CS 787: Advanced Algorithms	**Final Exam**
Out: 12/10/07	**Due: 12/17/07**

Guidelines

- This exam consists of 4 questions. Please answer all questions. The point distribution is noted below.
- Please return your solutions to Shuchi before midnight on Monday, Dec 17.
- Collaboration is not allowed. Please refrain from consulting any sources other than the class notes for this exam.

Problems

1. **(4 points)** In class, we analyzed an estimator for the second frequency moment $\mu_2 = \sum_{i\in[n]} m_i^2$ (here m_i is the number of copies that we see of element i). Here the basic form of the estimator consisted of a counter $Z = \sum_i m_i Y_i$ ($Y_i \in_{\text{u.}a.r.} \{+1, -1\}$) and the value of the estimator was Z^2. A number of independent copies of this scheme were required to obtain a good estimate with high probability. Consider the following variant to estimate $\mu_3 = \sum_{i\in[n]} m_i^3$: Maintain a counter $Z = \sum_i m_i Y_i$, where each Y_i is picked uniformly and at random from $\{1, \omega, \omega^2\}$ (here ω is a complex cube root of unity, i.e., $\omega^3 = 1$, $\omega \neq 1$). For the purpose of this question, ignore the issues of independence in the choices of Y_i and assume that all Y_i are picked independently.
   (a) Show that $\mathbf{E}[Z^3] = \mu_3$.
   (b) Consider the estimator Re$[Z^3]$ = the real part of Z^3. Give an upper bound on the number of independent copies of this estimator needed to obtain a $1+\epsilon$ approximation of μ_3 with probability $1-\delta$.

2. **(3 points)** Consider the following variant of the Move-To-Front algorithm for the list update problem—When an element at position k in the list is accessed, *Half-MTF* moves the element to position $\lfloor k/2 \rfloor$. Prove that Half-MTF is 4-competitive for the list update problem.

3. **(8 points)** Given a graph $G = (V, E)$ with capacities $c(u, v)$ on edges $(u, v) \in E$, and a set of source-sink pairs $D \subset V \times V$ with "demands" $d(s, t)$ for every $(s, t) \in D$, the *sparsity* of a cut $(S, \bar{S})$, with $S \subseteq V$, and $\bar{S} = V \setminus S$, is defined as the ratio of the total capacity of the edges going across the cut to the total demand cut. That is,

$$\alpha(S, \bar{S}) = \frac{\sum_{(u,v)\in E\cap(S\times\bar{S})} c(u, v)}{\sum_{(s,t)\in D\cap(S\times\bar{S})} d(s, t)}$$

   In the *Sparsest Cut* problem, our goal is to find a set S with the minimum possible sparsity $\alpha(S, \bar{S})$.
   (a) Prove that the following LP is a relaxation of the Sparsest Cut problem. (In other words, for every feasible solution of the Sparsest Cut problem, we can construct a feasible solution to this LP with the same cost.) Here $\mathcal{P}_{s,t}$ for $(s, t) \in D$ denotes the set of all paths between s and t.
   (b) Note that the size of the given LP is exponential in the size of the problem. Give a polynomial-size linear program that is equivalent to this one (that is, any solution from the first can be coverted into a solution for the second with the same cost, and vice versa).
   (c) Find the dual of the above LP. Give a natural interpretation to this dual in terms of flows. You may assume that $D \cap E = \emptyset$.

$$
\begin{array}{lll}
\text{minimize} & \sum_{(u,v)\in E} c(u,v)x_{u,v} & \\
\text{subject to} & \sum_{(s,t)\in D} d(s,t)x_{s,t} \geq 1 & \\
& x_{s,t} \leq \sum_{(u,v)\in P} x_{u,v} & \forall (s,t)\in D, \forall P\in \mathcal{P}_{s,t} \\
& x_{u,v} \geq 0 & \forall u,v \in E\cup D
\end{array}
$$

(d) Consider an instance of the Sparsest Cut problem with a special node $r \in V$ called the root, and $D \subset V \times \{r\}$. (That is, the sink in every source-sink pair is r.) Prove that for any such instance, the "integrality gap" of the above LP is 1. In other words, given an optimal solution to the above LP we can come up with a solution to the Sparsest Cut problem with no larger cost.

(Note that feasible solutions for Sparsest Cut do not necessarily correspond to integral solutions for the above LP; So the term integrality gap is a misnomer here.)

4. **(5 points)** In this question we will use gradient descent to solve the following online linear optimization problem. We are given a convex set F in an n dimensional space. At step $t \in [0, T]$, our algorithm needs to output a vector $x^t \in F$, and is then given a cost function $c^t \in \mathbb{R}^n$. The cost of our algorithm at step t is the dot product between c^t and x^t, that is $\sum_{i=1}^n c_i^t x_i^t$. The goal of the algorithm is to minimize its total cost over all the steps. We compare the performance of the algorithm against the single vector $x \in F$ that minimizes $\sum_t x \cdot c^t$.

Note that this is like a typical online learning problem, except that the experts here are all vectors x in F, so a regular application of the exponential updates method is infeasible. On the other hand, we can make use of the fact that the costs or losses of different experts are related to each other through the vectors c^t.

Consider the following gradient descent algorithm for this problem. (The algorithm should be reminiscent of the perceptron algorithm discussed in class.)

(i) Start with an arbitrary vector $x^0 \in F$.

(ii) At step t, output the current vector x^t.

(iii) Upon obtaining the cost vector c^t, update the current vector as follows: $y^t = x^t - \frac{1}{\sqrt{T}} c^t$;
If $y^t \in F$, then $x^{t+1} = y^t$, otherwise, $x^{t+1} = \text{argmin}_{x\in F} |x - y^t|$.

Here $|x - y|$ for $x, y \in \mathbb{R}^n$ denotes the Euclidean distance between the two vectors, and $|x|$ denotes $|x - \mathbf{0}|$ where **0** is the all-zeroes vector.

We will bound the performance of this algorithm in terms of the following input parameters: the diameter of the set F, $\Delta = \max_{x,y\in F} |x - y|$, and the size of the cost vectors, $S = \max_t |c^t|$.

(a) For arbitrary vectors $u \in F$ and $v \in \mathbb{R}^n$, suppose that $w = \text{argmin}_{x\in F} |x - v|$. Prove that $|u - w| \leq |u - v|$. (*Hint: Use a geometric argument along with the fact that F is convex.*)

(b) Let $x^* \in F$ be the optimal solution, that is, the single vector that minimizes $\sum_t x \cdot c^t$. We will now analyze the progress over time of our vector x^t towards x^*. Using part (a), give an upper bound on $|x^{t+1} - x^*|^2$ in terms of $|x^t - x^*|^2$, the cost of the algorithm at step t, and the cost of x^* at step t.
(*Hint: Use the fact that for any two vectors a and b, $|a + b|^2 = |a|^2 + 2a \cdot b + |b|^2$.*)

(c) Use part (b) to prove that the algorithm's total cost is no more than the total cost of x^* plus $O(\sqrt{T}(\Delta^2 + S^2))$.