Problem Set no. 4

Given: May 25, 2016
Due: June 1, 2016, Box 288

Exercise 4.1 On each day, each one of n experts recommends Up or Down. Based on the advice given by the experts we need to choose Up or Down. Suppose that one of the experts is always right. Describe a deterministic algorithm that for any number T of days makes at most $\lfloor \log_2 n \rfloor$ mistakes.

(Brief review of the general setting: Let n be the number of experts. On day t, expert i, where $i \in [n]$, incurs a cost $m_i^{(t)} \in [-1, 1]$. On each day the algorithm chooses a probability distribution $\mathbf{p}^{(t)}$ on the experts, knowing the costs associated with the experts on all previous days, but not costs of day t. The costs of day t are then revealed and the algorithm's cost for day t is $\mathbf{p}^{(t)} \cdot \mathbf{m}^{(t)}$.)

Exercise 4.2 Suppose that there are only 2 experts, i.e, $n = 2$. The costs associated with the experts are $m_1^{(t)} = 1$ if t is odd, and $m_1^{(t)} = 0$, if t is even, and $m_2^{(t)} = 1 - m_1^{(t)}$. We run the multiplicative weights algorithm MW_η with $0 < \eta \leq 1/2$. Suppose that T is even. What is the cost of each expert after T days? What is the cost of MW_η? (The answer should be a simple expression that depends on T and η.) Is this consistent with the general result proved in class?

Exercise 4.3 (a) Suppose that the total number of days T is known in advance. Describe an algorithm whose total cost is greater than the cost of the best expert, with hindsight, by at most $O(\sqrt{T \ln n})$. (The difference between the cost of the algorithm and the cost of the best expert is also known as *regret*.) (Hint: Run the multiplicative weights algorithm with an appropriate choice of η.)

(b) Describe an algorithm whose regret after T days is at most $O(\sqrt{T \ln n})$ *without* knowing T in advance. (Hint: Try $T = 1, 2, 4, \ldots, 2^k, \ldots$.)

Exercise 4.4 In class, we used the multiplicative weights algorithm to learn a linear classifier. Namely, given points $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^n$ for which there exists $\mathbf{x} \in (\mathbb{R}^+)^n$, $\mathbf{1} \cdot \mathbf{x} = 1$, such that $\mathbf{a}_j \cdot \mathbf{x} \geq \varepsilon$, $\mathbf{b}_j \cdot \mathbf{x} \leq -\varepsilon$, for $j \in [m]$, finds $\mathbf{x}' \in (\mathbb{R}^+)^n$, $\mathbf{1} \cdot \mathbf{x}' = 1$, such that $\mathbf{a}_j \cdot \mathbf{x}' \geq 0$, for $j \in [m]$.
Suppose that we are again given $2m$ points $\mathbf{a}_1, \ldots, \mathbf{a}_m, \mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^n$ and are now promised that there exists $\mathbf{x} \in \mathbb{R}^n$, $\sum_{i=1}^n |x_i| = 1$ and $-1 \leq y \leq 1$, such that $\mathbf{a}_j \cdot \mathbf{x} \geq y + \varepsilon$, $\mathbf{b}_j \cdot \mathbf{x} \leq y - \varepsilon$, for $j \in [m]$. (Note that now $\mathbf{x}$ is *not* assumed to be non-negative.) Show, using a simple reduction to the previous problem, how to find $\mathbf{x}' \in \mathbb{R}^n$, $\sum_{i=1}^n |x_i'| \leq 1$ and $-1 \leq y' \leq 1$, such that $\mathbf{a}_j \cdot \mathbf{x}' \geq y'$, $\mathbf{b}_j \cdot \mathbf{x}' \leq y'$, for $j \in [m]$. (Hint: Replace each point $\mathbf{a}_j \in \mathbb{R}^n$ by a point $\mathbf{a}_j' \in \mathbb{R}^{2n+2}$, and similarly for $\mathbf{b}_j$. The last two coordinates of each $\mathbf{a}_j'$ and $\mathbf{b}_j'$ will be 1 and -1.)

Exercise 4.5 The input to the multicommodity flow problem is a directed graph $G = (V, E)$ with a capacity function $c : E \to \mathbb{R}$ defined on its edges, and k sink-source pairs $(s_1, t_1), \ldots, (s_k, t_k)$. The goal is to simultaneously send flow of commodity i from s_i to t_i, for $i \in [k]$, maximizing the sum of these flows. Flows of different commodities may share edges but the total flow on each edge cannot exceed the capacity of the edge.

Express the maximum multicommodity flow as a linear program of polynomial size. What is the number of variables and constraints in your formulation? (Hint: For every edge e introduce k flow variables $f_1(e), f_2(e), \ldots, f_k(e)$, where k is the number of commodities.)