Algorithms Under Uncertainty : Endsem
Full Marks : 40

Solutions must be brief and precise. I would seriously penalize meaningless rambles. No external resources other then 4 sheets of A4 sized handwritten notes can be consulted. Violaters will face disciplinary actions as mandated by IIIT-Delhi. No solutions will be considered valid without formal proofs.

1. (5 points) Consider the online paging problem where the online algorithm is allowed clairvoyance: We say that the algorithm is ℓ-strongly clairvoyant for any positive integer ℓ if at any time t, it knows the next ℓ *distinct* pages that will be requested.

   Describe an $\mathcal{O}(k-\ell)$-competitive deterministic online algorithm for paging assuming that the online algorithm is ℓ-strongly clairvoyant.

   Solution. The algorithm is a modification of the deterministic 1-bit MARKING done in lectures. The algorithm proceeds in phases. At the beginning of a phase, all pages in cache are 'unmarked', *except the ones that are seen by clairvoyance at this point in time*. The rest of the algorithm proceeds exactly as before - evict an unmarked page upon a cache miss and mark the new page. If there is a cache hit, just 'mark' the hit page (or do nothing if already marked). A phase ends when we see k distinct page requests.

   +2 for getting the algorithm correct

   Now for the analysis. Consider any phase $i \geq 1$ (Let us call the first phase 0). Let n_i denote the number of distinct pages that are seen by clairvoyance at the beginning of phase i, but are not in cache and $o_i = \ell - n_i$ denote the pages that are seen by clairvoyance and are inside the cache. Hence, inside a phase, the number of cache missed suffered by our algorithm can be at most $k-\ell+n_i \leq (k-\ell+1)n_i$. Now let us count the number of distinct page accesses in phases $i-1$ and i combined. Clearly, this number if k (for phase $i-1$ by definition) plus n_i (since in phase i, there are at least n_i requests which are not in the cache of the algorithm and hence are all distinct from pages accessed in phase $i-1$). Hence, the number of cache misses in OPT for phases $i-1$ and i combined is at least n_i. Hence, the total number of cache misses for OPT in all phases combined is $\frac{\sum_i n_i}{2}$. Hence the competitive ratio is $\mathcal{O}(k-\ell)$.

   +3 for correct, 0 for incorrect/missing, +1.5 for partially correct

2. (10 points) In this problem, use Yao's lemma to show that the competitive ratio of any online algorithm for online set cover problem is $\Omega(\log n)$, where n is the number of elements in the set cover instance. Recall that a set cover instance is given by a ground set X and a collection S of subsets of X. Our goal is to pick the minimum cardinality sub-collection of subsets in S such that their union is all of X. In the online set cover instance, the elements in X arrive in an online manner. Consider the following instance: we are given a complete binary tree T with n leaves (assume that n is a power of 2). The ground set X is given by the set of edges in the tree T. For each leaf $v \in T$, let P_v be the set of edges in the path from v to the root. The collection S consists of such subsets P_v for each leaf v. Now consider the following random input: we pick a leaf uniformly at random and let $e_1, \cdots e_k$ be

the edges in the tree T ordered from the root r to the leaf v. We present the edges in this order to the (deterministic) online algorithm. Use this random input to show that the competitive ratio of any randomized algorithm for online set cover must be $\Omega(\log n)$.

a) (2 points) Show that OPT is always 1.

Solution. This is trivial. Since OPT knows which leave has been picked, it knows all edges that are requested and hence can just choose the corresponding path.

(+2 for correct, 0 for anything else)

b) (6 points) Define indicator random variables X_i for every edge e_i which indicates whether the algorithm buys a new set (path) to cover this edge. Show that the expectation of any of these variables is a constant.

Solution. Let us consider any edge e_i in the ordering $e_1, e_2, \cdots e_k$ as seen by the deterministic algorithm. Now, we want to find the probability that the algorithm will need to choose a new path (set) for e_i. Considering the deterministic algorithm to behave reasonably (that is, not buy a set for a new edge if there is already a path covering it in its partial solution), this will happen if none of the paths selected for covering $e_1, e_2, \cdots e_{i-1}$ is the path that contains e_i. Now consider any index $j < i$. By the property that the tree is complete binary, there are 2^{j-i+1} possible root to leaf paths that contain e_j as well as e_i. The input distribution will hit one of these uniformly at random. Hence, the chance that the path chosen by the deterministic algorithm to cover e_j is the one which also contains e_i is at most $1/2^{j-i+1}$. Hence the probability that e_i is already covered before it arrives, by Union Bound is

$$\sum_{j=i-1}^{k-1} \frac{1}{2^{j-i+1}} \leq \frac{1}{2}$$

Hence, the chance that a new path needs to be bought for e_i is at least 1/2. **+6 for all arguments correct, +3 for partial , 0 for incomplete/wrong/vague arguments**

c) (2 points) Use the above to prove the final theorem. **Solution.**

Using the above claim, it can be seen that the expected number of sets bought by any deterministic algorithm for the random input sequence given in the problem is at least $\log k/2$. OPT on the othr hand is 1. Hence, the competitive ratio of any deterministic algorithm on the random input sequence is at least $k/2 = \log n/2$. Hence, by Yao's principle, any randomized algorithm will have at least this competitive ratio.

+2 for the complete argument, +1 for partial, 0 for incorrect/missing

3. (10 points) Recall the 0/1 prediction game with expert advice. For each day $t = 1, 2, \cdots T$, you receive the opinion of n experts which is either 0 or 1. Based on these advice, you take a decision and observe the reality, thereby either incurring or not incurring a mistake. The goal is to make as few mistakes as possible and we compare ourselves against the best expert (one of who makes fewest mistakes). Here is a variation on the deterministic Weighted Majority algorithm that we had studied to solve this problem, designed to make it more adaptive.

   (a) Each expert begins with weight 1 (as before).

   (b) At every step, we predict the result of a weighted-majority vote of the experts (as before).

   (c) At every step, for every expert that makes a mistake, we penalize it by dividing its weight by 2, but only if its weight was at least 1/4 of the average weight of experts.

   Prove that in any contiguous block of steps (e.g., the 51st step through the 77th step), the number of mistakes made by the algorithm is at most $O(m + \log n)$, where m is the number of mistakes made by the best expert in that block, and n is the total number of experts. (Hint: As always, try to figure out upper and lower bounds for total weight W. Note that you need to prove the claim for an arbitrary block and hence cannot start with the assumption that the sum of weights is n at the beginning of the block)

   Solution. Consider any block between times $[T_s, T_e]$. Let W_t denote the total weight of all experts at the beginning of time point t (before the prediction). Let $m_{\mathcal{A}}$ be the number of mistakes made by the algorithm in this block and m the number of mistakes by the best expert.

   Suppose the algorithm makes a wrong prediction at any time point t. We want to estimate by how much the total weight goes down. Since the algorithm goes by weighted majority, we know that the total weight of experts who were wrong at round t is at least $W_t/2$. Suppose the number of wrong experts whose weight is at least $W_t/4n$ is less than $n/4$. But then they contribute $W_t/16$ to the total weight. Hence the number of wrong experts required to cover the rest of the weight, $W_t/2 - W_t/16 = 7W_t/16$ would be at least 7n/4 (since each can contribute less than $W_t/4n$). This is a contradiction since the total number of experts is n. Hence, whenever we make a mistake, the weights of at least $n/4$ experts, each with weight at least $W_t/4n$ reduces by half. Hence, the total reduction is at least $W_t/32$. Thus $W_t \leq \frac{31}{32} W_{t-1}$. Hence, we can conclude

$$W_{T_e} \leq W_{T_s} \left(\frac{31}{32}\right)^{m_{\mathcal{A}}}$$

   Now we find a suitable lower bound for W_{T_e}. Since the number of mistakes of the best expert is m, the weight of such an expert can be halved *at most* m times (at most because we do not always penalize for a mistake). We also claim that the weight of the expert at T_e cannot be less than $W_{T_S}/8n$. Suppose it is. Now every expert starts with a weight of 1. Hence, at some point, its weight must have been halved to reach a value less than $W_{T_S}/8n$ - let that time point be T'. But what was its weight then at T' - clearly it was strictly less than $W_{T_S}/4n \leq W_{T'}/4n$ since weights do not increase over time. Hence, the total weight

$$W_{T_e} \geq \frac{W_{T_S}}{8n} \left(\frac{1}{2}\right)^{m}$$

   Combining the above two inequalities and taking logs give the result.

   +10 for writing all the steps clearly . +5 for at least getting one bound for total weight correct. 0 for wrong/missing.

4. (10 points) Consider the online learning framework where at each time t, we receive a convex loss function $f_t : \mathbb{R}^d \to \mathbb{R}$. Assume that the function $f_t(w)$ is minimized at $w = 0$. You are given an online learning algorithm $\mathcal{A}$ which has regret $R(T)$, where T is length of the time horizon. Assume that $R(T)$ is an increasing function of T. Now you would like to design another online learning algorithm to have the following property: let w_t be the point selected by the algorithm at time t. If it so happens that $f_t(w_t) = 0$, then the next point w_{t+1} must be same as w_t. Show how to use $\mathcal{A}$ as black box to get an online learning algorithm which has this property and whose regret is bounded by $R(T)$.

   Solution. Let us call the algorithm which we design algorithm $\mathcal{B}$. We will use $\mathcal{A}$ given in the problem as a subroutine inside $\mathcal{B}$. At time 0, $\mathcal{B}$ just calls $\mathcal{A}$ and outputs the answer given by $\mathcal{A}$ as w_0. Now at any time point $t \geq 0$, $\mathcal{B}$ does one of the following. If $f_t(w_t) = 0$, then just set $w_{t+1} = w_t$. Else, it send f_t to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ predicts as w_{t+1}.

   To prove that the regret of $\mathcal{B}$ is at most $R(T)$, let us define some notations. We define $Z = \{0 \leq t \leq T : f_t(w_t) = 0\}$ - that is subset of time indices where $f_t(w_t) = 0$ and $NZ = T \setminus S$. Now, note that the algorithm $\mathcal{A}$ only receives the functions f_t for $t \in NZ$. Hence, the regret of $\mathcal{A}$ in the above procedure is $R(|NZ|)$. Let $w^\star$ be the optimal point that minimizes the sum of f_t's for all $t \in NZ$. Finally, let $\widehat{w}$ be the minimizer of the sum of all the functions $f_t, t = 1, 2, \cdots T$. Now we shall bound the regret of our actual algorithm $\mathcal{B}$. We know from above,

$$
\begin{aligned}
&\sum_{t \in NZ} f_t(w_t) \leq \sum_{t \in NZ} f_t(w^\star) + R(|NZ|) \\
&\implies 0 + \sum_{t \in NZ} f_t(w_t) \leq \sum_{t \in NZ} f_t(w^\star) + R(|NZ|) \\
&\implies \sum_{t \in Z} f_t(w_t) + \sum_{t \in NZ} f_t(w_t) \leq \sum_{t \in NZ} f_t(\widehat{w}) + R(|NZ|) \\
&\implies \sum_{t \in T} f_t(w_t) \leq \sum_{t \in Z} f_t(\widehat{w}) + \sum_{t \in NZ} f_t(w^\star) + R(|NZ|) \\
&\implies \sum_{t \in T} f_t(w_t) \leq \sum_{t \in T} f_t(\widehat{w}) + R(|NZ|) \\
&\implies \sum_{t \in T} f_t(w_t) \leq \sum_{t \in T} f_t(\widehat{w}) + R(T)
\end{aligned}
$$

   we use all the properties and assumptions throughout the series of inequalities - that $f_t(w) \geq 0$ for all w, t, that $w^\star$ is the minimizer for sum of functions f_t over $t \in NZ$ and the final inequality uses the fact that $R(T)$ is a non-decreasing function.

   +4 for getting the algorithm completely correct, +2 for somewhat correct idea, 0 for completely wrong/missing. +6 for complete proof, +3 for right direction but incomplete, 0 for wrong/missing.

5. (5 points) Prove the the Entropical regularizer as defined in the lectures is $1/\eta$-strongly convex under ℓ_1 inside the d-dimensional probability simplex. You may need to use the following inequality : for two points p and q inside the probability simplex of dimension d, $\sum_{i=1}^{d} p_i \ln\left(\frac{p_i}{q_i}\right) \geq ||p - q||_1^2$

   Solution. Recall the Entropical regularizer is given by

$$R(v) = \frac{1}{\eta} \sum_{i=1}^{d} v_i \log v_i$$

and we would be interested in the domain where $v \in R^d$ and $\sum_{i=1}^{d} v_i = 1$. Now recall a function is $1/\eta$-strongly convex under ℓ_1-norm inside the prob-simplex if for any u, v in the simplex,

$$F(u) = F(v) + \langle \nabla F(v), u - v \rangle + \frac{1}{2\eta} ||u - v||_1^2$$

$$\begin{aligned}
&F(u) - F(v) - \langle \nabla F(v), u - v \rangle \\
&= \frac{1}{\eta} \left(\sum_{i=1}^{d} u_i \log u_i - \sum_{i=1}^{d} v_i \log v_i - \frac{1}{\eta} \sum_{i=1}^{d} (u_i - v_i)(1 + \log v_i) \right) \\
&= \frac{1}{\eta} \left(\sum_{i=1}^{d} u_i \log u_i - \sum_{i=1}^{d} u_i \log v_i) + \sum_{i=1}^{d} u_i - v_i \right) \\
&= \frac{1}{\eta} \left(\sum_{i=1}^{d} u_i \log \left(\frac{u_i}{v_i} \right) \right) \\
&\geq \frac{1}{2\eta} ||u - v||_1^2
\end{aligned}$$

where the second last inequality uses that both u and v are inside the probability simplex and the last inequality follows from the one in the hint.

+5 for the whole correct calculation. +3 for missing reasons. 0 for wrong calculations