Fall 2007 Analytical Methods in CS	**Homework 3** **Due 2008/3/13**	**Oded Regev** **Dept. of Computer Science** **Tel Aviv University**

Instructions as before.

1. **The Nisan-Szegedy bound [2]:** Let $f : \{0,1\}^n \to \mathbb{R}$ be a nonzero function of degree at most d (i.e., $\hat{f}(S) = 0$ for all S of size at least $d+1$).

   (a) Show that $\Pr[f(x) \neq 0] \geq 2^{-d}$ (this is known as the Schwartz-Zippel lemma). Hint: induction on n.

   (b) Show that if in addition f maps into $[-1,1]$ then $\mathbb{I}(f) \leq d$.

   (c) Show that if in addition f maps into $\{-1,1\}$ then f is a $d2^d$-junta.

   (d) Consider the address function $\mathrm{Addr}_k : \{0,1\}^{k+2^k} \to \{-1,1\}$ defined by

   $$\mathrm{Addr}_k(x_1, \ldots, x_k, y_1, \ldots, y_{2^k}) = (-1)^{y_x}$$

   where we think of x here as an element of $[2^k]$. Show that $\deg(\mathrm{Addr}_k) = k+1$. Conclude that the bound in (c) must be at least $2^{d-1} + d - 1$.

2. **Total influence of DNFs:**

   (a) Assume f can be expressed as a DNF of width w (i.e., each clause has at most w literals). Show that $\mathbb{I}(f) \leq 2w$. Open question: improve on the constant 2.

   (b) Deduce that width-w DNFs can be learned from random examples in time $n^{O(w/\varepsilon)}$. We will improve this in class.

3. **Unbalanced functions have a low Fourier coefficients:** Let $f : \{0,1\}^n \to \{-1,1\}$ be such that $\hat{f}(\emptyset) \notin \{-1,0,1\}$ (i.e., f is neither constant nor balanced).

   (a) Show that there must exist a nonempty S of size at most $2n/3$ such that $\hat{f}(S) \neq 0$. Hint: f^2

   (b) Optional: show that the $2n/3$ bound above is tight.

   (c) Does a similar statement hold for *balanced* functions?

4. **Bent functions:** Compute the maximum possible value of $\|\hat{f}\|_1 := \sum_S |\hat{f}(S)|$ among all functions $f : \{0,1\}^n \to \{-1,1\}$. For infinitely many n, show a function achieving this bound.

5. **Deterministically estimating Fourier coefficients:** A set $\mathcal{A} \subseteq \{0,1\}^n$ is called ε-*biased* if for x chosen uniformly from $\mathcal{A}$ and for all nonempty $S \subseteq [n]$, $|\operatorname{Exp}_x[\chi_S(x)]| \leq \varepsilon$. There is a known algorithm that on inputs ε, n, outputs an ε-biased set of size $(n/\varepsilon)^2$ in time $\mathrm{poly}(n, 1/\varepsilon)$. Use this to show how to *deterministically* estimate $\hat{f}(S)$ to within $\pm\varepsilon$ for any given S in time $\mathrm{poly}(\|\hat{f}\|_1, n, 1/\varepsilon)$ using query access to $f : \{0,1\}^n \to \mathbb{R}$. You can assume the algorithm knows $\|\hat{f}\|_1$.

6. **Close functions and concentration:** Recall that f is ε-concentrated on a family $\mathcal{S}$ if $\sum_{S \notin \mathcal{S}} \hat{f}(S)^2 \leq \varepsilon$. Show that if $\|f - g\|_2^2 \leq \varepsilon$ and g is ε-concentrated on $\mathcal{S}$ then f is 4ε-concentrated on $\mathcal{S}$.

7. **Learning functions with low $\|\hat{f}\|_1$:**

   (a) For $f : \{0,1\}^n \to \mathbb{R}$ let $L = \|\hat{f}\|_1$. Show that for any $\varepsilon > 0$, f is ε-concentrated on a set of size at most L^2/ε.

   (b) Deduce that the set of Boolean functions f with $\|\hat{f}\|_1 \leq L$ can be learned in time poly$(L, \frac{1}{\varepsilon}, n)$ using membership queries.

   (c) Define a *decision tree on parities* as a decision tree where on each node we can branch on an arbitrary parity of variables (as opposed to just a single variable in the usual definition of decision trees). Show that decision trees on parities of size L can be learned in time poly$(L, \frac{1}{\varepsilon}, n)$ using membership queries.

8. **The Goemans-Williamson MAX-CUT 0.87856-approximation algorithm [1]: (no need to hand in)** The input to the algorithm is an undirected graph $G = (V, E)$ on n vertices. The first step is to solve the following optimization problem over vector variables $v_1, \ldots, v_n \in \mathbb{R}^n$: maximize $\sum_{\{i,j\} \in E}(1 - \langle v_i, v_j \rangle)/2$ subject to all vectors being unit vectors. It is known that this optimization problem can be solved efficiently (because it is a *convex optimization problem*, and in fact a *semidefinite program*). Notice that the value of the optimum is at least the number of edges in the optimal MAX-CUT. The second step in the algorithm is to take the optimal solution $v_1, \ldots, v_n$ and to construct from it a good solution to MAX-CUT (this step is known as *rounding*). This is done as follows: choose a random unit vector $w \in \mathbb{R}^n$ uniformly and partition the vertices according to the sign of $\langle w, v_i \rangle$. Notice that each edge $\{i, j\}$ is cut with probability $\frac{1}{\pi} \arccos \langle v_i, v_j \rangle$. Hence the expected size of the cut given by the algorithm is $\frac{1}{\pi} \sum_{ij} \arccos \langle v_i, v_j \rangle$. To complete the proof, notice that this is at least $\alpha \cdot \sum_{\{i,j\} \in E}(1 - \langle v_i, v_j \rangle)/2$ where $\alpha = \frac{2}{\pi} \min_{\beta \in [-1,1]} \arccos(\beta)/(1 - \beta) \approx 0.87856$.

References

[1] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.*, 42(6):1115–1145, 1995. Preliminary version in STOC'94.

[2] N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. *Comput. Complexity*, 4(4):301–313, 1994. Preliminary version in STOC'92.