MS&E 111X/MS&E211X Suggested Course Project III: Online Linear Programming and Resource Allocation

October 24, 2021

In resource allocation described in class, we consider a linear program of the form

$$\begin{array}{ll} \text{maximize}_{\mathbf{x}} & \sum_{j=1}^{n} \pi_j x_j \\ \text{s.t.} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \ \forall\ i = 1, 2, ..., m \\ & 0 \leq x_j \leq 1, \ \forall\ j = 1, ..., n; \end{array} \tag{1}$$

where π_j is the gain to allocate a combination of goods/resources to bidder j (we can think of this as the bid of bidder j), a_{ij} is the requested quantity of good/resource i by bidder j, and b_i is the total available quantity of good/resource i; see [3] and references therein. For simplicity, in this project we assume that a_{ij} is either 0 or 1.

The classical offline LP algorithm would compute the full optimal solution $\mathbf{x}$ in one go, while the online algorithm would compute the solution sequentially x_1, then x_2,.... Specifically, when computing the decision variables x_1 to x_k, we don't consider information associated with x_{k+1} and beyond. We can interpret the online model as following. At each time k, one order comes in. Then, the decision maker needs to decide whether fully accept ($x_j = 1$) the order, partially accept it ($x_j \in (0, 1)$) or reject it ($x_j = 0$). The decision is irrevocable and the costumer will leave before time $k + 1$. The goal of the decision maker is to maximize the reward or welfare, which is represented by the objective function $\sum_{j=1}^{n} \pi_j x_j$.

In this course project, you are asked to study and explore some theories of online linear and convex programming and perform computational observations on some simulated or real data.

The online algorithm described in [3] is as follows: suppose there is a reliable estimate that there will be a total of n bidders in the market; then we wait for the first k bidders to arrive, solve the resulting partial linear program:

$$
\begin{array}{llll}
\text{(SLPM):} & \text{maximize}_{x_1,\dots,x_k} & \sum_{j=1}^{k} \pi_j x_j & \\
& \text{s.t.} & \sum_{j=1}^{k} a_{ij} x_j \le \frac{k}{n} b_i, \ \forall\ i = 1,2,\dots,m, & \\
& & 0 \le x_j \le 1, \ \forall\ j = 1,\dots,k. &
\end{array}
$$

and then use the *dual prices*, say $\bar{\mathbf{y}}^k$, of the (partial) LP for future online decisions:

$$x_j = 1, \text{ if } \pi_j > \mathbf{a}_j^T \bar{\mathbf{y}}^k \text{ and there are remaining goods left;} \quad \text{and } x_j = 0, \text{ otherwise.}$$

The interpretation of this is that we allow the allocation to bidder j if the their bid is higher than the value, calculated after k bids, of the goods they require.

You may run the online and offline auctions using simulated bidding data with $m = 10$ and $b_i = 1,000$ for all i. Fix a ground truth price vector $\bar{\mathbf{p}} > 0$. One way to generate a sequence of random bids, $k = 1, 2, ...$, is as follows: generate a vector $\mathbf{a}_k$ whose each entry is either zero or one at random, then let $\pi_k = \bar{\mathbf{p}}^T \mathbf{a}_k + randn(0, 0.2)$ where $randn(0, 0.2)$ represents the Gauss random variable with zero mean and variance 0.2 in Matlab.

Question 1: Let $n = 10,000$. Then run the one-time online learning (SLPM) algorithm using the same simulated bidding data above based on three different sizes of $k = 50, 100, 200$ to see how sensitive the ratio of the generated revenue over the offline revenue is to k. Note that we keep the estimated prices $\bar{\mathbf{y}}^k$ fixed in this case. What is the trade-off between choosing large and small k?

Question 2: Now let us dynamically update the dual prices at time points $k = 50, 100, 200, 400, 800...$ and use the prices to make decision for the immediate subsequent period. How does the dynamic learning perform on the revenue? Do the dual price vectors $\bar{\mathbf{y}}^k$ generated from the online LP model approach the ground truth vector $\bar{\mathbf{p}} > 0$? Explain your observations and findings. Again, use the same data generated in Question 1.

The online algorithm described above does not use information on how much good inventory remains for allocation in the decision process. One online approach that uses the information of inventory (See [6]) is the Action-history-dependent Learning Algorithm, which means this algorithm uses historic actions to update the dual price, i.e., at each time point $k \ge 2$, decide the value of x_k:

$$x_k = 1, \text{ if } \pi_k > \mathbf{a}_k^T \bar{\mathbf{y}}^{k-1} \text{ and there are remaining goods left;} \quad \text{and } x_k = 0, \text{ otherwise,}$$

update remaining resources:

$$b_i^{(k)} = b_i^{(k-1)} - a_{ik} x_k \text{ for i = 1,2,...,m,}$$

and, then update the dual price by solving the linear programming:

$$
\begin{array}{lll}
\text{maximize}_{x_1,\ldots,x_k} & \sum_{j=1}^{k} \pi_j x_j & \\
\text{s.t.} & \sum_{j=1}^{k} a_{ij} x_j \le \frac{k}{n-k} b_i^{(k)}, \ \forall\, i = 1, 2, \ldots, m, \\
& 0 \le x_j \le 1, \ \forall\, j = 1, \ldots, k, &
\end{array} \tag{2}
$$

and, just compute the initial dual price by (2) when $k = 1$.

Question 3: Solve the LP by the Action-history-dependent Learning Algorithm. At each time point $k \ge 2$, compute $\sum\limits_{j=1}^{k} r_j x_j - \frac{k}{n} OPT$ for the Action-history-dependent Learning Algorithm, where OPT is the optimal of the offline problem (1). Then, do the same thing for the algorithm in Question 2. Which algorithm has better performance? Provide your result and intuition. Again, use the same data generated in Question 1.

Question 4: In [6], note that if $(\pi_j, \mathbf{a}_j) \sim (\pi, \mathbf{a}),\ j = 1, 2, \ldots, n$ is a sequence of i.i.d. random vectors, The problem (1) is closely related to

$$
\begin{array}{ll}
\text{minimize}_{\bar{\mathbf{y}}} & \mathbf{d}^T \bar{\mathbf{y}} + \mathbb{E}\left(\pi - \mathbf{a}^T \bar{\mathbf{y}}\right)^+, \\
\text{s.t.} & \bar{\mathbf{y}} \ge 0,
\end{array} \tag{3}
$$

where $\mathbf{d} = \mathbf{b}/n$ and $(\cdot)^+ = \max\{\cdot, 0\}$. Identify whether (3) is a convex optimization problem and find the connection between (1) and (3). You are not required to mathematically prove your claim and only need to intuitively interpret it.

(Hint: Write down the dual problem of (1).)

Question 5 (Optinal): Although both dynamic online algorithms above can provide good results, one drawback of them is that one has to solve a large scale LP to update the dual price. When n is large, the computation is both time- and memory- consuming. A way to fix it is to utilize the idea of Stochastic Gradient Descent.

Under some mild conditions, the objective in (3) is differentiable and the derivative is

$$
\mathbf{f}(\bar{\mathbf{y}}) = \mathbb{E}\left(\mathbf{d} - \mathbf{a}\mathbb{I}_{\{\pi > \mathbf{a}^T \bar{\mathbf{y}}\}}\right).
$$

Assume at each time step k, $\mathbb{E}(\mathbf{f}(\pi, \mathbf{a}))$ can be approximated by $\mathbf{f}(\pi_k, \mathbf{a}_k)$. Using Stochastic Gradient Descent, derive a new online Algorithm with $\bar{\mathbf{y}}^0 = 0$ as the initialization. For here, set $\beta = \sqrt{k}$ at each time point, which is different from the lecture notes where β is fixed. How does the dynamic learning algorithm perform on the revenue? Do the approximated dual price vectors $\bar{\mathbf{y}}$ generated from this gradient descent version of online LP algorithm converge to the true vector $\bar{\mathbf{p}} > 0$? Explain your observations and findings with the same data in Question 1.

References

[1] Shipra Agrawal and Nikhil R. Devanur. Fast Algorithms for Online Stochastic Convex Programming http://arxiv.org/abs/1410.7596, SODA2015

[2] S. Agrawal, E. Delage, M. Peters, Z. Wang and Y. Ye. A Unified Framework for Dynamic Prediction Market Design. Operations Research, 59:3 (2011) 550-568.

[3] S. Agrawal, Z. Wang and Y. Ye. A Dynamic Near-Optimal Algorithm for Online Linear Programming. http://arxiv.org/abs/0911.2974; to appear in Operations Research.

[4] Anupam Gupta and Marco Molinaro How the Experts Algorithm Can Help Solve LPs Online. https://arxiv.org/abs/1407.5298, 2014.

[5] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, Berthold Vöcking. Primal Beats Dual on Online Packing LPs in the Random-Order Model. https://arxiv.org/abs/1311.2578, STOC 2014.

[6] X. Li and Y. Ye. Online Linear Programming: Dual Convergence, New Algorithms, and Regret Bounds. https://arxiv.org/abs/1909.05499

[7] M. Peters, A. M-C. So and Y. Ye. Pari-mutuel Markets: Mechanisms and Performance. *The 3rd International Workshop On Internet And Network Economics*, 2007.