in G, where $b_{a,\epsilon} > 0$ is a constant depending on a, ϵ. (Hint: Take a larger power G^k of G for $k = \Theta(\log N)$. This graph is too big, so work with a subgraph obtained by sampling a suitable polynomial number $N^{O(1)}$ vertices from G^k.)

e) Using (d), argue that the same hypothesis about existence of approximation algorithms for Clique implies that a 3-colorable graph on N vertices can be colored using $O(\log N)$ colors in randomized polynomial time.

A remark: The proof technique of (d) can also be used to show that for some $\alpha > 0$, an $n^{-\alpha}$-approximation algorithm for Clique implies NP = RP. The conclusion can also be strengthened to NP = P using a derandomization of the sampling procedure.

5. Amplification fails beyond $1/2$. (This problem is due to Andrej Bogdanov.) As we saw in class, the Powering step in Dinur's construction yielded $\text{gap}' \geq 2\min(\text{gap}, 10^{-6})$ when the parameter t was a large enough fixed constant. But what if gap is already quite large — could repeating the Powering step push gap′ all the way towards 1? This problem gives a negative answer.

a) It is known that for infinitely many constants d there exist (n, d, λ)-expanders G for infinitely many n, with the following two properties: (i) $\lambda(G) \leq 2\sqrt{d}$; (ii) G has "girth" at least $\frac{2}{3}\log_d n$, where the girth of a graph is the length of the smallest cycle in it. Suppose we make G into a constraint graph over the alphabet $\{0, 1\}$ by putting an "inequality" constraint on every edge. Show that the satisfiability gap of G is at least $1/2 - O(1/\sqrt{d})$.

b) On the other hand, show that for any fixed parameter t, if n is large enough, then the Powered constraint graph G' produced from it via Dinur's method has $\text{gap}' \leq 1/2$.

6. Fourier interpretations. Let $f : \{-1,1\}^n \to \mathbb{R}$ and write the "Fourier expansion of f", $f = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S$. All probabilities and expectations in this question are with respect to the uniform product probability distribution on $\{-1,1\}^n$.

a) Given a set $S \subseteq [n]$, define $f^{\leq S} : \{-1,1\}^n \to \mathbb{R}$ by

$$f^{\leq S} = \sum_{T : T \subseteq S} \hat{f}(T)\chi_T.$$

Note that $f^{\leq S}(x)$ actually only depends on the bits of x in S; call these bits x_S. Show that $f^{\leq S}(x_S)$ is equal to the expected value of f conditioned on the bits x_S. (The expectation is thus over the bits of x *not* in S.)

b) Suppose f's range is $\{-1,1\}$; i.e., f is a boolean-valued function. We define the *influence of the ith coordinate on f* to be $\mathrm{Inf}_i(f) := \Pr_x[f(x) \neq f(x^{(i)})]$, where $x^{(i)}$ denotes the string x with the ith bit flipped. This measures how sensitive f is to flipping the ith coordinate. Show that

$$\mathrm{Inf}_i(f) = \sum_{S : i \in S} \hat{f}(S)^2.$$

c) Again, suppose f is a boolean-valued function. f is said to be *monotone* if $f(x) \geq f(y)$ whenever $x \geq y$. (By $x \geq y$ we mean $x_i \geq y_i$ for all i.) For example, AND, OR, and Majority are monotone functions; Parity is not monotone. Show that if f is monotone then $\mathrm{Inf}_i(f) = \hat{f}(\{i\})$ for each $i \in [n]$.

d) Once more, suppose f is boolean-valued. Suppose we pick $x \in \{-1,1\}^n$ at random and then form a string $y \in \{-1,1\}^n$ as follows: for each $i = 1 \ldots n$ independently, we set $y_i = x_i$ with probability ρ and set y_i to be a *uniformly random bit* with probability $1 - \rho$. The *noise stability of f at ρ* is defined to be

$$\mathrm{Stab}_\rho(f) := 2\Pr[f(x) = f(y)] - 1,$$

a number in the range $[-1,1]$. This measures in some way how stable f is when you flip about $\frac{1}{2}(1-\rho)$ input bits. Show that

$$\mathrm{Stab}_\rho(f) = \sum_{S \subseteq [n]} \hat{f}(S)^2 \rho^{|S|}.$$

7. A "Long Code" test. Let $\mathcal{C}$ be a set of boolean functions $\{-1,1\}^n \to \{-1,1\}$. A *local test* for $\mathcal{C}$ works as follows: Given an unknown $f : \{-1,1\}^n \to \{-1,1\}$ (as a table of values), a local test makes some q queries to f. If $f \in \mathcal{C}$ the test should accept with probability 1; if f is δ-far from every function in $\mathcal{C}$ then the test should reject with probability at least $\Omega(\delta)$. In class, we saw the BLR test, which is a 3-query local test for the class of linear functions $\mathcal{L} = \{\chi_S : S \subseteq [n]\}$. In this problem we will develop a 6-query local test for the set of "dictator functions", $\mathcal{D} = \{\chi_{\{i\}} : i \in [n]\}$; i.e., the set of n functions of the form $f(x) = x_i$.

a) Explain why a local test for a class $\mathcal{C}$ is *not* necessarily also a local test for a subclass $\mathcal{C}' \subset \mathcal{C}$. Give an example of a function that demonstrates that BLR is not a proper local test for $\mathcal{D}$.

b) Let a, b, $c \in \{-1,1\}$ be bits. Write an expression that is 1 they are "not all equal" (NAE) and is 0 if they are all equal.

c) Consider the following 3-query test, called the "NAE test", on an unknown function f: Pick 3 strings $x, y, z \in \{-1,1\}^n$ at random by choosing each triple (x_i, y_i, z_i) independently and uniformly at random from the set of strings $\{-1,1\}^3 \setminus \{(-1,-1,-1),(1,1,1)\}$; then test that $f(x)$, $f(y)$, and $f(z)$ are NAE. Show that

$$\Pr[\text{NAE test accepts}] = \frac{3}{4} - \frac{3}{4} \sum_{S \subseteq [n]} \hat{f}(S)^2 (-1/3)^{|S|}.$$

Clearly if $f \in \mathcal{D}$, the NAE test accepts with probability 1.

d) Give a 6-query local test for $\mathcal{D}$. (Hint: combine the BLR test and the NAE test.)

Remark: Actually, the NAE test is already a 3-query local test for $\mathcal{D}$; however it is a little tricky to prove this. As for the title of this problem, historically in the PCP literature the dictator functions in $\mathcal{D}$ are called Long Code codewords; the reason is that one can think of the n strings in $\mathcal{D}$ as encoding n messages from $\{-1,1\}^{\log n}$. This is an error-correcting code with *double exponential blowup*; in fact, it is the *longest* binary error-correcting code which doesn't have duplicated bits in the encoding.

8. Orthogonal decomposition. Using the "Fourier representation", any function $f : \{-1,1\}^n \to \mathbb{R}$ can be written as $f = \sum_{S \subseteq [n]} f^S$, where the decomposition has the following three properties: (i) $f^S(x)$ depends only on the coordinates of x in S; (ii) $\mathbf{E}_x[f^S(x) f^T(x)] = 0$ if $S \neq T$; (iii) $\sum_{T \subseteq S} f^T$, denoted $f^{\leq S}$, gives the conditional expectation of f conditioned on the coordinates in S. (See problem (6a).) To achieve this decomposition, we simply take f^S to be $\hat{f}(S)\chi_S$.

In this problem we establish the same kind of decomposition for general functions on product probability spaces. Specifically, let X be any finite set and let π be a probability distribution on X. We think of the n-fold product set X^n as having the product probability distribution given by π. Let $f : X^n \to \mathbb{R}$ be any function.

a) We first make condition (iii) above hold by fiat: For $S \subseteq [n]$, we *define* $f^{\leq S} : X^n \to \mathbb{R}$ to be the function depending only on the coordinates in S giving the conditional expectation; i.e., $f^{\leq S}(x_S) = \mathbf{E}[f \mid x_S]$, where the expectation is over the product probability distribution on the coordinates outside S. Now given this definition, explicitly write how we should define the functions f^S so that (i) holds and so that the equations $f^{\leq S} = \sum_{T \subseteq S} f^T$ hold. (Hint: inclusion-exclusion.)

b) Show from the definition that $\mathbf{E}_x[f^{\leq S}(x) f^{\leq T}(x)] = \mathbf{E}_x[f^{\leq (S \cap T)}(x)^2]$.

c) Now show that $\mathbf{E}[f^S(x) f^T(x)] = 0$ when $S \neq T$. (Hint: write the definition from (a) and then use (b).)

Remark: This "orthogonal decomposition" of functions f is often a good substitute for Fourier analysis when the domain is a product probability space other than $\{-1,1\}^n$.