Stateful Posted Pricing with Vanishing Regret via Dynamic Deterministic Markov Decision Processes[*]

Yuval Emek [†] Ron Lavi [‡] Rad Niazadeh [§] Yangguang Shi [¶]

Abstract

An online problem called *dynamic resource allocation with capacity constraints (DRACC)* is introduced and studied in the realm of posted price mechanisms. This problem subsumes several applications of stateful pricing, including but not limited to posted prices for online job scheduling and matching over a dynamic bipartite graph. Since existing online learning techniques do not yield vanishing-regret for this problem, we develop a novel online learning framework over deterministic Markov decision processes with *dynamic* state transition and reward functions. Following that, we prove, based on a reduction to the well-studied problem of online learning with switching costs, that if the Markov decision process admits a *chasing oracle* — i.e., an oracle that simulates any given policy from any initial state with bounded loss — then the online learning problem can be solved with vanishing regret. Our results for the DRACC problem and its applications are then obtained by devising (randomized and deterministic) chasing oracles that exploit the particular structure of these problems.

1 Introduction

Price posting is a common selling mechanism across various corners of electronic commerce. Its applications span from more traditional domains such as selling flight tickets on Delta's website or selling products on Amazon, to more emerging domains such as selling cloud services on Amazon Web Services (AWS) or pricing ride-shares in Uber. The prevalence of price posting comes from its several important advantages: it is incentive compatible (at least in a one-shot setting or against myopic users), simple to grasp, and can easily fit in an online (or dynamic) environment where buyers arrive sequentially over time. Therefore, online posted pricing mechanisms, also known as dynamic pricing, have been studied quite extensively in computer science, operations research, and economics (for a comprehensive survey, see den Boer [41]).

Among the successful approaches to devise dynamic pricing algorithms, one approach — that is particularly effective when the environment is changing in a non-stationary fashion and price experimentation can be done — is that of non-parametric dynamic pricing. The queen of killer methods for devising online non-parametric posted prices is via *adversarial online learning*, where the goal is to design online learning algorithms in an adversarial environment with vanishing regret (Bubeck et al. [27, 26], Leme and Schneider [65], Feldman et al. [50], Blum and Hartline [22], Blum et al. [23], Kleinberg and Leighton [61]). Here, a sequence of buyers arrive, each associated with her own valuation function that is assumed to be devised by a malicious adversary, and the goal is to post a sequence of price vectors that performs almost as good as the best fixed pricing policy in hindsight. Despite its success, a technical limitation of this method (shared by the aforementioned papers) forces the often less natural assumption of unlimited item supply to ensure that the selling platform is *stateless*. However, in many applications of online posted pricing, the platform is *stateful*; indeed, prices can depend on previous sales that determine the platform's state. Examples for such stateful platforms include

[*]Accepted for publication at Mathematics of Operations Research

[†]Faculty of Data and Decision Sciences, Technion - Israel Institute of Technology, Haifa, Israel, yemek@technion.ac.il

[‡]University of Bath, UK, and Technion - Israel Institute of Technology, Israel, ron.lavi.ac@gmail.com

[§]University of Chicago Booth School of Business, Chicago, IL, United States, rad.niazadeh@chicagobooth.edu

[¶]School of Computer Science and Technology, Shandong University, Qingdao, Shandong, China, shiyangguang@sdu.edu.cn

selling resources of limited supply, in which the state encodes the number of remaining inventories of different products, and selling resources in cloud computing to schedule online jobs, in which the state encodes the currently scheduled jobs.

The above mentioned limitation is in sharp contrast to the posted prices literature that considers (the more parametric) stochastic settings where the buyers' valuations or demands are drawn independently and identically from unknown distributions (Besbes and Zeevi [20], Araman and Caldentey [10], Broder and Rusmevichientong [25], Babaioff et al. [17], Badanidiyuru et al. [19], Zhang et al. [77], Agrawal and Devanur [5]), or independently from known distributions (Chawla et al. [33], Harrison et al. [55], Feldman et al. [49], Chawla et al. [31]). By exploiting the randomness (and distributional knowledge) of the input and employing other algorithmic techniques, these papers cope with limited supply and occasionally, with more complicated stateful pricing scenarios. However, the stochastic approach does not encompass the (realistic) scenarios in which the buyers' valuations are correlated in various complex ways, scenarios that are typically handled using adversarial models. The only exception that we are aware of in this regard is the work of Chawla et al. [32] that takes a different approach: they consider the online job scheduling problem, and given access to a collection of (truthful) posted price scheduling mechanisms, show how to design a (truthful) online scheduling mechanism in an adversarial environment whose regret with respect to the given mechanism collection is vanishing.

Motivated by the abundance of stateful posted pricing platforms and the lack of an appropriate algorithmic theory for such settings as discussed above, and inspired by Chawla et al. [32], we study the design of adversarial online learning algorithms with vanishing regret for a rather general online resource allocation framework. In this framework, termed *dynamic resource allocation with capacity constraints (DRACC)*, dynamic resources of limited inventories arrive and depart over time, and an online mechanism sequentially posts price vectors to (myopically) strategic buyers with adversarially chosen combinatorial valuations (refer to Section 2 for the formal model). The goal is to post a sequence of price vectors with the objective of maximizing revenue, while respecting the inventory restrictions of dynamic resources for the periods of time in which they are active. We consider a full-information setting, in which the buyers' valuations are elicited by the platform after posting prices in each round of the online execution.

Given a collection of pricing policies for the DRACC framework, we aim to construct a sequence of price vectors that is guaranteed to admit a vanishing regret with respect to the best fixed pricing policy in hindsight. Interestingly, our abstract framework is general enough to admit, as special cases, two important applications of stateful posted pricing, namely, online job-scheduling and matching over a dynamic bipartite graph; these applications, for which existing online learning techniques fail to obtain vanishing regret, are discussed in detail in Section 4.

1.1 Our contributions and techniques

Our main result is a vanishing-regret posted price mechanism for the DRACC problem (refer to Section 3 for a formal exposition).

> *For any DRACC instance with T users and for any collection Γ of pricing policies, the regret of our proposed posted price mechanism (in terms of expected revenue) with respect to the in-hindsight best policy in Γ is sublinear in T.*

We prove this result by abstracting away the details of the pricing problem and considering a more general stateful decision making problem. To this end, we introduce a new framework, termed *dynamic deterministic Markov decision process (Dd-MDP)*, which generalizes the classic *deterministic MDP* problem to an adversarial online learning dynamic setting. In this framework, a decision maker picks a feasible action for the current state of the MDP, not knowing the state transitions and the rewards associated with each transition; the state transition function and rewards are then revealed. The goal of the decision maker is to pick a sequence of actions with the objective of maximizing her total reward. In particular, we look at vanishing-regret online learning, where the decision maker is aiming at minimizing her regret, defined with respect to the in-hindsight best fixed policy (i.e., a mapping from states to actions) among the policies in a given collection Γ.

Not surprisingly, vanishing-regret online learning is impossible for this general problem (see Proposition 1). To circumvent this difficulty, we introduce a structural condition on Dd-MDPs that enables online learning with vanishing regret. This structural condition ensures the existence of an ongoing *chasing oracle* that allows one to simulate a given fixed policy from any initial state, irrespective of the actual current state, while ensuring a small (vanishing) *chasing regret*. The crux of our technical contribution is cast in proving that the Dd-MDPs induced by DRACC instances satisfy this *chasability* condition.

Subject to the chasability condition, we establish a reduction from designing vanishing-regret online algorithms for Dd-MDP to the extensively studied (classic stateless) setting of *online learning with switching cost* (Kalai and Vempala [57]). At high level, we have one arm for each policy in the given collection Γ and employ the switching cost online algorithm to determine the next policy to pick. Each time this algorithm suggests a switch to a new policy $\gamma \in \Gamma$, we invoke the chasing oracle that attempts to simulate γ, starting from the current state of the algorithm which may differ from γ's current state. In summary, we obtain the following result (see Theorem 4 for a formal exposition).

> *For any T-round Dd-MDP instance that satisfies the chasability condition and for any collection Γ of policies, the regret of our online learning algorithm with respect to the in-hindsight best policy in Γ is sublinear (and optimal) in T.*

We further study the bandit version of the above problem, where the state transition function is revealed at the end of each round, but the learner only observes the current realized reward instead of the complete reward function. By adapting the chasability condition to this setting, we obtain near optimal regret bounds. See Theorem 11 and Corollary 5 in Section 6 for a formal statement.

Our abstract frameworks, both for stateful decision making and stateful pricing, are rather general and we believe that they will turn out to capture many natural problems as special cases (on top of the applications discussed in Section 4).

1.2 Additional related work and discussion

In the DRACC problem, the class of feasible prices at each time t is determined by the remaining inventories, which in turn depend on the prices picked at previous times $t' < t$. This kind of dependency cannot be handled by the conventional online learning algorithms, such as follow-the-perturbed-leader (Kalai and Vempala [57]) and EXP3 (Auer et al. [14]). That is why we aim for the stateful model of online learning, which allows a certain degree of dependence on the past actions.

Several attempts have been made to formalize and study stateful online learning models. The authors of Arora et al. [11], Feldman et al. [50] consider an online learning framework where the reward (or cost) at each time depends on the k recent actions for some fixed $k > 0$. This framework can be viewed as a reward function that depends on the system's state that, in this case, encodes the last $k - 1$ actions.

A reward function that depends on the k recent actions is also considered in Feige et al. [47], where online learning algorithms are developed to compete against stateful policies. The states considered in Feige et al. [47] are owned by the policies, rather than specified by the environment. Their algorithms and the contending policies are allowed to make switches between arbitrary states with no costs. This setting is very different from the multi-state model in the current paper, where the state transition functions are specified by the (adversarial) environment.

There is an extensive line of work on online learning models that address general multi-state systems, typically formalized by means of stochastic (Even-Dar et al. [46], Guan et al. [53], Yu et al. [76], Abbasi-Yadkori et al. [1], Neu et al. [70]) or deterministic (Dekel and Hazan [40]) MDPs. The disadvantage of these models from our perspective is that they all have at least one of the following two restrictions: (a) all actions are always feasible regardless of the current state (see Abbasi-Yadkori et al. [1], Even-Dar et al. [46], Guan et al. [53], Yu et al. [76]); or (b) the state transition function is fixed (static) and known in advance (see Dekel and Hazan [40], Even-Dar et al. [46], Guan et al. [53], Neu et al. [70], Yu et al. [76]).

In the DRACC problem, however, not all actions (price vectors) are feasible for every state and the state transition function at time t is revealed only after the decision maker has committed to its action. Moreover, the aforementioned MDP-based models require a certain type of state *connectivity* in the sense that the Markov chain induced by each action should be irreducible (Abbasi-Yadkori et al. [1], Even-Dar et al. [46], Guan et al. [53], Neu et al. [70], Yu et al. [76]) or at least the union of all induced Markov chains should form a strongly connected graph (Dekel and Hazan [40]). In contrast, in the DRACC problem, depending on the inventories of the resources, it may be the case that certain inventory vectors can never be reached (regardless of the decision maker's actions).

On the algorithmic side, a common feature of all aforementioned online learning models is that for every instance, there exists some $k > 0$ that can be computed in a preprocessing stage (and does not depend on T) such that the online learning can "catch" the state (or distribution over states) of any given sequence of actions in *exactly* k time units. While this feature serves as a corner stone for the existing online learning algorithms, it is not present in our model, hence our online learning algorithm has to employ different ideas.

In terms of applications, our work resembles some aspects of the literature on online two-sided allocation problems (and online convex programming) with stochastic input. In Devanur et al. [42], Kesselheim et al. [59], Agrawal and Devanur [4], a family of online resource allocation problems is investigated under a different setting from ours. The resources in their problem models are static, which means that every resource is revealed at the beginning, and remains active from the first user to the last one. Different from our adversarial model, these papers take different stochastic settings on the users, such as the random permutation setting where a fixed set of users arrive in a random order (Kesselheim et al. [59], Agrawal and Devanur [4]), and the random generation setting where the parameters of each user are drawn i.i.d. from some unknown distribution (Devanur et al. [42], Agrawal and Devanur [4]). In these papers, the assignment of the resources to the requests are fully determined by a single decision maker, and the decision for each request depends on the revealed parameters of the current request and previous ones. By contrast, we study the scenario where each strategic user makes her own decision of choosing the resources, and the price posted to each user should be specified independently of the valuation of the current user.

Our allocation problem resembles some aspect of the adversarial multi-armed bandit problems with knapsacks. For example, in Immorlica et al. [56], the online resource allocation problem with static resources is studied under the adversarial setting, where the rewards and the consumption of resources incurred by the chosen prices are specified by the adversary. The algorithms in Immorlica et al. [56] are designed to compete with the best fixed distribution over the prices, while the benchmarks used in our paper are pricing policies that can specify different price vectors over the active resources depending on the time and inventory vectors. The main results in Immorlica et al. [56] are presented in the form of competitive ratios, rather than regrets.

Static resources with limited supply are also investigated in Brantley et al. [24] under the topic of *episodic reinforcement learning* (episodic RL) with a stochastic setting. In particular, the state transitions and the consumption of resources in Brantley et al. [24] are separated from each other and are decided by a static stochastic MDP. The episodic RL is studied under the adversarial setting in Lykouris et al. [68]. In comparison to our setting, the decision maker in episodic RL always moves back to the initial state when making a choice at the beginning of each episode. Under such a setting, the cumulative reward in a whole episode can be viewed as a stateless function (specified by the adversary) over the choices, and the decision maker can always easily "catch" the state of any fixed benchmark policy.

There is also a rich and recent literature on dynamic pricing and auctions for long-lived (i.e., non-myopic) buyers, e.g., Amin et al. [7, 8], Kanoria and Nazerzadeh [58], Mohri and Munoz [69], Agrawal et al. [3]. The setup and the techniques in these work diverge from us, mostly because our buyers are making a simpler one-shot decision (and hence the sellers is reacting accordingly).

Beyond online learning models and obtaining vanishing regret, there is a rich literature on dynamic pricing under known valuation distributions using techniques from *prophet inequality*, originated from the seminal work of Samuel-Cahn [73]. In the standard prophet inequality for selling one item, a static posted pricing obtains $\frac{1}{2}$ of the expected value of the maximum value, which can be easily translated into a sequential pricing algorithm that obtains $\frac{1}{2}$ of the revenue obtained by the optimal revenue mechanism. There are several work to extend this idea to various settings, e.g., Correa et al. [35, 38], Niazadeh et al. [71]. See also Correa et al. [36] for some of the recent developments. These developments include many variations such as prophet inequalities with limited samples form the distributions (Azar et al. [15]), i.i.d. and random order (Esfandiari et al. [45], Abolhassani et al. [2]), and finally ordered prophets (a.k.a. the free-order sequential posted pricing problem, see Yan [74], Azar et al. [16], Beyhaghi et al. [21], Correa et al. [37]) have been explored, and discovering connections to the price of anarchy (Düetting et al. [43]), online contention resolution schemes (Feldman et al. [50], Lee and Singla [62]), and online combinatorial optimization (Göbel et al. [52]) have been of particular interest in this literature. Generalizations of the simple prophet inequality problem to combinatorial settings have also been studied, where the examples are matroids (Kleinberg and Weinberg [60]), knapsack (Feldman et al. [50]), k-uniform matroids (for better bounds, see Hajiaghayi et al. [54], Alaei [6], Anari et al. [9]), or even general downward-closed (Rubinstein [72]). Finally, techniques and results in this literature had an immense impact on mechanism design (Chawla et al. [33], Cai et al. [28], Feldman et al. [48], Babaioff et al. [18], Cai et al. [29], Chawla and Miller [34], Dütting et al. [44], Correa et al. [35]). For a full list of recent and old related results, refer to Lucier [67]. Our work diverges from this literature in that we do not have distributional knowledge or assumption about future users, and instead of competitive ratio we consider vanishing regret with respect to the best fixed pricing policy.

2 Model and Definitions

The DRACC problem. Consider N dynamic *resources* and T strategic myopic *users* arriving sequentially over *rounds* $t = 1, \ldots, T$, where round t lasts over the time interval $[t, t+1)$. Resource $i \in [N]$ arrives at the beginning of round $t_a(i)$ and departs at the end of round $t_e(i)$, where $1 \leq t_a(i) \leq t_e(i) \leq T$; upon arrival, it includes $c(i) \in \mathbb{Z}_{>0}$ units. We say that resource i is *active* at time t if $t_a(i) \leq t \leq t_e(i)$ and denote the set of resources active at time t by $A_t \subseteq [N]$. Let C and W be upper bounds on $\max_{i \in [N]} c(i)$ and $\max_{t \in [T]} |A_t|$, respectively.

The arriving user at time t has a *valuation function* $v_t : 2^{A_t} \to [0, 1)$ that determines her value $v_t(A)$ for each subset $A \subseteq A_t$ of resources active at time t. We assume that $v_t(\emptyset) = 0$ and that the users are quasi-linear, namely, if a subset A of resources is allocated to user t and she pays a total payment of q in return, then her utility is $v_t(A) - q$. A family of valuation functions that receives a special attention in this paper is that of k_t*-demand* valuation functions, where user t is associated with an integer parameter $1 \leq k_t \leq |A_t|$ and with a value $w_t^i \in [0, 1)$ for each active resource $i \in A_t$ so that her value for a subset $A \subseteq A_t$ is $\max_{A' \subseteq A : |A'| \leq k_t} \sum_{i \in A'} w_t^i$.

Stateful posted price mechanisms. We restrict our attention to dynamic posted price mechanisms that work based on the following protocol. In each round $t \in [T]$, the mechanism first observes which resources $i \in [N]$ arrive at the beginning of round t, together with their initial capacity $c(i)$, and which resources departed at the end of round $t-1$, thus updating its knowledge of A_t. It then posts a *price vector* $\boldsymbol{p}_t \in (0, 1]^{A_t}$ that determines the price $\boldsymbol{p}_t(i)$ of each resource $i \in A_t$ at time t. Following that, the mechanism elicits the valuation function v_t of the current user t and allocates (or in other words sells) one unit of each resource in the demand set $D_t^{\boldsymbol{p}_t}$ to user t at a total price of $\hat{q}_t^{\boldsymbol{p}_t}$, where

$$D_t^{\boldsymbol{p}} = \operatorname{argmax}_{A \subseteq A_t} \left\{ v_t(A) - \textstyle\sum_{i \in A} \boldsymbol{p}(i) \right\} \qquad \text{and} \qquad \hat{q}_t^{\boldsymbol{p}} = \textstyle\sum_{i \in D_t^{\boldsymbol{p}}} \boldsymbol{p}(i) \tag{1}$$

for any price vector $\boldsymbol{p} \in (0, 1]^{A_t}$, consistently breaking argmax ties according to the lexicographic order on A_t. A virtue of posted price mechanisms is that if the choice of $\boldsymbol{p}_t$ does not depend on v_t, then it is dominant strategy for (myopic) user t to report her valuation v_t truthfully.

Let $\boldsymbol{\lambda}_t \in \{0, 1, \ldots, C\}^{A_t}$ be the *inventory* vector that encodes the number $\boldsymbol{\lambda}_t(i)$ of units remaining from resource $i \in A_t$ at time $t = 1, \ldots, T$. Formally, if $t_a(i) = t$, then $\boldsymbol{\lambda}_t(i) = c(i)$; and if (a unit of) i is allocated to user t and i is still active at time $t+1$, then $\boldsymbol{\lambda}_{t+1}(i) = \boldsymbol{\lambda}_t(i) - 1$. We say that a price vector $\boldsymbol{p}$ is *feasible* for the inventory vector $\boldsymbol{\lambda}_t$ if $\boldsymbol{p}(i) = 1$ for every $i \in A_t$ such that $\boldsymbol{\lambda}_t(i) = 0$, that is, for every (active) resource

i exhausted by round t. To ensure that the resource inventory is not exceeded, we require that the posted price vector $\boldsymbol{p}_t$ is feasible for $\boldsymbol{\lambda}_t$ for every $1 \leq t \leq T$; indeed, since v_t is always strictly smaller than 1, this requirement ensures that the utility of user t from any resource subset $A \subseteq A_t$ that includes an exhausted resource is negative, thus preventing A from becoming the selected demand set (recall that the utility obtained by user t from the empty set is 0).

In this paper, we aim for posted price mechanisms whose objective is to maximize the extracted *revenue* defined to be the total expected payment $\mathbb{E}[\sum_{t=1}^{T} \hat{q}_t^{\boldsymbol{p}_t}]$ received from all users, where the expectation is over the mechanism's internal randomness.[1]

Adversarial online learning over pricing policies. To measure the quality of the aforementioned posted price mechanisms, we consider an adversarial online learning framework, where at each time $t \in [T]$, the decision maker picks the price vector $\boldsymbol{p}_t$ and an adaptive adversary simultaneously picks the valuation function v_t. The resource arrival times $t_a(i)$, departure times $t_e(i)$, and initial capacities $c(i)$ are also determined by the adversary. We consider the full information setting, where the valuation function v_t of user t is reported to the decision maker at the end of each round t. It is also assumed that the decision maker knows the parameters C and W upfront and that these parameters are independent of the instance length T.

A *pricing policy* γ is a function that maps each inventory vector $\boldsymbol{\lambda} \in \{0, 1, \ldots, C\}^{A_t}$, $t \in [T]$, to a price vector $\boldsymbol{p} = \gamma(\boldsymbol{\lambda})$, subject to the constraint that $\boldsymbol{p}$ is feasible for $\boldsymbol{\lambda}$.[2] The pricing policies are used as the benchmarks of our online learning framework: Given a pricing policy γ, consider a decision maker that repeatedly plays according to γ; namely, she posts the price vector $\boldsymbol{p}_t^{\gamma} = \gamma(\boldsymbol{\lambda}_t^{\gamma})$ at time $t = 1, \ldots, T$, where $\boldsymbol{\lambda}_t^{\gamma}$ is the inventory vector at time t obtained by applying γ recursively on previous inventory vectors $\boldsymbol{\lambda}_{t'}^{\gamma}$ and posting prices $\gamma(\boldsymbol{\lambda}_{t'}^{\gamma})$ at times $t' = 1, \ldots, t-1$. Denoting $\hat{q}_t^{\gamma} = \hat{q}_t^{\boldsymbol{p}_t^{\gamma}}$, the revenue of this decision maker is given by $\sum_{t=1}^{T} \hat{q}_t^{\gamma}$.

Now, consider a collection Γ of pricing policies. The quality of a posted price mechanism $\{\boldsymbol{p}_t\}_{t=1}^{T}$ is measured by means of the decision maker's *regret* that compares her own revenue to the revenue generated by the in-hindsight best pricing policy in Γ. Formally, the regret (with respect to Γ) is defined to be

$$\max_{\gamma \in \Gamma} \sum_{t=1}^{T} \hat{q}_t^{\gamma} - \mathbb{E}\left[\sum_{t=1}^{T} \hat{q}_t^{\boldsymbol{p}_t}\right],$$

where the expectation is taken over the decision maker's randomness. The mechanism is said to have *vanishing regret* if it is guaranteed that the decision maker's regret is sublinear in T, which means that the average regret per time unit vanishes as $T \to \infty$.

3 Dynamic Posted Pricing via Dd-MDP with Chasability

The online learning framework underlying the DRACC problem as defined in Section 2 is stateful with the inventory vector $\boldsymbol{\lambda}$ playing the role of the framework's state. In the current section, we first introduce a generalization of this online learning framework in the form of a stateful online decision making, formalized by means of *dynamic deterministic Markov decision processes (Dd-MDPs)*. Following that, we propose a structural condition called *chasability* and show that under this condition, the Dd-MDP problem is amenable to vanishing-regret online learning algorithms. This last result is obtained through a reduction to the extensively studied problem of "experts with switching cost" (Kalai and Vempala [57]). Finally, we prove that the Dd-MDP instances that correspond to the DRACC problem indeed satisfy the chasability condition.

[1] The techniques we use in this paper are applicable also to the objective of maximizing the social welfare.

[2] The seemingly more general setup, where the time t is passed as an argument to γ on top of $\boldsymbol{\lambda}$, can be easily reduced to our setup (e.g., by introducing a dummy resource i_t active only in round t).

3.1 Viewing DRACC as a Dd-MDP

A (static) *deterministic Markov decision process (d-MDP)* is defined over a set $\mathcal{S}$ of states and a set $\mathcal{X}$ of actions. Each state $s \in \mathcal{S}$ is associated with a subset $X_s \subseteq \mathcal{X}$ of actions called the feasible actions of s. A *state transition function* g maps each state $s \in \mathcal{S}$ and action $x \in X_s$ to a state $g(s, x) \in \mathcal{S}$. This induces a directed graph over $\mathcal{S}$, termed the *state transition graph*, where an edge labeled by $\langle s, x \rangle$ leads from node s to node s' if and only if $g(s, x) = s'$. The d-MDP also includes a *reward function* f that maps each state-action pair $\langle s, x \rangle$ with $s \in \mathcal{S}$ and $x \in X_s$ to a real value in $[0, 1]$.

Dynamic deterministic MDPs. Notably, static d-MDPs are *not* rich enough to capture the dynamic aspects of the DRACC problem. We therefore introduce a more general object where the state transition and reward functions are allowed to develop in an (adversarial) dynamic fashion.

Consider a sequential game played between an online decision maker and an adversary. As in static d-MDPs, the game is defined over a set $\mathcal{S}$ of states, a set $\mathcal{X}$ of actions, and a feasible action set X_s for each $s \in \mathcal{S}$. We further assume that the state and action sets are finite. The game is played in $T \in \mathbb{N}$ rounds as follows. The decision maker starts from an initial state $s_1 \in \mathcal{S}$. In each round $t = 1, \ldots, T$, she plays a (randomized) feasible action $x_t \in X_{s_t}$, where $s_t \in \mathcal{S}$ is the state at the beginning of round t. Simultaneously, the adversary selects the state transition function g_t and the reward function f_t. The decision maker then moves to a new state $s_{t+1} = g_t(s_t, x_t)$ (which is viewed as a movement along edge $\langle s_t, x_t \rangle$ in the state transition graph induced by g_t), obtains a reward $f_t(s_t, x_t)$, and finally, observes g_t and f_t as the current round's (full information) feedback.[3] The game then advances to the next round $t + 1$. The goal is to maximize the expected total reward $\mathbb{E}[\sum_{t \in [T]} f_t(s_t, x_t)]$.

Policies, simulation, & regret. A *policy* $\gamma : \mathcal{S} \mapsto \mathcal{X}$ is a function that maps each state $s \in \mathcal{S}$ to a feasible action $\gamma(s) \in X_s$. A *simulation* of policy γ over the round interval $[1, T]$ is given by the state sequence $\{s^\gamma(t)\}_{t=1}^T$ and the action sequence $\{x^\gamma(t)\}_{t=1}^T$ defined by setting

$$s^\gamma(t) \triangleq \begin{cases} s_1 & \text{if } t = 1 \\ g_{t-1}\left(s^\gamma(t-1), x^\gamma(t-1)\right) & \text{if } t > 1 \end{cases} \quad \text{and} \quad x^\gamma(t) \triangleq \begin{cases} \gamma(s_1) & \text{if } t = 1 \\ \gamma\left(s^\gamma(t)\right) & \text{if } t > 1 \end{cases} . \tag{2}$$

The cumulative reward obtained by this simulation of γ is given by $\sum_{t \in [T]} f_t\left(s^\gamma(t), x^\gamma(t)\right)$.

Consider a decision maker that plays the sequential game by following the (randomized) state sequence $\{s_t\}_{t=1}^T$ and action sequence $\{x_t\}_{t=1}^T$, where $x_t \in X_{s_t}$ for every $1 \leq t \leq T$. For a (finite) set Γ of policies, the decision maker's *regret* with respect to Γ is defined to be

$$\max_{\gamma \in \Gamma} \sum_{t \in [T]} f_t\left(s^\gamma(t), x^\gamma(t)\right) - \sum_{t \in [T]} \mathbb{E}\left[f_t(s_t, x_t)\right] . \tag{3}$$

<u>Relation to the DRACC Problem:</u>

Dynamic posted pricing for the DRACC problem can be modeled as a Dd-MDP. To this end, we identify the state set $\mathcal{S}$ with the set of possible inventory vectors $\boldsymbol{\lambda}_t$, $t = 1, \ldots, T$. If state $s \in \mathcal{S}$ is identified with inventory vector $\boldsymbol{\lambda}_t$, then we identify X_s with the set of price vectors feasible for $\boldsymbol{\lambda}_t$. The reward function f_t is defined by setting

$$f_t(s, x) = \hat{q}_t^x , \tag{4}$$

where $\hat{q}_t^x$ is defined as in Eq. (1), recalling that the valuation function v_t, required for the computation of $\hat{q}_t^x$, is available to the decision maker at the end of round t. As for the state transition function g_t, the new state $s' = g_t(s, x)$ is the inventory vector obtained by posting the price vector x to user t given the inventory vector s, namely,

$$s'(i) = \begin{cases} s(i) - 1_{i \in D_t^x} & \text{if } i \in A_{t+1} \cap A_t \\ c(i) & \text{if } i \in A_{t+1} \setminus A_t \end{cases} .$$

[3]No (time-wise) connectivity assumptions are made for the dynamic transition graph induced by $\{g_t\}_{t=1}^T$, hence it may not be possible to devise a path between two given states as is done in Dekel and Hazan [40] for static d-MDPs.

Given the aforementioned definitions, the notion of (pricing) policies and their recursive simulations and the notion of regret translate directly from the DRACC setting to that of Dd-MDPs.

3.2 The Chasability Condition

As the Dd-MDP framework is very inclusive, it is not surprising that in general, it does not allow for vanishing regret.

Proposition 1. *For every online learning algorithm, there exists a T-round Dd-MDP instance for which the algorithm's regret is $\Omega(T)$.*

Proof. Consider a simple scenario where there are only two states $\{s, s'\}$ with s being the initial state and two actions $\{x, x'\}$ that are feasible for both states. Without loss of generality, let x be the action that the decision maker's algorithm chooses with probability at least $1/2$ at time $t = 1$. Now, consider an adversary that works in the following manner: It sets $f_t(s, \cdot) = 1$ and $f_t(s', \cdot) = 0$ for every $t \in [T]$. Regarding the state transition, the adversary sets $g_1(s, x) = s'$ and $g_1(s, x') = s$; and for every $t \in [2, T]$, it sets $g_t(s, \cdot) = s$ and $g_t(s', \cdot) = s'$. In such case, the expected cumulative reward of the decision maker is at most $1 + T/2$, while the policy that always plays action x' obtains a cumulative reward of T. $\square$

As a remedy to the impossibility result established in Proposition 1, we introduce a structural condition for Dd-MDPs that makes them amenable to online learning with vanishing regret.

Definition 1 (*Chasability condition for Dd-MDPs*)**.** A Dd-MDP instance is called *σ-chasable* for some $\sigma > 0$ if it admits an ongoing *chasing oracle* $\mathcal{O}^{\texttt{Chasing}}$ that works as follows for any given target policy $\gamma \in \Gamma$. The chasing oracle is invoked at the beginning of some round t_{init} and provided with an initial state $s_{\text{init}} \in \mathcal{S}$; this invocation is halted at the end of some round $t_{\text{final}} \geq t_{\text{init}}$. The halt time t_{final} is specified by an oblivious adversary, while $\mathcal{O}^{\texttt{Chasing}}$ is unaware of t_{final} before it is terminated. In each round $t_{\text{init}} \leq t \leq t_{\text{final}}$, the chasing oracle generates a (random) action $\hat{x}(t)$ that is feasible for state

$$\hat{s}(t) = \begin{cases} s_{\text{init}} & \text{if } t = t_{\text{init}} \\ g_{t-1}\left(\hat{s}(t-1), \hat{x}(t-1)\right) & \text{if } t_{\text{init}} < t \leq t_{\text{final}} \end{cases} ; \tag{5}$$

following that, the chasing oracle is provided with the Dd-MDP's reward function $f_t(\cdot, \cdot)$ and state transition function $g_t(\cdot, \cdot)$. The main guarantee of $\mathcal{O}^{\texttt{Chasing}}$ is that its *chasing regret (CR)* satisfies

$$\text{CR} \triangleq \sum_{t=t_{\text{init}}}^{t_{\text{final}}} f_t\left(s^{\gamma}(t), x^{\gamma}(t)\right) - \sum_{t=t_{\text{init}}}^{t_{\text{final}}} \mathbb{E}\left[f_t\Big(\hat{s}(t), \hat{x}(t)\Big)\right] \leq \sigma .$$

We emphasize that the initial state s_{init} provided to the chasing oracle may differ from $s^{\gamma}(t_{\text{init}})$. The subscript σ of the notation $\mathcal{O}^{\texttt{Chasing}}$ may be omitted when it can be inferred from the context.

Relation to the DRACC Problem (continued):

In terms of the DRACC problem, chasability means that the online algorithm can simulate a given pricing policy, while incurring a small revenue loss, even if the online algorithm starts from a (coordinate-wise) smaller inventory vector. Interestingly, the Dd-MDPs corresponding to DRACC instances are σ-chasable for $\sigma = o(T)$, where the exact bound on σ depends on whether we consider general or k_t-demand valuation functions. Before establishing these bounds, we show that the chasing oracle must be randomized.

Proposition 2. *There exists a family of T-round DRACC instances whose corresponding Dd-MDPs do not admit a deterministic chasing oracle with $o(T)$ chasing regret CR.*

Proof. Consider an ongoing chasing oracle that is implemented in a deterministic manner for a DRACC instance with $C = 1$, $W = 2$. The adversary chooses initial step t_{init} and initial state s_{init} so that $t_{\text{init}} = o(T)$, $|A_{t_{\text{init}}}| = 2$, and $\boldsymbol{\lambda}_{t_{\text{init}}} = \langle 0, 1\rangle$. Note that throughout this proof, the inventory vectors and price vectors containing two elements are presented in an ordered way, which means that the first element corresponds to the resource with the smaller index.

The target policy γ is chosen to have $\boldsymbol{\lambda}^{\gamma}_{t_{\text{init}}} = \langle 1, 1\rangle$. Moreover, it maps every inventory vector to a price vector of $\langle \frac{2}{3}, \frac{1}{3}\rangle$. The adversary ensures the feasibility of such a policy by setting $t_e(i) = t$ for each resource that is sold out at t with the price vector generated by γ, and setting $t_a(i') = t + 1$ for a new resource. With this setting, it holds for every $t \geq t_{\text{init}}$ that $\boldsymbol{\lambda}^{\gamma}_t = \langle 1, 1\rangle$.

The adversary configures the valuation functions v_t for each $t \geq t_{\text{init}}$ in an adaptive way, and ensures that for all such t

$$\hat{\boldsymbol{\lambda}}_t = \langle 0, 1\rangle\,. \tag{6}$$

With the initial state s_{init} chosen by the adversary, Eq. (6) holds for t_{init}. Suppose it holds for some $t \geq t_{\text{init}}$. Then the price vector $\hat{\boldsymbol{p}}$ generated by the oracle must be in the form of $\langle 1, p\rangle$ for some $p \in (0, 1]$. Let i and i' be the two resources in A_t with $i < i'$. If $p \leq \frac{1}{3}$, the adversary sets $v_t(i) = \frac{2}{3}$ and $v_t(i') = \frac{1}{3}$. Then with the price vector generated by γ, payment $\frac{2}{3}$ is obtained from the user for resource i, while the oracle obtains payment $\frac{1}{3}$ from the user for i'. The difference in rewards is

$$f_t(\boldsymbol{\lambda}^{\gamma}_t, \boldsymbol{p}^{\gamma}_t) - f_t(\hat{\boldsymbol{\lambda}}_t, \hat{\boldsymbol{p}}_t) = \frac{1}{3}\,. \tag{7}$$

Moreover, since i is sold out with $\boldsymbol{p}^{\gamma}_t$, the adversary sets $t_e(i) = t$ and $t_a(i'') = t + 1$ for a new resource $i'' > i'$. In such case, it is guaranteed that Eq. (6) holds for $t + 1$.

For the case where $p > \frac{1}{3}$, it can be verified that Eq. (7) still holds for t and Eq. (6) holds for $t + 1$ when the adversary sets $v_t(i) = v_t(i') = \frac{1}{3}$. Since Eq. (7) is established for every $t \geq t_{\text{init}}$, $CR = \frac{1}{3}(t_{\text{final}} - t_{\text{init}})$. With $t_{\text{init}} = o(T)$, taking $t_{\text{final}} = T$ gives the desired bound. □

We now turn to study chasing oracles for DRACC instances implemented by randomized procedures.

Theorem 1. The Dd-MDPs corresponding to T-round DRACC instances with k_t-demand valuation functions are $O(\sqrt{CW \cdot T})$-chasable.

Proof. Consider some DRACC instance and fix the target pricing policy $\gamma \in \Gamma$; in what follows, we identify γ with a decision maker that repeatedly plays according to γ. Given an initial round t_{init} and an initial inventory vector $\hat{\boldsymbol{\lambda}}_{t_{\text{init}}}$, we construct a randomized chasing oracle $\mathcal{O}^{\texttt{Chasing}}$ that works as follows until it is halted at the end of round $t_{\text{final}} \geq t_{\text{init}}$. For each round $t_{\text{init}} \leq t \leq t_{\text{final}}$, recall that $\boldsymbol{\lambda}^{\gamma}_t$ is the inventory vector at time t obtained by running γ from round 1 to t, and let $\hat{\boldsymbol{\lambda}}_t$ be the inventory vector at time t obtained by $\mathcal{O}^{\texttt{Chasing}}$ as defined in Eq. (5). We partition the set A_t of resources active at time t into $\texttt{Good}_t = \{i \in A_t \mid \boldsymbol{\lambda}^{\gamma}_t(i) \leq \hat{\boldsymbol{\lambda}}_t(i)\}$ and $\texttt{Bad}_t = A_t \setminus \texttt{Good}_t$. In each round $t_{\text{init}} \leq t \leq t_{\text{final}}$, the chasing oracle posts the ($|A_t|$-dimensional) all-1 price vector with probability ϵ, where $\epsilon \in (0, 1)$ is a parameter to be determined later on; and it posts the price vector

$$\hat{\boldsymbol{p}}_t = \begin{cases} \boldsymbol{p}^{\gamma}_t(i) & \text{if } i \in \texttt{Good}_t \\ 1 & \text{if } i \in \texttt{Bad}_t \end{cases}$$

with probability $1 - \epsilon$, observing that this price vector is feasible for $\hat{\boldsymbol{\lambda}}_t$ by the definition of $\texttt{Good}_t$ and $\texttt{Bad}_t$. Notice that $\mathcal{O}^{\texttt{Chasing}}$ never sells a resource $i \in \texttt{Bad}_t$ and that $\hat{\boldsymbol{p}}_t(i) \geq \boldsymbol{p}^{\gamma}_t(i)$ for all $i \in A_t$. Moreover, if resource i arrives at time $t_a(i) = t > t_{\text{init}}$, then $i \in \texttt{Good}_t$.

To analyze the CR, we classify the rounds in $[t_{\text{init}}, t_{\text{final}}]$ into two classes called `Following` and `Missing`: round t is said to be `Missing` if at least one (unit of a) resource in $\texttt{Bad}_t$ is sold by γ in this round; otherwise, round t is said to be `Following`. For each `Following` round t, if $\mathcal{O}^{\texttt{Chasing}}$ posts $\hat{\boldsymbol{p}}_t$ in round t, then $\mathcal{O}^{\texttt{Chasing}}$ sells exactly the same resources as γ for the exact same prices; otherwise ($\mathcal{O}^{\texttt{Chasing}}$ posts the all-1 price vector in round t), $\mathcal{O}^{\texttt{Chasing}}$ does not sell any resource. Hence, the CR increases in round t by at most ϵ in expectation. For each `Missing` round t, the CR increases in round t by at most 1. Therefore the total CR over the interval $[t_{\text{init}}, t_{\text{final}}]$ is upper bounded by $\epsilon \cdot \mathbb{E}[\#\mathtt{F}] + \mathbb{E}[\#\mathtt{M}] \leq \epsilon \cdot T + \mathbb{E}[\#\mathtt{M}]$, where $\#\mathtt{F}$ and $\#\mathtt{M}$ denote the number of `Following` and `Missing` rounds, respectively.

To bound $\mathbb{E}[\#\mathtt{M}]$, we introduce a potential function $\phi(t)$, $t_{\text{init}} \leq t \leq t_{\text{final}}$, defined by setting

$$\phi(t) \;=\; \sum_{i \in \mathtt{Bad}_t} \boldsymbol{\lambda}_t^{\gamma}(i) - \hat{\boldsymbol{\lambda}}_t(i)$$

By definition, $\phi(t_{\text{init}}) \leq CW$ and $\phi(t_{\text{final}}) \geq 0$. We argue that $\phi(t)$ is non-increasing in t. To this end, notice that if t is a `Following` round, then γ and $\mathcal{O}^{\mathtt{Chasing}}$ sell exactly the same set of resources in $\mathtt{Good}_t$ with the same prices. This implies that the inventory of every resource $i \in A_{t+1} \cap A_t$ satisfies $\boldsymbol{\lambda}_{t+1}^{\gamma}(i) = \boldsymbol{\lambda}_t^{\gamma}(i)$ and $\hat{\boldsymbol{\lambda}}_{t+1}(i) = \hat{\boldsymbol{\lambda}}_t(i)$. Since for every resource $i \in A_{t+1} \setminus A_t$, we have $i \in \mathtt{Good}_{t+1}$, it holds that $\mathtt{Bad}_{t+1} \subseteq \mathtt{Bad}_t$. Then by definition, for a `Following` round t, we have

$$\phi(t+1) \;=\; \sum_{i \in \mathtt{Bad}_{t+1}} \boldsymbol{\lambda}_{t+1}^{\gamma}(i) - \hat{\boldsymbol{\lambda}}_{t+1}(i) \;=\; \sum_{i \in \mathtt{Bad}_{t+1}} \boldsymbol{\lambda}_t^{\gamma}(i) - \hat{\boldsymbol{\lambda}}_t(i) \;\leq\; \sum_{i \in \mathtt{Bad}_t} \boldsymbol{\lambda}_t^{\gamma}(i) - \hat{\boldsymbol{\lambda}}_t(i) \;=\; \phi(t)\,,$$

where the third transition holds because every resource $i \in \mathtt{Bad}_t$ satisfies $\boldsymbol{\lambda}_t^{\gamma}(i) > \hat{\boldsymbol{\lambda}}_t(i)$. If t is a `Missing` round and $\mathcal{O}^{\mathtt{Chasing}}$ posts the all-1 price vector, then $\phi(t+1) < \phi(t)$ as $\mathcal{O}^{\mathtt{Chasing}}$ sells no resource whereas γ sells at least one (unit of a) resource in $\mathtt{Bad}_t$. So, it remains to consider a `Missing` round t in which $\mathcal{O}^{\mathtt{Chasing}}$ posts the price vector $\hat{\boldsymbol{p}}_t$. Let S^{γ} and $\hat{S}$ be the sets of (active) resources sold by γ and $\mathcal{O}^{\mathtt{Chasing}}$, respectively, in round t and notice that a resource $i \in \hat{S} \setminus S^{\gamma}$ may move from $i \in \mathtt{Good}_t$ to $i \in \mathtt{Bad}_{t+1}$. The key observation now is that since v_t is a k_t-demand valuation function, it follows that $S^{\gamma} \cap \mathtt{Good}_t \subseteq \hat{S} \cap \mathtt{Good}_t$, thus $|S^{\gamma} \cap \mathtt{Bad}_t| \geq |\hat{S} \setminus S^{\gamma}|$. As both γ and $\mathcal{O}^{\mathtt{Chasing}}$ sell exactly one unit of each resource in S^{γ} and $\hat{S}$, respectively, we conclude that $\phi(t+1) \leq \phi(t)$.

Therefore, $\mathbb{E}[\#\mathtt{M}]$ is upper bounded by CW plus the expected number of `Missing` rounds in which $\phi(t)$ does not decrease. Since $\phi(t)$ strictly decreases in each `Missing` round t in which $\mathcal{O}^{\mathtt{Chasing}}$ posts the all-1 price vector, it follows that the number of `Missing` rounds in which $\phi(t)$ does not decrease is stochastically dominated by a negative binomial random variable Z with parameters CW and ϵ. Recalling that $\mathbb{E}[Z] = (1-\epsilon) \cdot CW/\epsilon$, we conclude that $\mathbb{E}[\#\mathtt{M}] \leq CW + \mathbb{E}[Z] = CW/\epsilon$. The assertion is now established by setting $\epsilon = \sqrt{CW/T}$. $\square$

Remark 1. Theorem 1 can be in fact extended – using the exact same line of arguments – to a more general family of valuation functions v_t defined as follows. Let $\boldsymbol{p}$ be a price vector, $B \subseteq A_t$ be a subset of the active resources, and $\boldsymbol{p}'$ be the price vector obtained from $\boldsymbol{p}$ by setting $\boldsymbol{p}'(i) = 1$ if $i \in B$; and $\boldsymbol{p}'(i) = \boldsymbol{p}(i)$ otherwise. Then, $|D_t^{\boldsymbol{p}} \cap B| \geq |D_t^{\boldsymbol{p}'} \setminus D_t^{\boldsymbol{p}}|$. Besides k_t-demand valuations, this class of valuation functions includes OXS valuations (Lehmann et al. [63]) and single-minded valuations (Lehmann et al. [64]).

Theorem 2. The Dd-MDPs corresponding to T-round DRACC instances with arbitrary valuation functions are $O\left(T^{\frac{CW}{CW+1}}\right)$-chasable.

Proof. The proof follows the same line of arguments as that of Theorem 1, only that now, it no longer holds that the potential function $\phi(t)$ is non-increasing in t. However, it is still true that (I) $0 \leq \phi(t) \leq CW$ for every $t_{\text{init}} \leq t \leq t_{\text{final}}$; (II) if $t_{\text{init}} \leq t < t_{\text{final}}$ is a `Missing` round and $\mathcal{O}^{\mathtt{Chasing}}$ posts the all-1 price vector in round t, then $\phi(t+1) < \phi(t)$; and (III) if $\phi(t) = 0$ for some $t_{\text{init}} \leq t \leq t_{\text{final}}$, then $\phi(t') = 0$ for all $t < t' \leq t_{\text{final}}$. We conclude that if $\mathcal{O}^{\mathtt{Chasing}}$ posts the all-1 price vector in CW contiguous `Missing` rounds, then $\phi(\cdot)$ must reach zero and following that, there are no more `Missing` rounds. Therefore the total number $\#\mathtt{M}$ of `Missing` rounds is stochastically dominated by CW times a geometric random variable Z with parameter ϵ^{CW}. Since $\mathbb{E}[Z] = \epsilon^{-CW}$, it follows that $\mathbb{E}[\#\mathtt{M}] \leq CW/\epsilon^{CW}$. Combined with the `Following` rounds, the CR is upper bounded by $\epsilon \cdot T + CW/\epsilon^{CW}$. The assertion is established by setting $\epsilon = (T/(CW))^{-1/(CW+1)}$. $\square$

3.3 Putting the Pieces Together: Reduction to Online Learning with Switching Cost

Having an ongoing chasing oracle with vanishing chasing regret in hand, our remaining key technical idea is to reduce online decision making for the Dd-MDP problem to the well-studied problem of *online learning*

ALGORITHM 1: Online Dd-MDP algorithm C&S

Input: Policy set Γ, OLSC algorithm $\mathcal{A}$, chasing oracle $\mathcal{O}^{\texttt{Chasing}}$, initial state s_1;
Output: Sequence $x_1, \ldots, x_T$ of actions, (implicit) sequence $s_2, \ldots, s_T$ of states;

Start from initial state s_1;
for *each round* $t \in [T]$ **do**
    Invoke $\mathcal{A}$ to pick a policy γ_t at the beginning of round t;
    if $t > 1$ *and* $\gamma_t \neq \gamma_{t-1}$ **then**
        Invoke $\mathcal{O}^{\texttt{Chasing}}$ from scratch with target policy γ_t, initialized with round t and state s_t;
        Select the action $x_t \leftarrow \hat{x}(t)$ returned by $\mathcal{O}^{\texttt{Chasing}}$;
    else
        Continue the existing run of $\mathcal{O}^{\texttt{Chasing}}$ and select the action $x_t \leftarrow \hat{x}(t)$ it returns;
    Feed $\mathcal{O}^{\texttt{Chasing}}$ with $g_t(\cdot, \cdot)$ and $f_t(\cdot, \cdot)$ as the state transition and reward functions of round t;
    for *each* $\gamma \in \Gamma$ **do**
        Compute $F_t(\gamma) \leftarrow f_t(s^\gamma(t), x^\gamma(t))$ by simulating policy γ up to time t (see Eq. (2));
    Feed $\mathcal{A}$ with $F_t(\cdot)$ as the reward function of round t;

with switching cost (OLSC) (Kalai and Vempala [57]). The problem's setup under full-information is exactly the same as the classic problem of learning from experts' advice, but the learner incurs an extra cost $\Delta > 0$, a parameter referred to as the *switching cost*, whenever it switches from one expert to another. Here, we have a finite set Γ of experts (often called actions or arms) and $T \in \mathbb{Z}_{>0}$ rounds. The expert reward function $F_t : \Gamma \mapsto [0, 1)$ is revealed as feedback at the end of round $t = 1, \ldots, T$. The goal of an algorithm for this problem is to pick a sequence $\gamma_1, \ldots, \gamma_T$ of experts in an online fashion with the objective of minimizing the regret, now defined to be

$$\max_{\gamma \in \Gamma} \sum_{t \in [T]} F_t(\gamma) - \left(\sum_{t \in [T]} \mathbb{E}\left[F_t(\gamma_t)\right] - \Delta \cdot \sum_{t=2}^{T} 1_{\gamma_t \neq \gamma_{t-1}} \right) .$$

Theorem 3 (Kalai and Vempala [57])**.** The OLSC problem with switching cost Δ admits an online algorithm $\mathcal{A}$ whose regret is $O\left(\sqrt{\Delta \cdot T \log |\Gamma|}\right)$.

Note that the same theorem also holds for independent stochastic switching costs with Δ as the upper bound on the expected switching cost, simply because of linearity of expectation and the fact that in algorithms for OLSC, such as the Following-The-Perturbed-Leader (Kalai and Vempala [57]), switching at each time is independent of the realized cost of switching.

We now present our full-information online learning algorithm for σ-chasable Dd-MDP instances; the reader is referred to Section 6 for the bandit version of this algorithm. Our (full-information) algorithm, called *chasing and switching (C&S)*, requires a black box access to an algorithm $\mathcal{A}$ for the OLSC problem with the following configuration: (1) the expert set of $\mathcal{A}$ is identified with the policy collection Γ of the Dd-MDP instance; (2) the number of rounds of $\mathcal{A}$ is equal to the number of rounds of the Dd-MDP instance (T); and (3) the switching cost of $\mathcal{A}$ is set to $\Delta = \sigma$.

The operation of C&S is described in Algorithm 1. This algorithm maintains, in parallel, the OLSC algorithm $\mathcal{A}$ and an ongoing chasing oracle $\mathcal{O}^{\texttt{Chasing}}$; $\mathcal{A}$ produces a sequence $\{\gamma_t\}_{t=1}^T$ of policies and $\mathcal{O}^{\texttt{Chasing}}$ produces a sequence $\{x_t\}_{t=1}^T$ of actions based on that. Specifically, $\mathcal{O}^{\texttt{Chasing}}$ is *restarted*, i.e., invoked from scratch with a fresh policy γ, whenever $\mathcal{A}$ switches to γ from some policy $\gamma' \neq \gamma$. Note that algorithm $\mathcal{A}$ and the chasing oracle $\mathcal{O}^{\texttt{Chasing}}$ do not share the same source of random bits. Therefore, it is appropriate to assume that the halt time t_{final} of the chasing oracle is generated by an oblivious adversary.

Theorem 4. The regret of C&S for T-round σ-chasable Dd-MDP instances is $O\left(\sqrt{\sigma \cdot T \log |\Gamma|}\right)$.

Proof. Partition the T rounds into episodes $\{1, 2, \dots\}$ so that each episode θ is a maximal contiguous sequence of rounds in which the policy γ_θ chosen by $\mathcal{A}$ does not change. Let t_θ and t'_θ be the first and last rounds of episode θ, respectively. Consider some episode θ with corresponding policy γ_θ. Since C&S follows an action sequence generated by $\mathcal{O}^{\texttt{Chasing}}$ during the round interval $[t_\theta, t'_\theta]$ and since the chasing regret of $\mathcal{O}^{\texttt{Chasing}}$ is upper bounded by $\sigma = \Delta$, it follows that

$$\sum_{t=t_\theta}^{t'_\theta} F_t(\gamma_\theta) - \sum_{t=t_\theta}^{t'_\theta} \mathbb{E}\left[f_t(s_t, x_t)\right] = \sum_{t=t_\theta}^{t'_\theta} f_t\left(s^{\gamma_\theta}(t), x^{\gamma_\theta}(t)\right) - \sum_{t=t_\theta}^{t'_\theta} \mathbb{E}\left[f_t(s_t, x_t)\right] \leq \Delta .$$

Therefore, for each policy $\gamma \in \Gamma$, we have

$$\begin{aligned}\sum_{t\in[T]} f_t\left(s^{\gamma}(t), x^{\gamma}(t)\right) - \sum_{t\in[T]} \mathbb{E}\left[f_t(s_t, x_t)\right] &\leq \sum_{t\in[T]} f_t\left(s^{\gamma}(t), x^{\gamma}(t)\right) - \sum_{\theta}\left(\sum_{t=t_\theta}^{t'_\theta} \mathbb{E}\left[F_t(\gamma_\theta)\right] - \Delta\right) \\ &= \sum_{t\in[T]} F_t(\gamma) - \left(\sum_{t\in[T]} \mathbb{E}\left[F_t(\gamma_t)\right] - \Delta \cdot \sum_{t=2}^{T} 1_{\gamma_t \neq \gamma_{t-1}}\right) .\end{aligned}$$

By Theorem 3, the last expression is at most $O\left(\sqrt{\Delta \cdot T \log|\Gamma|}\right) = O\left(\sqrt{\sigma \cdot T \log|\Gamma|}\right)$. $\square$

So far, we have only considered the notion of *policy regret* as defined in Eq. (3). An extension of our results to the notion of *external regret* (Arora et al. [12]) is discussed in Section 5. Furthermore, we investigate the bandit version of the problem in Section 6. In a nutshell, by introducing a stateless version of our full-information chasing oracle and reducing to the adversarial multi-armed-bandit problem (Audibert and Bubeck [13]), we obtain $O(T^{2/3})$ regret bound for Dd-MDP under bandit feedback. Finally, we obtain near-matching lower bounds for both the full-information and bandit feedback versions of the Dd-MDP problem under the chasability condition in Section 7.

Relation to the DRACC Problem (continued):

We can now use C&S (Algorithm 1) for Dd-MDPs that correspond to DRACC instances. This final mechanism is called *learning based posted pricing (*`LBPP`*)*. It first provides the input parameters of C&S, including the collection Γ of pricing policies, the OLSC algorithm $\mathcal{A}$, the ongoing chasing oracle $\mathcal{O}^{\texttt{Chasing}}$ and the initial state s_1. It then runs C&S by posting its price vectors (actions) and updating the resulting inventory vectors (states). For $\mathcal{O}^{\texttt{Chasing}}$, we employ the (randomized) chasing oracles promised in Theorem 1 and Theorem 2. The following theorems can now be inferred from Theorem 4, Theorem 1, and Theorem 2.

Theorem 5. The regret of `LBPP` for T-round DRACC instances with k_t-demand valuation functions (or more generally, with the valuation functions defined in Remark 1) is $O\left((CW)^{\frac{1}{4}} T^{\frac{3}{4}} \sqrt{\log|\Gamma|}\right)$.

Theorem 6. The regret of `LBPP` for T-round DRACC instances with with arbitrary valuation functions is $O\left(T^{\frac{1}{2}\left(1+\frac{CW}{CW+1}\right)} \sqrt{\log|\Gamma|}\right)$.

Note that the regret bounds in Theorem 5 and Theorem 6 depend on the parameters C and W of the DRACC problem; as shown in the following theorem, such a dependence is unavoidable.

Theorem 7. If $C \cdot W = \Omega(T)$, then the regret of any posted price mechanism is $\Omega(T)$.

Proof. Here we construct two instances of DRACC. The following settings are the same between these two instances.

- The parameters C and W are chosen so that $C \cdot W = T/2$. Set $N = W$.

- For each resource i, $t_a(i) = 1$ and $t_e(i) = T$. This setting implies that for every user t, $A_t = [N]$, which is consistent with $W = N$. Every $i \in [N]$ has the same capacity $c(i) = C$.

- For each user $t \in \Big[1, T/2\Big]$, the valuation function v_t is set as follows.

$$v_t(A') = \begin{cases} \frac{1}{2} & \text{if } |A'| = 1 \\ 0 & \text{otherwise} \end{cases} \quad \forall A' \subseteq A_t\,.$$

For the users $t \in \Big[\frac{T}{2}+1, T\Big]$, their valuation functions are different between the two instances. In particular, in the first instance, $v_t(A') = 0$ for any $A' \subseteq A_t$, while in the second instance

$$v_t(A') = \begin{cases} 1-\epsilon & \text{if } |A'| = 1 \\ 0 & \text{otherwise} \end{cases} \quad \forall A' \subseteq A_t\,.$$

where ϵ is some small enough constant in $(0, \frac{1}{2})$.

Now consider an arbitrary deterministic mechanism $\mathcal{M}$. Such a mechanism will output the same sequence of price vectors for the first half of the users in these two instances. Therefore, the total number of resources that are allocated by $\mathcal{M}$ to the first half of users must be the same k in the two instances for some $k \in \Big[0, \frac{T}{2}\Big]$. Then, the revenue of $\mathcal{M}$ is at most $\frac{k}{2}$ in the former instance, while at most $\frac{k}{2} + \Big(\frac{T}{2} - k\Big) \cdot (1-\epsilon) = \frac{1-\epsilon}{2}T - (\frac{1}{2} - \epsilon)k$ in the latter one. Now consider a pricing policy γ that maps every inventory vector except $\big\langle 0 \big\rangle$ to a price vector that only contains $\frac{1}{2}$. The revenue of γ in the first instance is $\frac{T}{4}$. Similarly, there exists a policy γ' with revenue $\frac{T}{2} \cdot (1-\epsilon)$ in the second instance. Therefore, the regret of $\mathcal{M}$ is at least

$$\max\left\{\frac{T}{4} - \frac{k}{2},\ \frac{T}{2}(1-\epsilon) - \Big[\frac{1-\epsilon}{2}T - (\frac{1}{2} - \epsilon)k\Big]\right\} \geq \frac{1-2\epsilon}{8(1-\epsilon)}T\,.$$

To generalize the result above to the mechanisms that can utilize the random bits, here we adopt Yao's principle (Yao [75]). In particular, we construct a distribution over the inputs which assigns probabilities $\frac{1-2\epsilon}{2-2\epsilon}$ and $\frac{1}{2-2\epsilon}$ to the two instances constructed above, respectively. It can be verified that against such a distribution, the expectation of any random mechanism's regret is at least $\frac{1-2\epsilon}{8(1-\epsilon)}T$. By Yao's principle, the lower bound on the regret of any mechanism that can utilizes the random bits is also $\frac{1-2\epsilon}{8(1-\epsilon)}T$. Therefore, this proposition is established. ☐

Theorem 8. There is no posted price mechanism has a regret that is better than $O(\sqrt{CW \cdot T})$ for DRACC when each user t has a k_t-demand valuation function with $k_t = 1$.

Proof. Consider the case where the T rounds are partitioned into T' episodes by the adversary such that the length of each episode ψ contains exactly $2CW$ rounds. At the beginning of the first round in each episode ψ, a set of W new resources with a uniform capacity C arrive, and these resources depart at the end of the last round in the same episode. For each user t arrive in the first CW rounds of each episode, the parameter w_t^i of her valuation function is set to $\frac{1-\epsilon}{2}$ for every active resource $i \in A_t$, where $\epsilon > 0$ is a small enough positive number. The adversary flips a coin at the beginning of each episode. If the result of the toss is HEAD, then for every user t arriving in the last CW rounds of the current episode and every active resource $i \in A_t$, $v_t\big(\{i\}\big)$ is set to $1-\epsilon$. Otherwise, $v_t\big(\{i\}\big)$ is set to 0. It is easy to see that the expected revenue of any online algorithm is at most $\frac{1-\epsilon}{2}T$.

Now assume that there are two different benchmark policies $\gamma_1, \gamma_2 \in \Gamma$. For each non-exhausted resource i, policy γ_1 sets a price of $\frac{1-\epsilon}{2}$, while policy γ_2 specifies a price of $1-\epsilon$. For each episode ψ, let X_ψ be a random variable that takes 1 with probability $\frac{1}{2}$ and takes -1 with probability $\frac{1}{2}$. Then the expected reward obtained by the in-hindsight best pricing policy is

$$\begin{aligned}\mathbb{E}\left[\max\left\{\sum_{t\in[T]} f_t(s^{\gamma_1}(t), x^{\gamma_1}(t)), \sum_{t\in[T]} f_t(s^{\gamma_2}(t), x^{\gamma_2}(t))\right\}\right] =&\mathbb{E}\left[\max\left\{\frac{1-\epsilon}{2}T, \frac{1-\epsilon}{2}CW\cdot\sum_{\psi\in T'}(1+X_\psi)\right\}\right]\\ =&\frac{1-\epsilon}{2}T+\frac{1-\epsilon}{2}CW\cdot\mathbb{E}\left[\max\left\{0,\sum_{\psi\in T'}X_\psi\right\}\right]\\ =&\frac{1-\epsilon}{2}T+\frac{1-\epsilon}{2}\Omega\left(CW\cdot\sqrt{T'}\right)\\ =&\frac{1-\epsilon}{2}T+\frac{1-\epsilon}{2}\Omega\left(\sqrt{CW\cdot T}\right).\end{aligned}$$

Above, the second transition and the fourth transition hold because $T'\cdot CW = T$. The third transition above holds because $\sum_{\psi\in[T']} X_\psi$ tends to a normal distribution $\mathcal{N}(0,\sqrt{T'})$ by using central limit theorem, which means that $\max\{0,\sum_{\psi\in T'} X_\psi\}$, a zero-mean normal distribution (in the limit) truncated from below at 0, coincides with a $\sqrt{T'}$-mean half-normal distribution. Therefore, we can conclude that compared to the revenue generated by the in-hindsight best pricing policy, any algorithm suffers a loss of $\Omega\left(\sqrt{CW\cdot T}\right)$. □

The following corollary can be directly obtained by combining Theorem 4 with Theorem 8.

Corollary 1. The CR of any chasing oracle is at least $\chi\cdot CW$ on DRACC instances with k_t-demand valuation functions, where χ is some constant larger than 0.

4 Applications of the DRACC Problem

The mechanism LBPP proposed for the DRACC problem can be directly applied to a large family of online pricing problems arising in practice. Two examples are presented in this section: the *online job scheduling (OJS)* problem and the problem of *matching over dynamic bipartite graphs (MDBG)*.

4.1 Online Job Scheduling.

The OJS problem described in this section is motivated by the application of assigning jobs that arrive online to limited bandwidth slots for maximizing the total payments collected from the jobs. Formally, in the OJS problem, there are T strategic myopic jobs, arriving sequentially over N time slots. Each slot $i\in[N]$ lasts over the time interval $[i,i+1)$ and is associated with a bandwidth $c(i)$, which means that this slot can be allocated to at most $c(i)$ jobs. For each job $t\in T$, the adversary specifies an arrival slot $1\le a_t\le N$, a departure slot $a_t\le d_t\le N$, a length $1\le l_t\le d_t-a_t+1$, and a value $v_t\in[0,1)$. We emphasize that any number (including zero) of jobs may have slot i as their arrival (or departure) slot. The goal of job t is to get an allocation of l_t contiguous slots within $[a_t,d_t]$, namely, a slot interval in

$$\mathcal{I}_t = \{[i,i+l_t-1]\mid a_t\le i\le d_t-l_t+1\},$$

with v_t being the job's value for each such allocation. Let C and W be upper bounds on $\max_{i\in[N]} c(i)$ and $\max_{t\in[T]} d_t-a_t+1$, respectively.

Job $t\in[T]$ is reported to the OJS mechanism at the beginning of slot a_t; if several jobs share the same arrival slot, then they are reported to the mechanism sequentially in an arbitrary order. At the beginning of slot a_t, the mechanism is also informed of the bandwidth parameter $c(i)$ of every slot $i\in A_t$, where A_t is defined to be the slot interval

$$A_t = [a_t, a_t+W-1];$$

note that the mechanism may have been informed of the bandwidth parameters of some slots in A_t beforehand (if they belong to $A_{t'}$ for $t'<t$). In response, the mechanism posts a price vector $\boldsymbol{p}_t\in(0,1]^{A_t}$ and elicits

the parameters d_t, l_t, and v_t. Subsequently, (one bandwidth unit of) the slots in the demand set $D_t^{\boldsymbol{p}_t}$ are allocated to job t at a total price of $\hat{q}_t^{\boldsymbol{p}_t}$, where

$$D_t^{\boldsymbol{p}} = \begin{cases} \emptyset & \text{if } v_t < \min_{I \in \mathcal{I}_t} \sum_{i \in I} \boldsymbol{p}(i) \\ \operatorname{argmin}_{I \in \mathcal{I}_t} \sum_{i \in I} \boldsymbol{p}(i) & \text{otherwise} \end{cases} \qquad \text{and} \qquad \hat{q}_t^{\boldsymbol{p}} = \sum_{i \in D_t^{\boldsymbol{p}}} \boldsymbol{p}(i)$$

for any price vector $\boldsymbol{p} \in (0,1]^{A_t}$, consistently breaking argmin ties according to the lexicographic order on A_t.

Let $\boldsymbol{\lambda}_t \in \{0, 1, \dots, C\}^{A_t}$ be the (remaining) bandwidth vector that encodes the number $\boldsymbol{\lambda}_t(i)$ of units remaining from the bandwidth of slot $i \in A_t$ before processing job $t = 1, \dots, T$. Formally, if slot i has not been allocated to any of the jobs in $\{1, \dots, t-1\}$, then $\boldsymbol{\lambda}_t(i) = c(i)$; and if (a bandwidth unit of) slot i is allocated to job t and $i \in A_{t+1}$, then $\boldsymbol{\lambda}_{t+1}(i) = \boldsymbol{\lambda}_t(i) - 1$. We say that a price vector $\boldsymbol{p}$ is feasible for the bandwidth vector $\boldsymbol{\lambda}_t$ if $\boldsymbol{p}(i) = 1$ for every $i \in A_t$ such that $\boldsymbol{\lambda}_t(i) = 0$, that is, for every slot i that has already been exhausted before job t is processed. To ensure that the slots' bandwidth is not exceeded, we require that the posted price vector $\boldsymbol{p}_t$ is feasible for $\boldsymbol{\lambda}_t$ for every $1 \le t \le T$. We aim for posted price OJS mechanisms whose objective is to maximize the total expected payment $\mathbb{E}[\sum_{t=1}^{T} \hat{q}_t^{\boldsymbol{p}_t}]$ received from all jobs, where the expectation is over the mechanism's internal randomness.

A pricing policy γ is a function that maps each bandwidth vector $\boldsymbol{\lambda} \in \{0, 1, \dots, C\}^{A_t}$, $t \in [T]$, to a price vector $\boldsymbol{p} = \gamma(\boldsymbol{\lambda})$, subject to the constraint that $\boldsymbol{p}$ is feasible for $\boldsymbol{\lambda}$. Given a pricing policy γ, consider a decision maker that repeatedly plays according to γ; namely, she posts the price vector $\boldsymbol{p}_t^\gamma = \gamma(\boldsymbol{\lambda}_t^\gamma)$ for job $t = 1, \dots, T$, where $\boldsymbol{\lambda}_t^\gamma$ is the bandwidth vector obtained by applying γ recursively on previous bandwidth vectors $\boldsymbol{\lambda}_{t'}^\gamma$ and posting prices $\gamma(\boldsymbol{\lambda}_{t'}^\gamma)$ for jobs $t' = 1, \dots, t-1$. Denoting $\hat{q}_t^\gamma = \hat{q}_t^{\boldsymbol{p}_t^\gamma}$, the revenue of this decision maker is given by $\sum_{t=1}^{T} \hat{q}_t^\gamma$. Given a collection Γ of pricing policies, the quality of a posted price OJS mechanism $\{\boldsymbol{p}_t\}_{t=1}^T$ is measured by means of the decision maker's regret with respect to Γ, namely

$$\max_{\gamma \in \Gamma} \sum_{t=1}^{T} \hat{q}_t^\gamma - \mathbb{E}\left[\sum_{t=1}^{T} \hat{q}_t^{\boldsymbol{p}_t}\right],$$

where the expectation is taken over the decision maker's randomness.

Reduction to DRACC. Given the aforementioned choice of notation, the transformation of an OJS instance to a DRACC instance should now be straightforward. Specifically: job t is mapped to user t; slot i is mapped to resource i; slot i's bandwidth parameter $c(i)$ is mapped to the capacity of resource i; job t's arrival slot a_t determines the set A_t of active resources at time t, and through these sets, the arrival and departure times of the resources; and job t's length l_t and value v_t parameters determine the valuation function of user t, assigning a value of v_t to each $I \in \mathcal{I}_t$; and a zero value to any other subset of A_t. The following corollary is now inferred directly from Theorem 6.

Corollary 2. The OJS problem admits a mechanism whose regret for T-round instances is:

$$O\left(T^{(1+(CW)/(CW+1))/2} \sqrt{\log |\Gamma|}\right).$$

Corollary 2 is derived from the regret bound of `LBPP` for the DRACC problem with arbitrary valuation functions, based on the (randomized) chasing oracle implementation developed in Theorem 2. It turns out though that one can exploit the structural properties of the OJS problem to design a chasing oracle with dramatically improved chasing regret, thus improving the regret bound for the OJS problem (see Corollary 3).

Lemma 1. The OJS problem admits a (deterministic) ongoing chasing oracle whose chasing regret is at most $2 \cdot CW$.

Proof. One property of OJS is that for any two slots i and i',

$$t_a(i) \le t_a(i') \quad \Rightarrow \quad t_e(i) \le t_e(i'). \tag{8}$$

We prove the claim by constructing a chasing ongoing oracle $\mathcal{O}^{\texttt{Chasing}}$ with the desired CR using this property. Given a target policy γ, an initial step t_{init}, and an initial state s_{init}, oracle $\mathcal{O}^{\texttt{Chasing}}$ posts a price vector $\hat{\boldsymbol{p}}_t$ for each $t \geq t_{\text{init}}$ as follows

$$\hat{\boldsymbol{p}}_t = \begin{cases} \langle 1 \rangle_{i \in A_t} & \text{if } t \leq \min\{T, \max_{i \in A_{t_{\text{init}}}} t_e(i)\} \\ \boldsymbol{p}_t^{\gamma} & \text{otherwise} \end{cases} .$$

Let $t' = \min\{T, \max_{i \in A_{t_{\text{init}}}} t_e(i)\}$. The price vector $\hat{\boldsymbol{p}}_t$ is trivially feasible for every $t \in [t_{\text{init}}, t']$. If $t' < T$, then for every slot i in $A_{t'+1}$, we have $i \notin A_{t_{\text{init}}}$, which gives $t_a(i) > t_{\text{init}}$. Since $\mathcal{O}^{\texttt{Chasing}}$ does not sell any slot to users from t_{init} to t', it holds that $\hat{\boldsymbol{\lambda}}_{t'+1}(i) = c(i) \geq \boldsymbol{\lambda}_{t'+1}^{\gamma}(i)$. Therefore, $\texttt{Good}_{t'+1} = A_{t'+1}$ and $\texttt{Bad}_{t'+1} = \emptyset$. Then it can be proved inductively that for any $t \geq t'+1$, $\texttt{Bad}_t = \emptyset$, which ensures the feasibility of $\hat{\boldsymbol{p}}_t$. Moreover, for each $t \geq t'+1$, since $\hat{\boldsymbol{p}}_t = \boldsymbol{p}_t^{\gamma}$, we have $f_t\big(\hat{\boldsymbol{\lambda}}_t, \hat{\boldsymbol{p}}_t\big) = f_t\big(\boldsymbol{\lambda}_t^{\gamma}, \boldsymbol{p}_t^{\gamma}\big)$.

It remains to bound $\sum_{t=t_{\text{init}}}^{t'} f_t\big(\boldsymbol{\lambda}_t^{\gamma}, \boldsymbol{p}_t^{\gamma}\big) - \sum_{t=t_{\text{init}}}^{t'} f_t\big(\hat{\boldsymbol{\lambda}}_t, \hat{\boldsymbol{p}}_t\big)$. Let S be the set of slots i with $t_a(i) \in (t_{\text{init}}, t']$. By Eq. (8), it holds for every $i \in S$ that $t_e(i) \geq t'$, because for every $i' \in A_{t_{\text{init}}}$, $t_a(i') \leq t_{\text{init}}$. By definition, $S \subseteq A_{t'}$, which gives $|S| \leq W$. Since the slots that can be sold by any policy to users in $[t_{\text{init}}, t']$ belong to $A_{t_{\text{init}}} \cup S$, we have

$$\sum_{t=t_{\text{init}}}^{t'} f_t\big(\boldsymbol{\lambda}_t^{\gamma}, \boldsymbol{p}_t^{\gamma}\big) \leq C \cdot |A_{t_{\text{init}}} \cup S| \leq C \cdot 2W .$$

Since $\sum_{t=t_{\text{init}}}^{t'} f_t\big(\hat{\boldsymbol{\lambda}}_t, \hat{\boldsymbol{p}}_t\big)$ is non-negative, this theorem is established. □

By plugging Lemma 1 into Theorem 4, we obtain the following improvement to Corollary 2; this bound is near-optimal due to Blum and Hartline [22].

Corollary 3. The OJS problem admits a mechanism whose regret for T-round instances is $O\left(\sqrt{CW \cdot T \log |\Gamma|}\right)$.

4.2 Matching Over Dynamic Bipartite Graphs.

The MDBG problem is a dynamic variation of the conventional bipartite matching problem with the goal of maximizing the revenue. Formally, in the MDBG problem, there are two sets of nodes, the *left-side* node set $\texttt{Left} = \{i\}_{i \in [N]}$ and the *right-side* node set $\texttt{Right} = \{t\}_{t \in [T]}$. The nodes in each of these two sets arrive sequentially and dynamically. For each node $i \in \texttt{Left}$, an adversary specifies a pair of parameters $t_a(i) \in [T]$ and $t_e(i) \in \Big[t_a(i), [T]\Big]$. It means that the node i arrives just before the arrival of the node $t = t_a(i) \in \texttt{Right}$, and expires immediately after the node $t' = t_e(i) \in \texttt{Right}$ is given. For each node $t \in \texttt{Right}$, define $A_t = \{i \in \texttt{Left} : t \in [t_a(i), t_e(i)]\}$. The adversary also specifies a weight $w_t(i) \in [0, 1)$ for each $t \in \texttt{Right}$ and $i \in A_t$.

A posted price mechanism is required to present a price vector $\boldsymbol{p}_t \in (0, 1]^{|A_t|}$ independently of $w_t(\cdot)$ upon the arrival of each node $t \in \texttt{Right}$. For any price vector $\boldsymbol{p}$ presented to t, define $D_t^{\boldsymbol{p}} = \operatorname{argmax}_{i \in A_t} w_t(i) - \boldsymbol{p}(i)$ with breaking ties in a fixed way. The mechanism matches the left-side node $D_t^{\boldsymbol{p}}$ to the right-side node t and charges t the payment $\boldsymbol{p}\Big(D_t^{\boldsymbol{p}}\Big)$ if $w_t\Big(D_t^{\boldsymbol{p}}\Big) \geq \boldsymbol{p}\Big(D_t^{\boldsymbol{p}}\Big)$. Otherwise, no left node is matched to t, and no payment is obtained. After that, $w_t(\cdot)$ is revealed to the mechanism.

In the MDBG problem, every left-side node i can only be matched to at most one right-side node t. We express this constraint as a feasibility requirement on the price vector that for each right-side node t, if a left-side node $i \in A_t$ has already been matched before the arrival of t, then the price of i should be set to 1. The states of whether the left-side nodes in A_t have been matched can be described with a Boolean vector of length $|A_t|$, and a pricing policy $\gamma \in \Gamma$ is a mapping from each possible Boolean vector to a feasible price vector. The objective of the MDBG problem is to find a feasible price vector $\boldsymbol{p}_t$ for every $t \in \texttt{Right}$ to maximize the total payments, and the regret is defined to be the difference between the revenue obtained by the best fixed in-hindsight pricing policy in a given collection Γ and the expected revenue of the mechanism.

Reduction to DRACC. This problem can be transformed to a special case of the DRACC problem by taking the nodes in `Left` (resp. `Right`) as the resources (resp. users). The capacity of every resource is exactly one. The valuation function of each user t maps each subset $A \subseteq A_t$ to $v_t(A) = \max_{i \in A} w_t(i)$. Such a setting is consistent because the price posted for each resource i is strictly larger than 0, which ensures that at most one resource is allocated to each user. Moreover, v_t is a k_t-demand valuation function with $k_t = 1$ for every t. Using Theorem 5, we get the following result.

Corollary 4. For the MDBG problem, the regret of the mechanism `LBPP` is bounded by $O\Big(W^{1/4}T^{3/4}\sqrt{\log|\Gamma|}\Big)$.

Remark 2. While the two applications we studied in Section 4.1 Section 4.2 could be reduced to the DRACC, our more general Dd-MDP model can have applications beyond the DRACC problem. In fact, one can tweak their models a bit so that rather than pricing policies we have arbitrary set of scheduling/matching polices with limited cost as they switch their states. Still under these variant models, our applications are instances of the Dd-MDP problem. For example, Chawla et al. [32] have considered a variant of our online job scheduling problem where we have an arbitrary set of online scheduling policies, and a decision maker's task is to learn how to switch between these policies to minimize the regret versus the best in-hindsight policy in this given set of policies. This problem is indeed an example of σ-chasable Dd-MDP, but not an instance of DRACC, as shown in Chawla et al. [32].

5 External Regret

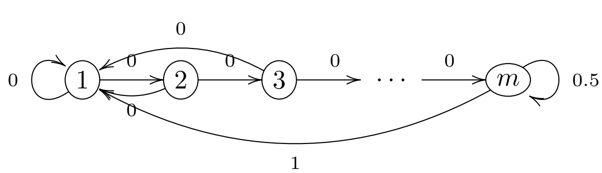
To complement our result, in this part we consider another natural alternative definition for regret, known as *external regret*, defined as follows (see Arora et al. [12] for more details).

$$\max_{\gamma \in \Gamma} \sum_{t \in [T]} f_t\Big(s_t, \gamma(s_t)\Big) - \sum_{t \in [T]} \mathbb{E}\Big[f_t(s_t, x_t)\Big]. \tag{9}$$

In words, while policy regret is the difference between the simulated reward of the optimal fixed policy and the actual reward of the algorithm, in external regret the reward that is being accredited to the optimal fixed policy in each round t is the reward that policy would have obtained when being in the actual state of the algorithm (versus being in its simulated current state). In Arora et al. [12], it is shown that for the online learning problems where the reward functions depend on the m-recent actions, the policy regret and the external regret are incomparable, which means that any algorithm with a sublinear policy regret has a linear external regret, and vice visa. Based on the techniques proposed in Arora et al. [12], we prove that such a statement also holds for the online learning problem on the Dd-MDP with chasability. This is the reason why we focus on obtaining vanishing policy regret in the main part of this paper.

Theorem 9. There exists a σ-chasable instance of the Dd-MDP so that for any online learning algorithm having a sublinear policy regret on this instance, it cannot guarantee a sublinear external regret on the same instance, and vice visa.

Proof. We start by constructing a deterministic MDP instance and proving that it is a feasible Dd-MDP instance with a constant σ.



(10)

Consider the deterministic MDP instance with $m > 2$ states in Eq. (10), where m is a constant independent of T. Each state in this instance is labeled with a distinct integer in $[m]$. The action set contains two actions, which are denoted by FORWARD and BACKWARD, respectively. These two actions are feasible for every state. The state transition functions g_t and reward functions f_t are fixed for all $t \in [T]$ as follows.

$$g_t(s,x) = \begin{cases} s+1 & \text{if } s < m \bigwedge x = \text{FORWARD} \\ m & \text{if } s = m \bigwedge x = \text{FORWARD} \\ 1 & \text{if } x = \text{BACKWARD} \end{cases},$$

$$f_t(s,x) = \begin{cases} 0.5 & \text{if } s = m \bigwedge x = \text{FORWARD} \\ 1 & \text{if } s = m \bigwedge x = \text{BACKWARD} \\ 0 & \text{otherwise} \end{cases}.$$

For any target policy $\gamma \in \Gamma$, initial time t_{init} and any initial state s_{init}, let $k = m - s_{\text{init}}$, and $\{\hat{x}_t\}_{t \geq t_{\text{init}}}$ be a sequence of actions so that

$$\hat{x}_t = \begin{cases} \text{FORWARD} & \text{if } t \leq t_{\text{init}} + k - 1 \\ \text{BACKWARD} & \text{otherwise} \end{cases}.$$

This sequence of actions are trivially feasible. For any $\tau \leq t_{\text{init}} + k - 1$, it is easy to see that

$$\sum_{t=t_{\text{init}}}^{\tau} f_t(s^\gamma(t), x^\gamma(t)) - f_t(\hat{s}_t, \hat{x}_t) \leq m - 1\,,$$

where $\{\hat{s}_t\}_{t \geq t_{\text{init}}}$ is a sequence of states defined in a similar with with Eq. (5). By the setting of the state transition function, we have $\hat{s}_{t_{\text{init}}+k} = m$. Let $t' = \operatorname*{argmin}_{t \geq t_{\text{init}}+k} x^\gamma(t) = \text{BACKWARD}$. Then

$$\sum_{t=t_{\text{init}}+k}^{t'} f_t(s^\gamma(t), x^\gamma(t)) - f_t(\hat{s}_t, \hat{x}_t) \leq 0\,,$$

and for every $t > t'$,

$$f_t(s^\gamma(t), x^\gamma(t)) = f_t(\hat{s}_t, \hat{x}_t)$$

because in such a case $s^\gamma(t) = \hat{s}_t$ and $x^\gamma(t) = \hat{x}_t$ always hold. Putting the three formulas above together, it is proved that this Dd-MDP instance is σ-chasable with $\sigma = m - 1$.

Let the number of rounds that an arbitrary algorithm performs the actions FORWARD and BACKWARD at the state m be k and k', respectively. Note that each time an algorithm performs BACKWARD at the state m, then it needs to take at least $m-1$ rounds to go back to the state m. It implies that $k + m \cdot k' \leq T$. The total reward obtained by this algorithm is

$$\frac{1}{2}k + k' \leq \frac{1}{2}k + \frac{1}{m}(T-k)\,. \tag{11}$$

Since the total reward by repeating a fixed policy γ_F that maps every state to FORWARD is at least $\frac{1}{2}(T-m)$, the policy regret is at least $(\frac{1}{2} - \frac{1}{m})(T-k) - \frac{m}{2}$. Therefore, if the policy regret is sublinear in T, we have $k = T - o(T)$. Now, consider another policy γ_B that maps every state to the action BACKWARD. We have

$$\sum_{t=1}^{T} f_t(s_t, \gamma'(s_t)) - \sum_{t=1}^{T} f_t(s_t, x_t) \;\geq\; \left(1 - \frac{1}{2}\right) \cdot k\,,$$

which implies that the external regret is linear in T.

Now consider an arbirary algorithm whose external regret is sublinear in T. Then, the total reward of this algorithm is at most $\frac{T}{m} + o(T)$, because otherwise it can still be inferred from Eq. (11) that k is linear in T, which leads to a linear external regret. Recall that the total reward of repeating the policy γ_F is $(T-m)/2$. Therefore, the policy regret is linear in T. $\square$

6 Bandit Setting

In Section 3, we investigate the online learning problem on Dd-MDPs under the full information setting, which means that for each round t, both the state transition function $g_t(s, x)$ and reward function $f_t(s, x)$ selected by the adversary are completely revealed to the decision maker after the decision maker chooses a (randomized) action x_t. In this part, we consider the *bandit* setting, where at each round t, the decision maker only knows the actual reward she receives, $f_t(s_t, x_t)$, with the state transition function $g_t(\cdot, \cdot)$. Obtaining vanishing regret for Dd-MDPs under the bandit setting requires a stronger condition than σ-chasability, which is defined in the following subsection.

6.1 Stateless Chasability

We say that an instance of Dd-MDP satisfies the *stateless chasability* condition for some parameter $\sigma > 0$ if there exists a chasing ongoing oracle $\mathcal{O}^{\texttt{Chasing}}$ which not only guarantees that $\texttt{CR} \leq \sigma$, but also ensures that for any target policy γ and any initial state t_{init}, the cumulative reward obtained by taking the generated actions $\{\hat{x}_t\}_{t \geq t_{\text{init}}}$ does not depend on the initial state s_{init}. More formally, let $s_{\text{init}}, s'_{\text{init}}$ be two arbitrary initial states, and $\{\hat{x}_t\}_{t \geq t_{\text{init}}}, \{\hat{x}'_t\}_{t \geq t_{\text{init}}}$ be two sequence of actions generated by the chasing ongoing oracle with starting from s_{init} and s'_{init}, respectively. The stateless chasability condition requires that for any $t_{\text{final}} \geq t_{\text{init}}$

$$\sum_{t \in [t_{\text{init}}, t_{\text{final}}]} \mathbb{E}\big[f_t(\hat{s}_t, \hat{x}_t)\big] = \sum_{t \in [t_{\text{init}}, t_{\text{final}}]} \mathbb{E}\big[f_t(\hat{s}'_t, \hat{x}'_t)\big] \,,$$

where $\hat{s}_t$ and $\hat{s}'_t$ are defined in a similar way with Eq. (5). A chasing ongoing oracle is said to be applicable to the bandit setting if its decision on each action $\hat{x}_t$ for $t \geq t_{\text{init}}$ only depends on s_{init}, $\{f_{t'}(\hat{s}_{t'}, \hat{x}_{t'})\}_{t' \in [t_{\text{init}}, t-1]}$ and $\{g_{t'}(\cdot, \cdot)\}_{t' \in [t_{\text{init}}, t-1]}$.

6.2 Multiarmed Bandit Problem

To develop vanishing-regret algorithms for σ-chasable Dd-MDPs under the bandit setting, we utilize technical tools that are related to the *Multiarmed Bandit Problem (MBP)* (Auer et al. [14]). Using a blackbox algorithm for this problem, Section 6.3 shows how to obtain vanishing regret for our problem.

In MBP, there is a set of arms Γ, and $\Psi \in \mathbb{N}$ rounds. At each round $\psi \in [\Psi]$, an adversary specifies a reward function $F_\psi : \Gamma \mapsto [0, 1]$, which is unknown to the online algorithm at the beginning of this round. Simultaneously, the algorithm chooses an action $\gamma_\psi \in \Gamma$. Then the reward $F_\psi(\gamma_\psi)$ obtained by the algorithm is revealed. The goal of the algorithm is to pick a sequence of actions $\gamma_1, \ldots, \gamma_\Psi$ in an online fashion to maximize $\mathbb{E}\Big[\sum_{\psi \in [\Psi]} F_t(\gamma_\psi)\Big]$. The regret is defined to be

$$\max_{\gamma \in \Gamma} \sum_{\psi \in [\Psi]} F_\psi(\gamma) - \sum_{\psi \in [\Psi]} \mathbb{E}\Big[F_\psi(\gamma_\psi)\Big] \,.$$

Theorem 10 (Audibert and Bubeck [13])**.** There exists an algorithm *Implicitly Normalized Forecaster* ($\texttt{INF}$) for MBP whose regret is bounded by $O\left(\sqrt{|\Gamma| \cdot \Psi}\right)$.

6.3 Decision Making Algorithm: Chasing & Switching in Fixed-Length Periods

We now present our decision making (DM) algorithm for the σ-chasable Dd-MDP problems under the bandit setting. Our DM algorithm *Chasing and Switching in Fixed-Length Periods* (C&S-FLP) requires blackbox accesses to a chasing ongoing oracle that is applicable to the bandit setting and Algorithm $\texttt{INF}$ for MBP, where the action set of MBP is set to be the collections of policies Γ in Dd-MDP. Algorithm $\texttt{INF}$ runs over consecutive periods of τ rounds for some $\tau > \sigma$, while the last period is allowed to have less than τ rounds. At the beginning of each period ψ, C&S-FLP invokes Algorithm $\texttt{INF}$ to choose a policy γ_ψ from Γ. Then it starts a new run of the chasing ongoing oracle $\mathcal{O}^{\texttt{Chasing}}$ with γ_ψ as the target policy, $s_{(\psi-1)\tau+1}$ as the initial state

s_{init} and $(\psi-1)\tau+1$ as the initial time t_{init}. Then C&S-FLP takes the sequence $\{\hat{x}_t\}_{t\in[(\psi-1)\tau+1,\min\{\psi\cdot\tau,T\}]}$ of actions generated by $\mathcal{O}^{\texttt{Chasing}}$ throughout the current period ψ, and send the reward $f_t(s_t,x_t)$ and state transition function $g_t(\cdot,\cdot)$ to $\mathcal{O}^{\texttt{Chasing}}$ after performing each $\hat{x}_t$ as the feedback. After the reward of the last step $t=\min\{T,\psi\tau\}$ of the current period is received, C&S-FLP computes $F_\psi(\gamma_\psi)=\frac{1}{\tau}\sum\limits_{t'=(\psi-1)\tau+1}^{t} f_{t'}(s_{t'},x_{t'})$ and feeds it to Algorithm `INF` as the reward of γ at t.

Theorem 11. The regret of C&S-FLP is bounded by $O\left(\frac{\sigma\cdot T}{\tau}+\sqrt{|\Gamma|T\tau}\right)$.

Proof. Let $\Psi=\left\lceil\frac{T}{\tau}\right\rceil$, and $R(\Psi,|\Gamma|)$ be the regret of Algorithm `INF`. With the stateless condition of the chasing ongoing oracle, $\{F_\psi\}_{\psi\in[\Psi]}$ is a sequence of stateless reward functions that satisfy the condition of Theorem 11. Therefore, we have

$$\max_{\gamma\in\Gamma}\sum_{\psi\in[\Psi]}F_\psi(\gamma)-\sum_{\psi\in[\Psi]}\mathbb{E}\Big[F_\psi(\gamma_\psi)\Big]\le R(\Psi,|\Gamma|)\,,$$

which gives that

$$\tau\cdot\max_{\gamma\in\Gamma}\sum_{\psi\in[\Psi]}F_\psi(\gamma)-\sum_{t\in[T]}\mathbb{E}\Big[f_t(s_t,x_t)\Big]\le\tau\cdot R(\Psi,|\Gamma|)\,.$$

By the definition of `CR`, for each period ψ we have

$$\sum_{t=(\psi-1)\tau+1}^{\psi\cdot\tau}f_t(s^\gamma(t),x^\gamma(t))-\tau\cdot\max_{\gamma\in\Gamma}\sum_{\psi\in[\Psi]}F_\psi(\gamma)\le\sigma\,.$$

Therefore, the regret of Algorithm C&S-FLP is bounded by

$$\sum_{t\in[T]}f_t(s^\gamma(t),x^\gamma(t))-\sum_{t\in[T]}\mathbb{E}\Big[f_t(s_t,x_t)\Big]\le\sigma\cdot\Psi+\tau\cdot R(\Psi,|\Gamma|)\,.$$

This proposition is proved by plugging Theorem 10 into the formula above. □

Corollary 5. By taking $\tau=T^{\frac{1}{3}}$, the regret of Algorithm C&S-FLP is bounded by $O\Big(\sigma T^{\frac{2}{3}}\sqrt{|\Gamma|}\Big)$.

7 Lower Bounds for Online Learning over Dd-MDPs with Chasability

In this part, we will prove lower bounds on the regret of online learning algorithms for σ-chasable Dd-MDP instances under the full-information setting and bandit setting, respectively.

Theorem 12. The regret of any online learning algorithm for 1-chasable Dd-MDP under full-information (resp., bandit) feedback is lower bounded by $\Omega(\sqrt{T\log|\Gamma|})$ (resp., $\Omega(|\Gamma|^{1/3}T^{2/3})$).

Proof. We first prove the full information lower bound, and the proof of the bandit version follows the same lines. Suppose the statement of the theorem does not hold. Then, there exists an online learning algorithm with regret of $o(\sqrt{T\log|\Gamma|})$ for any 1-chasable instance of the Dd-MDP problem. We show how to use this algorithm to design an OLSC algorithm with a unit switching cost whose regret is $o(\sqrt{T\log|\mathcal{X}|})$, where $\mathcal{X}$ is the action set of the OLSC instance. This is in contradiction to the known information theoretic $\Omega(\sqrt{T\log|\mathcal{X}|})$ lower bound on the regret of OLSC under the full-information setting, see Cesa-Bianchi and Lugosi [30], Freund and Schapire [51], Littlestone and Warmuth [66]. (For the bandit version of this proof we use the lower bound of Dekel et al. [39]).

Here is how the reduction works: Given an OLSC instance with a set $\mathcal{X}$ of actions and a unit switching cost, we construct a Dd-MDP instance with a state s^x for each action $x\in\mathcal{X}$. An arbitrary state $s\in S$ is selected to be the initial state s_1. Moreover, we set $\mathcal{X}_s = \mathcal{X}$ for every state s. For every $x \in \mathcal{X}$, we introduce a policy γ^x in the policy collection Γ of the Dd-MDP, defined so that it maps all states to action $x \in \mathcal{X}$. For each round t, when the adversary in OLSC specifies a reward function $F_t(\cdot)$, we construct the state transition function and reward function in the Dd-MDP by setting

$$g_t(s, x) = s^x \qquad \text{and} \qquad f_t(s, x) = \frac{1}{2} F_t(x) + \frac{1}{2} \cdot 1_{s=s^x} .$$

Obviously, this is a 1-chasable Dd-MDP instance. Moreover,

$$\max_{\gamma^x \in \Gamma} \sum_{t\in[T]} f_t(s^{\gamma^x}(t), x^{\gamma^x}(t)) - \sum_{t\in[T]} \mathbb{E}\Big[f_t(s_t, x_t)\Big] \le o(\sqrt{T \log |\Gamma|}) = o(\sqrt{T \log |\mathcal{X}|}) ,$$

where the inequality is due to the assumed regret bound. The construction of the Dd-MDP instance ensures that for every $\gamma \in \Gamma$,

$$\sum_{t\in[T]} f_t(s^{\gamma^x}(t), x^{\gamma^x}(t)) \ge \frac{1}{2} \cdot \sum_{t\in[T]} F_t(x) + \frac{1}{2}(T-1) ,$$

$$\sum_{t\in[T]} f_t(s_t, x_t) \le \frac{1}{2} \sum_{t\in[T]} F_t(x_t) + \frac{1}{2} T - \frac{1}{2} \sum_{t\in[2,T]} 1_{x_t \neq x_{t-1}}$$

Putting these pieces together, we get

$$\frac{1}{2}\left(\max_{x\in\mathcal{X}} \sum_{t\in[T]} F_t(x) - \sum_{t\in[T]} \mathbb{E}\Big[F_t(x_t)\Big] + \sum_{t\in[2,T]} 1_{x_t \neq x_{t-1}} \right) - 1 \le o\Big(\sqrt{T \log |\mathcal{X}|}\Big) ,$$

and therefore the regret of the OLSC instance is bounded by $o(\sqrt{T \log |\mathcal{X}|})$, a contradiction. ☐

Theorem 13. For an Dd-MDP instance $\mathcal{I}$, let $\sigma \ge 1$ be the minimum value such that $\mathcal{I}$ is σ-chasable. The regret of any online learning algorithm over $\mathcal{I}$ under full-information (resp. bandit) feedback is bounded from below by $\Omega\big(\sqrt{\sigma T}\big)$ (resp. $\Omega\big(\sigma^{\frac{1}{3}} T^{\frac{2}{3}}\big)$).

Proof. For the full information setting, recall the problem instance of DRACC that we construct in the proof of Theorem 8. An observation here is that there exists a trivial chasing oracle for this instance which just waits for the W new resources arriving at the beginning of the next episode. The chasing regret of such a chasing oracle is bounded by $2CW$, which means that Dd-MDP instance corresponding to this DRACC problem instance is σ-chasable with $\sigma \le 2CW$. Suppose that the current theorem does not hold, then there exists an online algothm with an $o(\sqrt{CW \cdot T})$-regret for the DRACC problem instance constructed in the proof of Theorem 8, a contradiction.

For the bandit setting, we make a reduction from the problem of *Online Pricing for Patient Buyers* (OPPB) investigated in Feldman et al. [50]. In OPPB, the algorithm is required to decide a price $p_t \in [0, 1]$ for every round $t \in [W]$ for some time window $W \in \mathbb{Z}_{\ge 1}$ at the beginning of the first round $t = 1$, and specifies a price $p_{t'} \in [0, 1]$ for $t' = t + W$ at the beginning of every round $t > 1$. The user arriving at each round t pays for the lowest price in $[t, W_t]$ if it does not exceed her own value $v_t \in [0, 1]$. Here $W_t \in [1, W]$ is a parameter associated with each user t. The algorithm for OPBB is only allowed to observe the payment of each user, but cannot obtain the valuation v_t of the users. By taking the price vectors over W rounds as states, one can easily transform an instance of OPPB to a Dd-MDP instance $\mathcal{I}$ with bandit feedback. Since there exists a chasing oracle for $\mathcal{I}$ whose chasing regret is bounded by W, we know that $\mathcal{I}$ is σ-chasable with $\sigma \le W$. If the current theorem is not true, then there exists an online learning algorithm for OPPB with an $o(W^{\frac{1}{3}} T^{\frac{2}{3}})$-regret, which conflicts with the known lower bound (Feldman et al. [50, Theorem 2]). ☐

8 Conclusion and Open Problems

We study the problem of stateful dynamic pricing in the context of resource allocation problems such as scheduling. Aiming towards obtaining vanishing regret with respect to the best fixed pricing policy in hindsight, we reduced this problem to the problem of obtaining vanishing regret in the setting of Dd-MDPs, where access to a chasing oracle is granted. By exploiting the power of chasability, we have managed to reduce the latter problem to the stateless problem of online learning with switching cost, thus obtaining an optimal $O(\sqrt{T})$ regret bound for this latter problem under chasability. For the bandit feedback version of the same problem, by introducing a stateless version of the chasability condition and reducing the problem to multi-armed bandits with switching cost, we were able to obtain an $O(T^{2/3})$ regret bound. We also established nearly matching lower-bounds for the Dd-MDP regret with chasability under both full information and bandit feedback. We complemented our results by designing chasing oracles for the DRACC problem under both general and k_t-demand valuations, resulting in dynamic pricing algorithms for these valuation classes with regret bounds of $O(T^{\frac{1}{2}(1+CW/(CW+1))})$ and $O(T^{3/4})$, respectively.

An immediate research question arising from our work concerns the design of low-regret chasing oracles for other valuation classes beyond those studied in the current paper. Also, while our regret bounds for the Dd-MDP problem with chasability are nearly optimal, one may hope to improve our regret bounds for the DRACC problem, a task that we leave for future work. Finally, the two frameworks introduced in this paper, namely, the Dd-MDP with chasability and DRACC, are rather general; exploring other applications that fit into either of these frameworks is yet another interesting research question.

Acknowledgments

An extended abstract of this paper appeared in the Proceedings of the 34th Conference on Neural Information Processing Systems.

The work of Yuval Emek was supported in part by an Israel Science Foundation grant number 1016/17. The work of Ron Lavi was partially supported by the Israel Science Foundation - Natural Science Foundation of China joint research program (grant No. 2560/17). The work of Rad Niazadeh was supported by the Chicago Booth School of Business. The work of Yangguang Shi was partially supported at the Technion by a fellowship of the Israel Council for Higher Education, and at Shandong University by the Science Fund Program of Shandong Province for Distinguished Oversea Young Scholars (grant 2023HWYQ-006).

References

[1] Abbasi-Yadkori Y, Bartlett PL, Kanade V, Seldin Y, Szepesvári C (2013) Online learning in markov decision processes with adversarially chosen transition probability distributions. *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, 2508–2516 (Lake Tahoe, Nevada, United States).

[2] Abolhassani M, Ehsani S, Esfandiari H, HajiAghayi M, Kleinberg R, Lucier B (2017) Beating 1-1/e for ordered prophets. *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 61–71 (ACM).

[3] Agrawal S, Daskalakis C, Mirrokni V, Sivan B (2018) Robust repeated auctions under heterogeneous buyer behavior. *19th ACM conference on Economics and Computation.*

[4] Agrawal S, Devanur NR (2015) Fast algorithms for online stochastic convex programming. *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '15*, 1405–1424 (San Diego, CA, USA: SIAM).

[5] Agrawal S, Devanur NR (2019) Bandits with global convex constraints and objective. *Operations Research* 67(5):1486–1502.

[6] Alaei S (2014) Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing* 43(2):930–972.

[7] Amin K, Rostamizadeh A, Syed U (2013) Learning prices for repeated auctions with strategic buyers. *Advances in Neural Information Processing Systems*, 1169–1177.

[8] Amin K, Rostamizadeh A, Syed U (2014) Repeated contextual auctions with strategic buyers. *Advances in Neural Information Processing Systems*, 622–630.

[9] Anari N, Niazadeh R, Saberi A, Shameli A (2019) Nearly optimal pricing algorithms for production constrained and laminar bayesian selection. *Proceedings of the 2019 ACM Conference on Economics and Computation*, 91–92.

[10] Araman VF, Caldentey R (2009) Dynamic pricing for nonperishable products with demand learning. *Operations research* 57(5):1169–1188.

[11] Arora R, Dekel O, Tewari A (2012) Online bandit learning against an adaptive adversary: from regret to policy regret. *Proceedings of the 29th International Conference on Machine Learning, ICML '12* (Edinburgh, Scotland, UK: icml.cc / Omnipress).

[12] Arora R, Dinitz M, Marinov TV, Mohri M (2018) Policy regret in repeated games. *Annual Conference on Neural Information Processing Systems 2018, NeurIPS '18*, 6733–6742 (Montréal, Canada).

[13] Audibert J, Bubeck S (2009) Minimax policies for adversarial and stochastic bandits. *COLT '09 - The 22nd Conference on Learning Theory* (Montreal, Quebec, Canada).

[14] Auer P, Cesa-Bianchi N, Freund Y, Schapire RE (2002) The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing* 32(1):48–77.

[15] Azar PD, Kleinberg R, Weinberg SM (2014) Prophet inequalities with limited information. *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, 1358–1377 (Society for Industrial and Applied Mathematics).

[16] Azar Y, Chiplunkar A, Kaplan H (2018) Prophet secretary: Surpassing the 1-1/e barrier. *Proceedings of the 2018 ACM Conference on Economics and Computation*, 303–318 (ACM).

[17] Babaioff M, Dughmi S, Kleinberg RD, Slivkins A (2015) Dynamic pricing with limited supply. *ACM Transactions on Economics and Computation* 3(1):4:1–4:26.

[18] Babaioff M, Immorlica N, Lucier B, Weinberg SM (2015) A simple and approximately optimal mechanism for an additive buyer. *ACM SIGecom Exchanges* 13(2):31–35.

[19] Badanidiyuru A, Kleinberg R, Slivkins A (2018) Bandits with knapsacks. *Journal of the ACM* 65(3):13:1–13:55.

[20] Besbes O, Zeevi A (2009) Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research* 57(6):1407–1420.

[21] Beyhaghi H, Golrezaei N, Leme RP, Pal M, Siva B (2018) Improved approximations for free-order prophets and second-price auctions. *arXiv preprint arXiv:1807.03435* .

[22] Blum A, Hartline JD (2005) Near-optimal online auctions. *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '05*, 1156–1163 (Vancouver, British Columbia, Canada: SIAM).

[23] Blum A, Kumar V, Rudra A, Wu F (2004) Online learning in online auctions. *Theoretical Computer Science* 324(2-3):137–146.

[24] Brantley K, Dudík M, Lykouris T, Miryoosefi S, Simchowitz M, Slivkins A, Sun W (2020) Constrained episodic reinforcement learning in concave-convex and knapsack settings. *CoRR* abs/2006.05051, accepted by NeurIPS '20.

[25] Broder J, Rusmevichientong P (2012) Dynamic pricing under a general parametric choice model. *Operations Research* 60(4):965–980.

[26] Bubeck S, Devanur NR, Huang Z, Niazadeh R (2017) Online auctions and multi-scale online learning. *Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17*, 497–514 (Cambridge, MA, USA: ACM).

[27] Bubeck S, Devanur NR, Huang Z, Niazadeh R (2019) Multi-scale online learning: Theory and applications to online auctions and pricing. *Journal of Machine Learning Research* 20:62:1–62:37.

[28] Cai Y, Daskalakis C, Weinberg SM (2012) Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization. *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, 130–139 (IEEE).

[29] Cai Y, Devanur NR, Weinberg SM (2016) A duality based unified approach to bayesian mechanism design. *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, 926–939 (ACM).

[30] Cesa-Bianchi N, Lugosi G (2006) *Prediction, learning, and games* (Cambridge University Press), ISBN 978-0-521-84108-5.

[31] Chawla S, Devanur NR, Holroyd AE, Karlin AR, Martin JB, Sivan B (2017) Stability of service under time-of-use pricing. *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC '17*, 184–197 (Montreal, QC, Canada: ACM).

[32] Chawla S, Devanur NR, Kulkarni J, Niazadeh R (2017) Truth and regret in online scheduling. *Proceedings of the 2017 ACM Conference on Economics and Computation, EC '17*, 423–440 (Cambridge, MA, USA: ACM).

[33] Chawla S, Hartline JD, Malec DL, Sivan B (2010) Multi-parameter mechanism design and sequential posted pricing. *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC '10*, 311–320 (Cambridge, Massachusetts, USA: ACM).

[34] Chawla S, Miller JB (2016) Mechanism design for subadditive agents via an ex ante relaxation. *Proceedings of the 2016 ACM Conference on Economics and Computation*, 579–596 (ACM).

[35] Correa J, Foncea P, Hoeksma R, Oosterwijk T, Vredeveld T (2017) Posted price mechanisms for a random stream of customers. *Proceedings of the 2017 ACM Conference on Economics and Computation*, 169–186 (ACM).

[36] Correa J, Foncea P, Hoeksma R, Oosterwijk T, Vredeveld T (2019) Recent developments in prophet inequalities. *ACM SIGecom Exchanges* 17(1):61–70.

[37] Correa J, Saona R, Ziliotto B (2018) Prophet secretary through blind strategies. *arXiv preprint arXiv:1807.07483* .

[38] Correa J, Saona R, Ziliotto B (2020) Prophet secretary through blind strategies. *Mathematical Programming* 1–39.

[39] Dekel O, Ding J, Koren T, Peres Y (2014) Bandits with switching costs: $T^{2/3}$ regret. *Symposium on Theory of Computing, STOC '14*, 459–467 (New York, NY, USA: ACM).

[40] Dekel O, Hazan E (2013) Better rates for any adversarial deterministic MDP. *Proceedings of the 30th International Conference on Machine Learning, ICML '13*, volume 28 of *JMLR Workshop and Conference Proceedings*, 675–683 (Atlanta, GA, USA: JMLR.org).

[41] den Boer AV (2015) Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in operations research and management science* 20(1):1–18.

[42] Devanur NR, Jain K, Sivan B, Wilkens CA (2019) Near optimal online algorithms and fast approximation algorithms for resource allocation problems. *Journal of the ACM* 66(1):7:1–7:41.

[43] Düetting P, Feldman M, Kesselheim T, Lucier B (2017) Prophet inequalities made easy: Stochastic optimization by pricing non-stochastic inputs. *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, 540–551 (IEEE).

[44] Dütting P, Fischer F, Klimm M (2016) Revenue gaps for discriminatory and anonymous sequential posted pricing. *arXiv preprint arXiv:1607.07105* .

[45] Esfandiari H, Hajiaghayi M, Liaghat V, Monemizadeh M (2017) Prophet secretary. *SIAM Journal on Discrete Mathematics* 31(3):1685–1701.

[46] Even-Dar E, Kakade SM, Mansour Y (2004) Experts in a markov decision process. *Advances in Neural Information Processing Systems 17, NIPS '04*, 401–408 (Vancouver, British Columbia, Canada).

[47] Feige U, Koren T, Tennenholtz M (2017) Chasing ghosts: Competing with stateful policies. *SIAM Journal on Computing* 46(1):190–223.

[48] Feldman M, Fu H, Gravin N, Lucier B (2013) Simultaneous auctions are (almost) efficient. *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, 201–210 (ACM).

[49] Feldman M, Gravin N, Lucier B (2015) Combinatorial auctions via posted prices. *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '15*, 123–135 (San Diego, CA, USA: SIAM).

[50] Feldman M, Koren T, Livni R, Mansour Y, Zohar A (2016) Online pricing with strategic and patient buyers. *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems, NIPS '16*, 3864–3872 (Barcelona, Spain).

[51] Freund Y, Schapire RE (1995) A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory, Second European Conference, EuroCOLT '95*, volume 904 of *Lecture Notes in Computer Science*, 23–37 (Barcelona, Spain: Springer).

[52] Göbel O, Hoefer M, Kesselheim T, Schleiden T, Vöcking B (2014) Online independent set beyond the worst-case: Secretaries, prophets, and periods. *International Colloquium on Automata, Languages, and Programming*, 508–519 (Springer).

[53] Guan P, Raginsky M, Willett R (2014) From minimax value to low-regret algorithms for online markov decision processes. *American Control Conference, ACC '14*, 471–476 (Portland, OR, USA: IEEE).

[54] Hajiaghayi MT, Kleinberg R, Sandholm T (2007) Automated online mechanism design and prophet inequalities. *AAAI*, volume 7, 58–65.

[55] Harrison JM, Keskin NB, Zeevi A (2012) Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution. *Management Science* 58(3):570–586.

[56] Immorlica N, Sankararaman KA, Schapire RE, Slivkins A (2019) Adversarial bandits with knapsacks. *60th IEEE Annual Symposium on Foundations of Computer Science*, 202–219, FOCS '19 (Baltimore, Maryland, USA: IEEE Computer Society).

[57] Kalai AT, Vempala SS (2005) Efficient algorithms for online decision problems. *Journal of Computer and System Sciences* 71(3):291–307.

[58] Kanoria Y, Nazerzadeh H (2014) Dynamic reserve prices for repeated auctions: Learning from bids. *International Conference on Web and Internet Economics*, 232–232 (Springer).

[59] Kesselheim T, Radke K, Tönnis A, Vöcking B (2014) Primal beats dual on online packing lps in the random-order model. *Symposium on Theory of Computing, STOC '14*, 303–312 (New York, NY, USA: ACM).

[60] Kleinberg R, Weinberg SM (2012) Matroid prophet inequalities. *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, 123–136 (ACM).

[61] Kleinberg RD, Leighton FT (2003) The value of knowing a demand curve: Bounds on regret for online posted-price auctions. *Symposium on Foundations of Computer Science, FOCS '03*, 594–605 (Cambridge, MA, USA: IEEE Computer Society).

[62] Lee E, Singla S (2018) Optimal online contention resolution schemes via ex-ante prophet inequalities. *arXiv preprint arXiv:1806.09251* .

[63] Lehmann B, Lehmann D, Nisan N (2006) Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior* 55(2):270–296.

[64] Lehmann D, O'Callaghan L, Shoham Y (2002) Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM* 49(5):577–602.

[65] Leme RP, Schneider J (2018) Contextual search via intrinsic volumes. *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, 268–282 (IEEE).

[66] Littlestone N, Warmuth MK (1989) The weighted majority algorithm. *30th Annual Symposium on Foundations of Computer Science, FOCS '89*, 256–261 (Research Triangle Park, North Carolina, USA: IEEE Computer Society).

[67] Lucier B (2017) An economic view of prophet inequalities. *ACM SIGecom Exchanges* 16(1):24–47.

[68] Lykouris T, Simchowitz M, Slivkins A, Sun W (2019) Corruption robust exploration in episodic reinforcement learning. *CoRR* abs/1911.08689.

[69] Mohri M, Munoz A (2014) Optimal regret minimization in posted-price auctions with strategic buyers. *Advances in Neural Information Processing Systems*, 1871–1879.

[70] Neu G, György A, Szepesvári C, Antos A (2014) Online markov decision processes under bandit feedback. *IEEE Transactions on Automatic Control* 59(3):676–691.

[71] Niazadeh R, Saberi A, Shameli A (2018) Prophet inequalities vs. approximating optimum online. *International Conference on Web and Internet Economics*, 356–374 (Springer).

[72] Rubinstein A (2016) Beyond matroids: Secretary problem and prophet inequality with general constraints. *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, 324–332 (ACM).

[73] Samuel-Cahn E (1984) Comparison of threshold stop rules and maximum for independent nonnegative random variables. *the Annals of Probability* 12(4):1213–1216.

[74] Yan Q (2011) Mechanism design via correlation gap. *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, 710–719 (Society for Industrial and Applied Mathematics).

[75] Yao AC (1977) Probabilistic computations: Toward a unified measure of complexity. *18th Annual Symposium on Foundations of Computer Science, FOCS '77*, 222–227 (Providence, Rhode Island, USA: IEEE Computer Society).

[76] Yu JY, Mannor S, Shimkin N (2009) Markov decision processes with arbitrary reward processes. *Mathematics of Operations Research* 34(3):737–757.

[77] Zhang X, Wu C, Huang Z, Li Z (2018) Occupation-oblivious pricing of cloud jobs via online learning. *2018 IEEE Conference on Computer Communications, INFOCOM '18*, 2456–2464 (Honolulu, HI, USA: IEEE).