A Primal-Dual Online Algorithm for Online Matching Problem in Dynamic Environments

Yu-Hang Zhou, Peng Hu, Chen Liang, Huan Xu, Guangda Huzhang, Yinfu Feng, Qing Da, Xinshang Wang, An-Xiang Zeng

Alibaba Group, Hangzhou, China

{zyh174606,sylar.hp,liangchen.lc,huan.xu,guangda.hzgd,yinfu.fyf,daqing.dq,xinshang.w}@alibaba-inc.com, renzhong@taobao.com

Abstract

Recently, the online matching problem has attracted much attention due to its wide application on real-world decision-making scenarios. In stationary environments, by adopting the stochastic user arrival model, existing methods are proposed to learn dual optimal prices and are shown to achieve a fast regret bound. However, the stochastic model is no longer a proper assumption when the environment is changing, leading to an optimistic method that may suffer poor performance. In this paper, we study the online matching problem in dynamic environments in which the dual optimal prices are allowed to vary over time. We bound the dynamic regret of online matching problem by the sum of two quantities, including a regret of online max-min problem and a dynamic regret of online convex optimization (OCO) problem. Then we propose a novel online approach named Primal-Dual Online Algorithm (PDOA) to minimize both quantities. In particular, PDOA adopts the primal-dual framework by optimizing dual prices with the online gradient descent (OGD) algorithm to eliminate the online max-min problem's regret. Moreover, it maintains a set of OGD experts and combines them via an expert-tracking algorithm, which gives a sublinear dynamic regret bound for the OCO problem. We show that PDOA achieves an $O(K\sqrt{T(1+P_T)})$ dynamic regret where K is the number of resources, T is the number of iterations and P_T is the path-length of any potential dual price sequence that reflects the dynamic environment. Finally, experiments on real applications exhibit the superiority of our approach.

Introduction

The online matching problem has been widely applied in many real-world tasks, e.g., internet advertising, resource allocation, and dynamic pricing. There are M potential items and K resources. The k-th resource possesses a budget B_k representing the maximum amount that can be consumed. The protocol is as follows: in each round, the t-th user is revealed with a reward vector $\boldsymbol{r}_t \in \mathbb{R}^M$ and a resource consumption matrix $\boldsymbol{C}_t \in \mathbb{R}^{M\times K}$. Denote r_{ti} as the i-th component of $\boldsymbol{r}_t$, C_{tik} as the i-th row and the k-th column component of $\boldsymbol{C}_t$. When assigning the i-th item to the t-th user, the reward and the consumption of the k-th resource can be represented by r_{ti} and C_{tik}, respectively. The learner chooses an item i_t according to the historical information $\{(\boldsymbol{r}_\tau, \boldsymbol{C}_\tau)\}_{\tau=1}^t$, obtaining the corresponding reward and consuming the corresponding resources. Given the total number T of arrival users, our goal is to maximize the cumulative reward while satisfying the budget constraints.

When all parameters of T users, i.e., $\{(\boldsymbol{r}_t, \boldsymbol{C}_t)\}_{t=1}^T$ are available in advance, the matching problem can be formulated as an offline linear program (LP). It is easy to see that the solution of offline LP is an upper bound of its online version since the online learner has to make irrevocable decisions without complete information about subsequent users. Moreover, the ratio of the cumulative reward obtained by an online method to that obtained by offline LP is called Competitive Ratio (C.R.). To make the online problem more feasible, researchers introduce several assumptions regarding user arrival fashion. In general, there are two common user arrival models, i.e., the adversarial model and the stochastic model. The adversarial model assumes that there is an adversary who knows the strategy of the algorithm and generates the worst user sequence accordingly, which is usually pessimistic in practice. The stochastic model, on the contrary, assumes that the users are drawn from a stationary distribution, which is often in accordance with the real data.

Adopting the stochastic user arrival model, most existing methods are based on the primal-dual framework, where dual optimal prices are learned by solving a fractional matching problem on the revealed users and are used for subsequent assignments. In stationary environments, these methods achieved a near-optimal solution. Recently, (Li, Sun, and Ye 2020) proposed online algorithms by establishing the dual convergence result. Rather than learning dual prices on fractional data, they run an OGD algorithm on the dual problem to update the dual prices online. Since the dual optimal prices are stable in stationary environments, their proposed online methods converge at a fast rate.

However, the stochastic model may derive optimistic methods that suffer from poor performance when the environment is changing. In particular, the dual optimal prices can be non-stable, which leads to the failure of the dual convergence result. Alternatively, minimizing the dynamic regret with respect to any feasible sequence of dual prices is a better choice in a dynamic environment. Similar works have been established in OCO problems (Zhang, Lu, and Zhou 2018; Zhao et al. 2020a).

In this paper, we study the online matching problem in dynamic environments where the dual optimal prices are allowed to vary over time. The main contributions of this paper are summarized as follows:

- We bound the dynamic regret of online matching problem by the sum of two quantities, including a regret of online max-min problem and a dynamic regret of OCO problem.
- A novel online approach, named Primal-Dual Online Algorithm (PDOA), is proposed to minimize both quantities. In particular, PDOA adopts the primal-dual framework by optimizing the dual prices with OGD to eliminate the online max-min problem's regret. Moreover, it maintains a set of OGD experts and combines them via an expert-tracking algorithm, which gives a sublinear dynamic regret bound for the OCO problem.
- The proposed PDOA achieves an $O(K\sqrt{T(1+P_T)})$ dynamic regret where K is the number of resources, T is the number of iterations and P_T is the path-length of any potential sequence of dual prices that reflect the dynamic environment. It is markable that with a sublinear path-length, our approach achieves a sublinear dynamic regret.
- The experiments on real applications exhibit the superiority of our approach. In the application from the international online retailer, PDOA completes 11.7% more tasks than OGD and achieves comparable total purchase numbers. Moreover, in the application from the e-commerce grocery retailer, PDOA increases 9.5% total GMV and reduces 3.3% inventory loss than OGD.

Related Works

Online Matching In the past decades, the online matching problem has attracted a surge of attention due to the increasing demand for social applications (Mehta 2013). There are two kinds of different assumptions on user arrival models, i.e., the adversarial model and the stochastic model. Existing methods fall into two categories accordingly.

The adversarial user arrival model assumes that there is an adversary who knows the strategy of the algorithm and generates the worst user sequence accordingly. It is a proper model when the user sequence is totally unpredictable. Under the small bid assumption, it has been proved that the upper bound of the C.R. of algorithms designed for the adversarial model is $1-1/e$ (Karp, Vazirani, and Vazirani 1990). Several algorithms, e.g., Perturbed Greedy (Aggarwal et al. 2011) and Balance Algorithm (Kalyanasundaram and Pruhs 2000) have been proposed to achieve the optimal value. For the most general bid problems, $1/2$-C.R. is a long-standing barrier achieved by the trivial greedy algorithm. Recently, (Fahrbach et al. 2020) proposed a novel online algorithm to break the barrier, where the most interesting ingredient is a subroutine called online correlated selection (OCS). The adversarial model is often pessimistic in practice.

The stochastic user arrival model assumes that users are drawn from a stationary distribution or arrive at a random order, which is often in accordance with the real data. Most existing methods based on the stochastic model follow the primal-dual framework, where dual optimal prices are learned by solving a fractional matching problem on the revealed users and used for assigning items to subsequent users. In the stationary environment, these methods achieved a near-optimal solution (Agrawal, Wang, and Ye 2014; Buchbinder and Naor 2009; Feldman et al. 2010). Moreover, (Agrawal and Devanur 2014; Agrawal, R, and Li 2016; Li, Sun, and Ye 2020) proposed online algorithms by establishing the dual convergence result. Rather than learning dual prices on fractional data, they run an OGD algorithm on the dual problem to update dual prices online. Since the dual optimal prices are stable in stationary environments, their proposed methods converge to the optimal solution at a fast rate, concretely, $O(\log T \log\log T)$. Moreover, an idea to generalize online matching is to increase the capacities of vertices. For instance, the online bipartite matching problem has been studied extensively in the literature (Aggarwal et al. 2011; Kesselheim et al. 2013; Huzhang et al. 2017).

Although algorithms for the stochastic model are near-optimal when users arrive randomly as expected, it is notable that they could have a sharply degenerated performance when the user sequence is adversarial (Esfandiari, Korula, and Mirrokni 2015). Since both adversarial and stochastic models have significant limitations, (Zhou et al. 2019) considered a novel user arrival model where users are drawn from a drifting distribution. They proposed a new approach named RDLA to deal with such an assumption, in which the distributionally robust optimization (DRO) technique is leveraged to learn dual prices. Due to the feature of DRO, the RDLA is more suitable for the case where there exists some spikes in the user sequence and can not be expected to perform better when the environment changes gradually.

Online Convex Optimization Online convex optimization (OCO) has become a well-established learning framework both on theory and practice (Hazan 2019). In static environment, OGD achieves an $O(\sqrt{T})$ regret bound for general convex functions. When the functions have a property of strong convexity, the regret bound of OGD becomes $O(\log T)$ (Shalev-Shwartz et al. 2011). For exp-convex functions, a second-order method named Online Newton Step has an $O(n\log T)$ regret bound, where n is the dimension of functions (Hazan, Agarwal, and Kale 2007).

When the environment is changing, the traditional regret is no longer a suitable measure for online learners, since it compares the learners to a static point. To break this limitation, researchers have introduced a new measure called dynamic regret recently. There are two different dynamic regrets, i.e., the general dynamic regret and the restricted dynamic regret. The general dynamic regret was first introduced by (Zinkevich 2003) where the cumulative reward of the learner is compared to any sequence of comparators. The restricted dynamic regret, on the other hand, compares the cumulative reward of the learner to the restricted comparator sequence consisting of local minimizers of online functions (Besbes, Gur, and Zeevi 2015). Since the general dynamic regret includes the static regret and the restricted dynamic regret as special cases, minimizing the general dynamic regret can adapt to both stationary and dynamic environments. In this paper, we focus on the general dynamic regret.

The first study investigating the general dynamic regret

is (Zinkevich 2003). They introduced the definition of path-length P_T that reflects the dynamic environment and showed that the OGD with a constant step size achieves a dynamic regret of $O(\sqrt{T}(1+P_T))$. (Zhang, Lu, and Zhou 2018) established a lower bound of $\Omega(\sqrt{T(1+P_T)})$ and developed a novel online method named Ader to achieve the optimal dynamic regret. Furthermore, (Zhao et al. 2020a) studied the bandit convex optimization in dynamic environments and proposed an algorithm achieving $O(T^{3/4}(1+P_T)^{1/2})$ and $O(T^{1/2}(1+P_T)^{1/2})$ dynamic regret respectively for the one-point and two-point feedback models. To exploit the smoothness condition, (Zhao et al. 2020b) proposed algorithms leveraging smoothness and replaced the dependence on T in the dynamic regret by problem-dependent quantities that are $o(T)$ and much tighter in benign environments.

It is notable that the protocol of online matching problem is different from OCO in some places. For the online matching problem, a key challenge is that the learner is required to allocate items under the long-term constraints of budgets. Moreover, in the t-th round, the information of the t-th user is accessible to the learner before making a decision, following the fashion of online linear programming.

Preliminaries

We introduce some notations and definitions. There are M potential items and K resources. The k-th resource possesses a budget B_k. Denote $\boldsymbol{B} = [B_1; ...; B_k] \in \mathbb{R}^K$ as the vector of budgets. In each round, the t-th user is revealed with a reward vector $\boldsymbol{r}_t \in \mathbb{R}^M$ and a resource consumption matrix $\boldsymbol{C}_t \in \mathbb{R}^{M\times K}$. According to the historical information $\{(\boldsymbol{r}_\tau, \boldsymbol{C}_\tau)\}_{\tau=1}^t$, the online learner chooses a one-hot decision vector $\hat{\boldsymbol{x}}_t \in \mathbb{R}^M$ where the i_t-th component equals to 1 representing the i_t-th item is assigned to the t-th user. The reward and the amount of consumed resources in the t-th round can be calculated as $\boldsymbol{r}_t^\top \hat{\boldsymbol{x}}_t$ and $\boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t$, respectively. Therefore the online learner's cumulative reward is given by

$$R(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T) = \sum_{t=1}^T \Big(\boldsymbol{r}_t^\top \hat{\boldsymbol{x}}_t \mathbb{I}[\sum_{\tau=1}^t \boldsymbol{C}_\tau^\top \hat{\boldsymbol{x}}_\tau \le \boldsymbol{B}] \Big) \tag{1}$$

where $\mathbb{I}(\cdot)$ is the indicator function and T is the total number of arrival users.

When all parameters of T users, i.e., $\{(\boldsymbol{r}_t, \boldsymbol{C}_t)\}_{t=1}^T$ are available in advance, the matching problem can be relaxed as an LP problem:

$$\max_{\boldsymbol{x}_1,...,\boldsymbol{x}_T} \Big\{ \sum_{t=1}^T \boldsymbol{r}_t^\top \boldsymbol{x}_t, \ \text{ s.t.} \sum_{t=1}^T \boldsymbol{C}_t^\top \boldsymbol{x}_t \le \boldsymbol{B} \Big\} \tag{2}$$

Denote $(\boldsymbol{x}_1^*, ..., \boldsymbol{x}_T^*)$ as the optimal solution of (2) and let $R^* = R(\boldsymbol{x}_1^*, ..., \boldsymbol{x}_T^*)$ be the cumulative reward of the optimal solution, then the regret for the online matching problem can be defined as

$$\text{Regret}(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T) \triangleq R^* - R(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T) \tag{3}$$

With the goal of minimizing the regret (3), the primal-dual framework has been adopted by many algorithms, where the dual prices are learned by optimizing the dual problem of the original LP and then used for the online assignment. Specifically, the dual form of LP (2) is

$$\min_{\boldsymbol{\lambda}\ge \boldsymbol{0}} \ \boldsymbol{B}^\top \boldsymbol{\lambda} + \sum_{t=1}^T \|\boldsymbol{r}_t - \boldsymbol{C}_t \boldsymbol{\lambda}\|_\infty \tag{4}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^K$ is the dual variable. The k-th component of $\boldsymbol{\lambda}$, denoted by λ_k, represents the amount of increased reward when one unit of the k-th resource is allowed to be added to its budget, which is the reason why $\boldsymbol{\lambda}$ is called *dual price* in the literature of online matching problem. Moreover, $\boldsymbol{\lambda}$ can be learned by solving a fractional dual problem with the revealed information (Agrawal, Wang, and Ye 2014) or adopting OCO algorithms on the dual problem (4) directly (Li, Sun, and Ye 2020). Let $\hat{\boldsymbol{\lambda}}_t$ be the dual prices used in the t-th round, with the help of strong duality and K.K.T conditions we derive the decision rule from dual prices as

$$\hat{x}_{ti} = \begin{cases} 1, \ i = \arg\max\{r_{ti} - \sum_{k=1}^K c_{tik}\hat{\lambda}_{tk}\} \\ 0, \ \text{otherwise.} \end{cases} \tag{5}$$

where $\hat{\lambda}_{tk}$ is the k-th component of $\hat{\boldsymbol{\lambda}}_t$ and $\hat{x}_{ti}$ is the i-th component of $\hat{\boldsymbol{x}}_t$.

The Proposed Algorithm

Assumptions

In this paper, we adopt the following assumptions: There exist constants $\overline{r}, \overline{c}, \overline{\lambda}, \overline{b}, \underline{b} \in \mathbb{R}^+$ such that $r_{ti} \le \overline{r}$, $c_{tik} \le \overline{c}$, $\|\boldsymbol{\lambda}\|_\infty \le \overline{\lambda}$ and $b_k = B_k/T \in [\underline{b}, \overline{b}]$. Moreover, we also assume that $\underline{b} \le \overline{c}$.

Motivations

We first reduce the online matching problem (2) to the online max-min problem. Then by introducing dual comparator sequences, we bound the dynamic regret (3) with two quantities, including a regret of the online max-min problem and a dynamic regret of the OCO problem. The proposed approach is designed to minimize both quantities.

The online matching can be reduced to an online max-min problem by introducing the function

$$L_t(\boldsymbol{x}_t, \boldsymbol{\lambda}) \triangleq \boldsymbol{r}_t^\top \boldsymbol{x}_t + \boldsymbol{\lambda}^\top \big(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \boldsymbol{x}_t\big) \tag{6}$$

To make the Slater condition hold for LP (2), we assume there exists a *null item* that gains no reward and consumes no resources. Then, by using the strong duality we have

$$\begin{aligned} R^* &= \max_{\boldsymbol{x}_1,...,\boldsymbol{x}_T} \min_{\boldsymbol{\lambda}\ge\boldsymbol{0}} \Big\{ \sum_{t=1}^T \boldsymbol{r}_t^\top \boldsymbol{x}_t + \boldsymbol{\lambda}^\top \big(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \boldsymbol{x}_t\big) \Big\} \\ &= \max_{\boldsymbol{x}_1,...,\boldsymbol{x}_T} \min_{\boldsymbol{\lambda}\ge\boldsymbol{0}} \sum_{t=1}^T L_t(\boldsymbol{x}_t, \boldsymbol{\lambda}) \end{aligned} \tag{7}$$

It can be seen that (7) is an online max-min problem involving a two-player zero-game. In each round, the function $L_t(\cdot)$ is accessible to the online learner before making a decision, which is different from the protocol of online saddle point problem (Rivera, Wang, and Xu 2018).

To further bound the online matching problem's regret, we involve variables to represent the maximum reward that can be gained by adding one unit of resource, which is also the upper bound of the dual price. Specifically, by defining $\Lambda_t = \prod_{k=1}^{K}[0, \lambda_{tk}^{\max}]$ where $\lambda_{tk}^{\max}$ is the maximum dual price for the k-th resource in the t-th round, the learner's cumulative reward for each sequence of $\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T$ is bounded by

$$R(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T) \geq \sum_{t=1}^{T} \left(\boldsymbol{r}_t^\top \hat{\boldsymbol{x}}_t + \min_{\boldsymbol{\lambda} \in \Lambda_t} \boldsymbol{\lambda}^\top \left(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t \right) \right) \tag{8}$$

To see this, consider a modified online matching problem where the resource constraints can be violated while the online learner must pay additional $\lambda_{tk}^{\max}$ for each unit of the k-th resource used over budget. Reminding that $\lambda_{tk}^{\max}$ is the maximum reward gained by adding one unit of resource, the total reward earned by the online learner in the modified problem is given by the right-hand side of (8), which can be no more than the cumulative reward in the original problem. Furthermore, let $\boldsymbol{\lambda}_t^* = \min_{\boldsymbol{\lambda} \in \Lambda_t} \boldsymbol{\lambda}^\top \left(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t \right)$, we have

$$\begin{aligned} R(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T) &\geq \sum_{t=1}^{T} \left(\boldsymbol{r}_t^\top \hat{\boldsymbol{x}}_t + \boldsymbol{\lambda}_t^{*\top} \left(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t \right) \right) \\ &= \sum_{t=1}^{T} L_t(\hat{\boldsymbol{x}}_t, \boldsymbol{\lambda}_t^*) \end{aligned} \tag{9}$$

Note that Λ_t is related to the revealed data, and $\boldsymbol{\lambda}_t^*$ can vary with Λ_t over time. When the environment changes faster, the variation of $\boldsymbol{\lambda}_t^*$ will be larger accordingly, and vice versa. Therefore, (Zinkevich 2003) introduced *path length*, defined as $P_T = \sum_{t=2}^{T} \|\boldsymbol{\lambda}_{t-1} - \boldsymbol{\lambda}_t\|_2$, to reflect the evolution of the dynamic environments. Furthermore, by substituting (7) and (9) into (3), the dynamic regret for the online matching w.r.t. $\boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_T^*$ is bounded by

$$\begin{aligned} &\text{Regret}(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T; \boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_T^*) \\ \leq& \max_{\boldsymbol{x}_1, ..., \boldsymbol{x}_T} \min_{\boldsymbol{\lambda} \geq \boldsymbol{0}} \sum_{t=1}^{T} L_t(\boldsymbol{x}_t, \boldsymbol{\lambda}) \ - \ \sum_{t=1}^{T} L_t(\hat{\boldsymbol{x}}_t, \boldsymbol{\lambda}_t^*) \\ =& \ \left(\max_{\boldsymbol{x}_1, ..., \boldsymbol{x}_T} \min_{\boldsymbol{\lambda} \geq \boldsymbol{0}} \sum_{t=1}^{T} L_t(\boldsymbol{x}_t, \boldsymbol{\lambda}) - \sum_{t=1}^{T} L_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{\lambda}}_t) \right) \\ &+ \left(\sum_{t=1}^{T} L_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{\lambda}}_t) - \sum_{t=1}^{T} L_t(\hat{\boldsymbol{x}}_t, \boldsymbol{\lambda}_t^*) \right) \end{aligned} \tag{10}$$

In (10), the dynamic regret for the online matching problem is bounded by the sum of two quantities (each in a bracket). In particular, the first term is related to the regret of online max-min problem, which can be eliminated by a primal-dual based algorithm, and the second term is equal to the dynamic regret of the OCO problem on dual prices. This decomposition is inspired by (Rivera, Wang, and Xu 2018), and the main difference is that we consider the dynamic regret in the dynamic environment in this paper.

Primal-Dual Online Algorithm

Motivated by the decomposition in (10), we propose a novel online approach, named Primal-Dual Online Algorithm (PDOA) to minimize both quantities. In particular, PDOA follows the primal-dual framework by optimizing the dual prices with OGD to eliminate the online max-min problem's regret. Moreover, inspired by (Zhang, Lu, and Zhou 2018), PDOA maintains a set of OGD experts, each tracking a potential path length of the dual price sequence, and combines them via an expert-tracking algorithm, which gives a sublinear dynamic regret bound for the second term. PDOA consists of two sub-algorithms, i.e., expert-algorithm and meta-algorithm, and the details of which are summarized in Algorithm 1 and Algorithm 2 respectively.

Expert-Algorithm The expert-algorithm is the OGD that regards the optimization of dual prices as an OCO problem, in which the loss function can be defined as

$$g_t(\boldsymbol{\lambda}) \triangleq L_t(\hat{\boldsymbol{x}}_t, \boldsymbol{\lambda}) = \boldsymbol{r}_t^\top \hat{\boldsymbol{x}}_t + \boldsymbol{\lambda}^\top \left(\boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t \right) \tag{11}$$

We run a series of instances of expert-algorithm concurrently. Each expert, which is called an *OGD expert*, takes a step size η as its input. As shown in Algorithm 1, the OGD expert submits its dual prices $\hat{\boldsymbol{\lambda}}_t^\eta$ to the meta-algorithm in Step 3, and receives the gradient $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ in Step 4. Then a gradient descent step is carried out to update the dual prices in Step 5:

$$\hat{\boldsymbol{\lambda}}_{t+1}^\eta = \max \left\{ 0, \hat{\boldsymbol{\lambda}}_t^\eta - \eta \nabla g_t(\hat{\boldsymbol{\lambda}}_t) \right\} \tag{12}$$

Note that instead of $\nabla g_t(\hat{\boldsymbol{\lambda}}_t^\eta)$, the OGD expert performs gradient descent with the same gradient $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ where $\hat{\boldsymbol{\lambda}}_t$ is the weighted average of dual prices and is calculated by the meta-algorithm. It is due to that we introduce a surrogate loss $l_t(\cdot)$ to replace $g_t(\cdot)$, the detail of which is presented in the meta-algorithm part.

Meta-Algorithm Learning from expert advice is a fundamental problem in the area of online decision making (Herbster and Warmuth 1998). The proposed meta-algorithm is built upon the Weighted Majority (Littlestone and Warmuth 1994; Hazan 2019), which is a weighted average learner adopting the exponential update scheme.

As shown in Algorithm 2, the meta-algorithm takes a set $\mathcal{H}$ of step size for OGD experts and its own step size β for the exponential update as inputs. In the initializing phase, we activate a set of OGD experts $\{E^\eta | \eta \in \mathcal{H}\}$ in Step 1 by invoking the expert-algorithm for each $\eta \in \mathcal{H}$, and set the initial weights for each OGD expert in Step 2. In this regard, we sort the step size in $\mathcal{H}$ and denote η_i as the i-th smallest one, then the weight for E^{η_i} is chosen as

$$w_1^{\eta_i} = \frac{|\mathcal{H}| + 1}{i(i+1)|\mathcal{H}|} \tag{13}$$

It is easy to verify that the sum of all weights is equal to 1, and the OGD expert with a larger step size is initialized with a smaller weight reasonably.

In each round, the t-th user's parameters $\boldsymbol{r}_t$ and $\boldsymbol{C}_t$ is revealed. The meta-algorithm receives a set of dual prices $\{\hat{\boldsymbol{\lambda}}_t^\eta | \eta \in \mathcal{H}\}$ from all OGD experts in Step 5, and calculates the weighted average of dual prices in Step 6:

$$\hat{\boldsymbol{\lambda}}_t = \sum_{\eta \in \mathcal{H}} w_t^\eta \hat{\boldsymbol{\lambda}}_t^\eta \tag{14}$$

Algorithm 1 PDOA: Expert-Algorithm

Require: T: total number of arrival users; η: step size

1: $\boldsymbol{\lambda}_0^\eta = \mathbf{0}$
2: **for** $t = 1$ **to** T **do**
3: Submit $\hat{\boldsymbol{\lambda}}_t^\eta$ to the meta-algorithm
4: Receive $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ from the meta-algorithm
5: Update dual prices

$$\hat{\boldsymbol{\lambda}}_{t+1}^\eta = \max\left\{0, \hat{\boldsymbol{\lambda}}_t^\eta - \eta \nabla g_t(\hat{\boldsymbol{\lambda}}_t)\right\}$$

6: **end for**

Algorithm 2 PDOA: Meta-Algorithm

Require: T: total number of arrival users; $\boldsymbol{B}$: resource budgets; β: step size; $\mathcal{H}$: set of step sizes for OGD experts

1: Active a set of OGD experts $\{E^\eta | \eta \in \mathcal{H}\}$ by invoking the expert-algorithm for each $\eta \in \mathcal{H}$
2: Sort the step size in $\mathcal{H}$ in the ascending order $\eta_1 \leq \eta_2 \leq ... \leq \eta_N$, and set $w_1^{\eta_i} = \frac{|\mathcal{H}|+1}{i(i+1)|\mathcal{H}|}$
3: **for** $t = 1$ **to** T **do**
4: Receive $\boldsymbol{r}_t$ and $\boldsymbol{C}_t$
5: Receive $\hat{\boldsymbol{\lambda}}_t^\eta$ from each E^η
6: Calculate the weighted average

$$\hat{\boldsymbol{\lambda}}_t = \sum_{\eta \in \mathcal{H}} w_t^\eta \hat{\boldsymbol{\lambda}}_t^\eta$$

7: Make decision $\hat{\boldsymbol{x}}_t$ where

$$\hat{x}_{ti} = \begin{cases} 1, \ i = \arg\max\{r_{ti} - \sum_{k=1}^K c_{tik}\hat{\lambda}_{tk}\} \\ 0, \ \text{otherwise.} \end{cases}$$

8: Calculate the gradient of $g_t(\cdot)$ at $\hat{\boldsymbol{\lambda}}_t$

$$\nabla g_t(\hat{\boldsymbol{\lambda}}_t) = \boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t$$

9: Construct the surrogate loss

$$l_t(\boldsymbol{\lambda}) \triangleq \nabla g_t(\hat{\boldsymbol{\lambda}}_t)^\top (\boldsymbol{\lambda} - \hat{\boldsymbol{\lambda}}_t)$$

10: Update the weight of each E^η

$$w_{t+1}^\eta = \frac{w_t^\eta \exp(-\beta l_t(\hat{\boldsymbol{\lambda}}_t^\eta))}{\sum_{\mu \in \mathcal{H}} w_t^\mu \exp(-\beta l_t(\hat{\boldsymbol{\lambda}}_t^\mu))}$$

11: Send the gradient $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ to each E^η
12: **end for**

Ensure: $\{\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T\}$: online assignment

where w_t^η is the weight for E^η. Therefore, by adopting the primal-dual approach, we derive the decision $\hat{\boldsymbol{x}}_t$ with the dual prices $\hat{\boldsymbol{\lambda}}_t$ in Step 7:

$$\hat{x}_{ti} = \begin{cases} 1, \ i = \arg\max\{r_{ti} - \sum_{k=1}^K c_{tik}\hat{\lambda}_{tk}\} \\ 0, \ \text{otherwise.} \end{cases} \tag{15}$$

where $\hat{x}_{ti}$ is the i-th component of $\hat{\boldsymbol{x}}_t$ and $\hat{\lambda}_{tk}$ is the k-th component of $\hat{\boldsymbol{\lambda}}_t$. It is important to point out that (15) gives an optimal $\boldsymbol{x}_t$, i.e., $L_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{\lambda}}_t) = \max L_t(\boldsymbol{x}_t, \hat{\boldsymbol{\lambda}}_t)$, which makes convenience for our regret analysis.

In the next step, we expect to calculate the value and the gradient of $g_t(\cdot)$ at $\boldsymbol{\lambda}_t^\eta$ for each E^η, which leads to an expensive calculation cost. To reduce the cost, inspired by (Zhang, Lu, and Zhou 2018) we define a surrogate loss $l_t(\cdot)$ as

$$l_t(\boldsymbol{\lambda}) \triangleq \nabla g_t(\hat{\boldsymbol{\lambda}}_t)^\top (\boldsymbol{\lambda} - \hat{\boldsymbol{\lambda}}_t) \tag{16}$$

Proposition 1. *The minimizer of the regret w.r.t. $l_t(\cdot)$ can be used to minimize that w.r.t. $g_t(\cdot)$.*

Proof. Proposition 1 can be proved with the first-order condition of convexity (Boyd and Vandenberghe 2004). Since $g_t(\cdot)$ is convex, we have

$$g_t(\boldsymbol{\lambda}) \geq g_t(\hat{\boldsymbol{\lambda}}_t) + \nabla g_t(\hat{\boldsymbol{\lambda}}_t)^\top (\boldsymbol{\lambda} - \hat{\boldsymbol{\lambda}}_t), \ \ \forall \boldsymbol{\lambda}$$

Then, with the definition (16), we can see that the regret w.r.t. $g_t(\cdot)$ is bounded by that w.r.t. $l_t(\cdot)$, i.e.,

$$g_t(\hat{\boldsymbol{\lambda}}_t) - g_t(\boldsymbol{\lambda}) \leq -l_t(\boldsymbol{\lambda}) = l_t(\hat{\boldsymbol{\lambda}}_t) - l_t(\boldsymbol{\lambda})$$

Therefore, the minimizer of the latter can be used to minimize the former. □

Since $l_t(\cdot)$ is linear, for any $\eta \in \mathcal{H}$ we have $\nabla l_t(\hat{\boldsymbol{\lambda}}_t^\eta) = \nabla g_t(\hat{\boldsymbol{\lambda}}_t)$. As a consequence, when we utilize the surrogate loss $l_t(\cdot)$, the meta-algorithm only need to calculate and send the same gradient $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ in Step 8:

$$\nabla g_t(\hat{\boldsymbol{\lambda}}_t) = \boldsymbol{B}/T - \boldsymbol{C}_t^\top \hat{\boldsymbol{x}}_t \tag{17}$$

and construct the surrogate loss $l_t(\cdot)$ as (16) in Step 9. The weight of each OGD expert is updated based on $l_t(\hat{\boldsymbol{\lambda}}_t^\eta)$ following the exponential update scheme in Step 10:

$$w_{t+1}^\eta = \frac{w_t^\eta \exp(-\beta l_t(\hat{\boldsymbol{\lambda}}_t^\eta))}{\sum_{\mu \in \mathcal{H}} w_t^\mu \exp(-\beta l_t(\hat{\boldsymbol{\lambda}}_t^\mu))} \tag{18}$$

In Step 11, the gradient $\nabla g_t(\hat{\boldsymbol{\lambda}}_t)$ is sent to all experts for their own updates.

Remark 1 In order to minimize the upper bound (10) of the dynamic regret, we construct $\mathcal{H}$ as

$$\mathcal{H} = \left\{ \eta_i = \frac{\overline{\lambda}\, 2^{i-1}}{\max\{\overline{b}, \overline{c} - \underline{b}\}} \sqrt{\frac{7}{2T}} \ \middle| \ i = 1, , , .N \right\} \tag{19}$$

where $N = \lceil \frac{1}{2}\log_2(1 + 4T/7) \rceil$, and have the following theorem.

Theorem 1. *Set $\frac{1}{\beta} = \left(\overline{r} + \overline{\lambda}(\overline{b} - \underline{b} + \overline{c})K\right)\sqrt{T/8}$ in Algorithm 2, then for each dual price sequence $\boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_T^*$ defined in* (9)*, the proposed PDOA's dynamic regret satisfies*

$$\begin{aligned} &\textit{Regret}(\hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_T; \boldsymbol{\lambda}_1^*, ..., \boldsymbol{\lambda}_T^*) \\ &\leq \frac{3}{4}\max\left\{\overline{b}, \overline{c} - \underline{b}\right\}\sqrt{2KT(7\overline{\lambda}^2 K + 4\overline{\lambda}\sqrt{K}P_T)} \\ &\quad + \frac{\left(\overline{r} + \overline{\lambda}(\overline{b} - \underline{b} + \overline{c})K\right)\sqrt{2T}}{4}\Big(1 + 2\ln(k+1)\Big) \\ &= O(K\sqrt{T(1+P_T)}) \end{aligned} \tag{20}$$

where

$$k = \left\lceil \frac{1}{2}\log_2\left(1 + \frac{4P_T}{7\overline{\lambda}\sqrt{K}}\right)\right\rceil$$

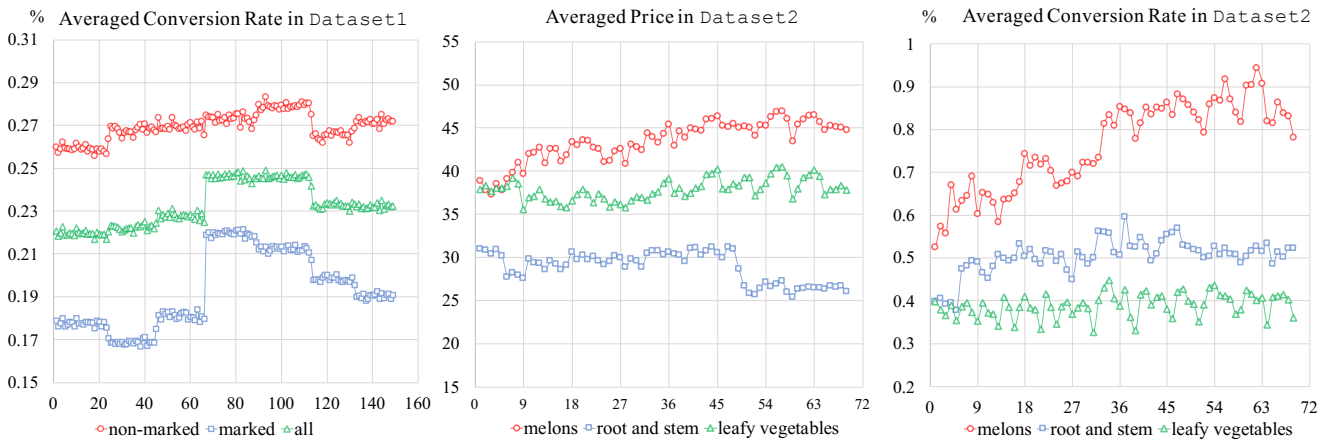

Figure 1: Statistics about user queries. Each point shows one of the statistics which is averaged over queries from the same bin.

The complete proof of Theorem 1 can be found in the appendix. For the OCO problem in dynamic environments, there exists a lower bound of the dynamic regret, i.e., $\Omega(\sqrt{T(1+P_T)})$, which is established by (Zhang, Lu, and Zhou 2018). Intuitively, the online matching problem is not easier than the OCO problem due to its additional consideration of budget constraints, thus we can extend $\Omega(\sqrt{T(1+P_T)})$ as a lower bound for the online matching problem in dynamic environments. Based on this discussion, the dynamic regret of our proposed method, i.e., $O(K\sqrt{T(1+P_T)})$, matches the lower bound for the problem. Moreover, Theorem 1 requires the advanced knowledge of T. For the tasks where T can not be estimated in advance, the doubling trick may be a helpful technique that has been studied in the context of bandits or online convex optimization (Cesa-Bianchi and Lugosi 2006; Erven et al. 2011).

Experiments

Settings

We evaluate the effectiveness of the proposed PDOA on two real-world applications.

`Problem1` is from one of the largest online retail platforms over Asian and Europe. In international trades, the timeliness of delivery is an important index that impacts users' satisfaction. The staff mark the commodities that had been stored in the local warehouse. To reduce the delivery time, algorithms are required to guarantee a certain number of purchases for each marked commodity (called a *task* here) when maximizing the total purchase amount of the platform. Formally, denote cr_{it} as the conversion rate from exposing the i-th commodity to the t-th user and g_i as the guaranteed purchase amount for i-th commodity, thus the problem can be formulated as

$$\begin{aligned} \max_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T} \ & \sum_{t=1}^{T}\sum_{i=1}^{M} cr_{it}x_{it} \\ \text{s.t.} \ & \sum_{t=1}^{T} -cr_i x_{it} \le -g_i, \ \forall i \in S \end{aligned} \tag{21}$$

where S is the set for marked commodities.

`Problem2` is from one of the largest e-commerce grocery retailers. Among all commodities supplied on the mobile application, there is plenty of fresh food such as vegetables, meat, and eggs, required to be sold out on the day. The remaining inventory will be discarded at the mid-night. Therefore, algorithms in this scenario aim at maximizing the Gross Merchandise Volume (GMV) and minimizing the inventory losses simultaneously. Formally, denote cr_{it} as the conversion rate from exposing the i-th commodity to the t-th user and c_i as the cost of the i-th commodity, p_i as the profile of selling the i-th commodity and b_i as the inventory of the i-th commodity, the problem can be formulated as

$$\begin{aligned} \max_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_T} \ & \sum_{t=1}^{T}\sum_{i=1}^{M} cr_{it}(p_i + \gamma c_i \mathbb{I}_i)x_{it} \\ \text{s.t.} \ & \sum_{t=1}^{T} cr_i x_{it} \le b_i, \ \forall i \in [M] \end{aligned} \tag{22}$$

where $\mathbb{I}_i$ indicates whether the i-th commodity is an fresh food that need to be sold out, and γ is the trade-off parameter between GMV and inventory losses.

It is easy to verified that both (21) and (22) are special cases of the online matching problem (2), which are suitable for our experiments.

Datasets In this paper, `Dataset1` and `Dataset2` are constructed for `Problem1` and `Problem2`, respectively. Each dataset consists of millions of queries and 600 commodities, all of which are sampled from the real online data. In `Dataset1`, there are 300 marked commodities stored in the local warehouse, and the target of purchase number for each marked commodity is appointed by staff. In `Dataset2`, there are 300 fresh foods to be sold out, and the inventories of the day are provided by the supply system. We collect the data from 5 days, each forming a matching problem, thus there are 5 problems to be evaluated for each application. Due to the limitation of commercial secrets, the detailed information about the datasets is masked.

	Greedy	Balance	OLA	OGD	PDOA	Greedy	Balance	OLA	OGD	PDOA
DAY1	67.0%	NaN	79.3%	91.0%	**91.3%**	181983.0	NaN	181696.0	181440.0	181434.0
DAY2	45.7%	NaN	55.0%	73.7%	**89.7%**	163176.0	NaN	162585.0	162079.0	161477.0
DAY3	46.0%	NaN	66.0%	79.3%	**93.0%**	168679.0	NaN	167312.0	167471.0	167084.0
DAY4	55.7%	NaN	63.7%	81.7%	**92.7%**	176353.0	NaN	175747.0	175464.0	174100.0
DAY5	60.3%	NaN	69.0%	86.3%	**93.7%**	182635.0	NaN	182089.0	181932.0	181546.0
AVG.	54.9%	NaN	66.6%	82.4%	**92.1%**	174565.2	NaN	173885.8	173677.2	173128.2
	`Dataset1`: Completed Task Proportion					`Dataset1`: Purchase Number				
DAY1	24917.0	24284.0	23054.0	22811.0	**21982.0**	112671.0	134299.0	112664.0	120953.0	**138213.0**
DAY2	24987.0	24194.0	22890.0	22826.0	**22153.0**	119319.0	127309.0	119444.0	122418.0	**131227.0**
DAY3	23958.0	23649.0	21542.0	21526.0	**21069.0**	122070.0	129068.0	122231.0	123086.0	**130704.0**
DAY4	22518.0	21833.0	20121.0	19981.0	**18846.0**	122125.0	127672.0	121809.0	122052.0	**129677.0**
DAY5	22615.0	22227.0	20594.0	20472.0	**19993.0**	104048.0	116177.0	103172.0	104418.0	**119310.0**
AVG.	23799.0	23237.4	21640.2	21523.2	**20808.6**	116046.6	126905.0	115864.0	118585.4	**129826.2**
	`Dataset2`: Inventory Loss					`Dataset2`: GMV				

Table 1: The results of PDOA and its compared methods over 5 days. Boldface highlights the significantly best performance.

In a dynamic environment, the distribution of arrival users is changing over time. To verify that, we sort the queries of all 5 days according to their arrival time and split them into several bins with equal length, then several statistics about user queries can be calculated for each bin. Note that every bin has 20,000 queries for both datasets. Specifically, the averaged conversion rates of the marked commodities, non-marked commodities, and all commodities are analyzed for `Dataset1`. The averaged conversion rates and the averaged price of each category are performed for `Dataset2`. The results are shown in Figure 1. It can be observed that the distributions of queries are changing all the time, which follows our expectations because the features of arrival users are not stationary.

Methods We compare the following approaches:

- Greedy that maximizes the rewards without the consideration of constraints.
- Balance that maximizes the rewards considering the constraints (Kalyanasundaram and Pruhs 2000).
- OLA (One-time Learning Algorithm) where dual prices are learned by solving a fractional matching problem on the first $1/4$ queries (Agrawal, Wang, and Ye 2014).
- OGD with step size $\eta = 0.01$ (Li, Sun, and Ye 2020).
- PDOA where the step size β in Algorithm 2 is set as 0.1 and the set $\mathcal{H}$ of step sizes is performed as (19).

Moreover, since the values of the budgets in (21) are all negative, Balance is not applicable for `Dataset1`.

Results

We adopt the *Completed Task Proportion* and the *Purchase Number* as evaluation measurements for `Dataset1`, and adopt the *Inventory Loss* and the *GMV* as evaluation measurements for `Dataset2`. The results of all methods over 5 days are shown in Table 1.

`Dataset1`. For the Completed Task Proportion, PDOA achieves the highest results on every single day, and OGD is better than Greedy and OLA, since the latter two are proposed for stationary environments. The results of OGD lack stability, ranging from 73.7% to 91.0%, because the optimal step sizes of OGD vary widely for different dynamic environments. By contrast, PDOA completes 11.7% more tasks than OGD over 5 days, which is more stable in practice. Note that Greedy has the best Purchase Number, merely because of its ignorance of assigning tasks with an averaged Completed Task Proportion of 54.9%, which is unacceptable in the real application. Actually, the Purchase Numbers of all compared methods are comparable according to the paired t-test at significance level 95%, thus the Completed Task Proportion becomes the most discriminative measurement for the dataset.

`Dataset2`. The results of PDOA show great superiority to the compared methods, namely, PDOA achieves the least Inventory Loss and the highest GMV on every single day. The Inventory Losses of three primal-dual based methods are significantly less than Greedy and Balance. Since the dual prices are fixed in the most assignments, OLA can not adapt to the dynamic environment. On the contrary, OGD benefits from the online update of dual prices and thus performs better than OGD. Moreover, the value of step size η is vital to the OGD's performance, but we can not obtain the optimal one in advance. By maintaining a set of OGD experts and combining them via an expert-tracking algorithm, PDOA is able to get rid of this difficulty. In particular, PDOA increases 9.5% total GMV and reduces 3.3% Inventory Loss than OGD over 5 days.

Conclusion

We study the online matching problem in dynamic environments where the dual optimal prices are allowed to vary over time. We bound the dynamic regret of online matching problem by the sum of two quantities, including a regret of online max-min problem and a dynamic regret of OCO problem. We proposed a novel approach named PDOA to minimize both quantities, achieving an $O(K\sqrt{T(1+P_T)})$ dynamic regret. In particular, PDOA adopts the primal-dual framework by optimizing the dual prices with OGD to eliminate the online max-min problem's regret. Moreover, it maintains a set of OGD experts and combines them via an expert-tracking algorithm, which gives a sublinear dynamic regret bound for the OCO problem. The experiments on real-world applications exhibit the superiority of our approach.

Acknowledgements

The authors want to thank reviewers for the helpful comments and thank Peng Zhao and Dr. Yao-Xiang Ding for their helpful discussions.

References

Aggarwal, G.; Goel, G.; Karande, C.; and Mehta, A. 2011. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, 1253–1264.

Agrawal, S.; and Devanur, N. R. 2014. Fast algorithms for online stochastic convex programming. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1405–1424.

Agrawal, S.; R, N. D.; and Li, L. 2016. An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives. In *Proceedings of the 29th Annual Conference on Learning Theory*, 4–18.

Agrawal, S.; Wang, Z.; and Ye, Y. 2014. A dynamic near-optimal algorithm for online linear programming. *Operations Research* 62(4): 876–890.

Besbes, O.; Gur, Y.; and Zeevi, A. 2015. Non-stationary stochastic optimization. *Operations Research* 63(5): 1227–1244.

Boyd, S.; and Vandenberghe, L. 2004. *Convex optimization*. Cambridge University Press.

Buchbinder, N.; and Naor, J. 2009. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science* 3(2-3): 93–263.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge University Press.

Erven, T.; Koolen, W. M.; Rooij, S.; and Grünwald, P. 2011. Adaptive hedge. *Advances in Neural Information Processing Systems* 24: 1656–1664.

Esfandiari, H.; Korula, N.; and Mirrokni, V. S. 2015. Online allocation with traffic spikes: Mixing adversarial and stochastic models. In *Proceedings of the 16th ACM Conference on Economics and Computation*, 169–186.

Fahrbach, M.; Huang, Z.; Tao, R.; and Zadimoghaddam, M. 2020. Edge-weighted online bipartite matching. *arXiv preprint arXiv:2005.01929* .

Feldman, J.; Henzinger, M.; Korula, N.; Mirrokni, V. S.; and Stein, C. 2010. Online stochastic packing applied to display ad allocation. In *Proceedings of the 18th Annual European Symposium*, 182–194.

Hazan, E. 2019. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207* .

Hazan, E.; Agarwal, A.; and Kale, S. 2007. Logarithmic regret algorithms for online convex optimization. *Machine Learning* 69(2-3): 169–192.

Herbster, M.; and Warmuth, M. K. 1998. Tracking the best expert. *Machine Learning* 32(2): 151–178.

Huzhang, G.; Huang, X.; Zhang, S.; and Bei, X. 2017. Online roommate allocation problem. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 235–241.

Kalyanasundaram, B.; and Pruhs, K. 2000. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science* 233(1-2): 319–325.

Karp, R. M.; Vazirani, U. V.; and Vazirani, V. V. 1990. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, 352–358.

Kesselheim, T.; Radke, K.; Tönnis, A.; and Vöcking, B. 2013. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *European Symposium on Algorithms*, 589–600. Springer.

Li, X.; Sun, C.; and Ye, Y. 2020. Simple and fast algorithm for binary integer and online linear programming. *arXiv preprint arXiv:2003.02513* .

Littlestone, N.; and Warmuth, M. K. 1994. The weighted majority algorithm. *Information and Computation* 108(2): 212–261.

Mehta, A. 2013. Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science* 8(4): 265–368.

Rivera, A.; Wang, H.; and Xu, H. 2018. The online saddle point problem and online convex optimization with knapsacks. *arXiv preprint arXiv:1806.08301* .

Shalev-Shwartz, S.; Singer, Y.; Srebro, N.; and Cotter, A. 2011. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical Programming* 127(1): 3–30.

Zhang, L.; Lu, S.; and Zhou, Z.-H. 2018. Adaptive online learning in dynamic environments. In *Advances in Neural Information Processing Systems*, 1323–1333.

Zhao, P.; Wang, G.; Zhang, L.; and Zhou, Z.-H. 2020a. Bandit convex optimization in non-stationary environments. In *International Conference on Artificial Intelligence and Statistics*, 1508–1518.

Zhao, P.; Zhang, Y.-J.; Zhang, L.; and Zhou, Z.-H. 2020b. Dynamic regret of convex and smooth functions. *Advances in Neural Information Processing Systems* 33.

Zhou, Y.-H.; Liang, C.; Li, N.; Yang, C.; Zhu, S.; and Jin, R. 2019. Robust online matching with user arrival distribution drift. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 459–466.

Zinkevich, M. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, 928–936.