Fully Dynamic Online Selection through Online Contention Resolution Schemes

Vashist Avadhanula[1], Andrea Celli[2], Riccardo Colini-Baldeschi[1], Stefano Leonardi[3], Matteo Russo[3]

[1] Core Data Science, Meta, London, UK
[2]Department of Computing Sciences, Bocconi University, Milan, Italy
[3]Department of Computer, Control and Management Engineering Antonio Ruberti, Sapienza University, Rome, Italy
vas1089@gmail.com, andrea.celli2@unibocconi.it, rickuz@fb.com, leonardi@diag.uniroma1.it, mrusso@diag.uniroma1.it

Abstract

We study *fully dynamic* online selection problems in an adversarial/stochastic setting that includes Bayesian online selection, prophet inequalities, posted price mechanisms, and stochastic probing problems subject to combinatorial constraints. In the classical "incremental" version of the problem, selected elements remain active until the end of the input sequence. On the other hand, in the *fully dynamic* version of the problem, elements stay active for a limited time interval, and then leave. This models, for example, the online matching of tasks to workers with task/worker-dependent working times, and sequential posted pricing of perishable goods. A successful approach to online selection problems in the adversarial setting is given by the notion of *Online Contention Resolution Scheme* (OCRS), that uses *a priori* information to formulate a linear relaxation of the underlying optimization problem, whose optimal fractional solution is rounded online for any adversarial order of the input sequence. Our main contribution is providing a general method for constructing an OCRS for fully dynamic online selection problems. Then, we show how to employ such OCRS to construct no-regret algorithms in a partial information model with *semi-bandit* feedback and adversarial inputs.

Introduction

Consider the case where a financial service provider receives multiple operations every hour/day. These operations might be malicious. The provider needs to assign them to *human reviewers* for inspection. The time required by each reviewer to file a reviewing task and the reward (weight) that is obtained with the review follow some distributions. The distributions can be estimated from historical data, as they depend on the type of transaction that needs to be examined and on the expertise of the employed reviewers. To efficiently solve the problem, the platform needs to compute a matching between tasks and reviewers based on the a priori information that is available. However, the time needed for a specific review, and the realized reward (weight), is often known only after the task/reviewer matching is decided.

A multitude of variations to this setting are possible. For instance, if a cost is associated with each reviewing task, the total cost for the reviewing process might be bounded by a budget. Moreover, there might be various kinds of restrictions on the subset of reviewers that are assigned at each time step. Finally, the objective function might not only be the sum of the rewards (weights) we observe, if, for example, the decision maker has a utility function with "diminishing return" property.

To model the general class of sequential decision problems described above, we introduce *fully dynamic online selection problems*. This model generalizes online selection problems (Chekuri, Vondrák, and Zenklusen 2011), where elements arrive online in an adversarial order and algorithms can use a priori information to maximize the weight of the selected subset of elements, subject to combinatorial constraints (such as matroid, matching, or knapsack).

In the classical version of the problem (Chekuri, Vondrák, and Zenklusen 2011), once an element is selected, it will affect the combinatorial constraints throughout the entire input sequence. This is in sharp contrast with the fully dynamic version, where an element will affect the combinatorial constraint only for a limited time interval, which we name *activity time* of the element. For example, a new task can be matched to a reviewer as soon as she is done with previously assigned tasks, or an agent can buy a new good as soon as the previously bought goods are perished. A large class of Bayesian online selection (Kleinberg and Weinberg 2012), prophet inequality (Hajiaghayi, Kleinberg, and Sandholm 2007), posted price mechanism (Chawla et al. 2010), and stochastic probing (Gupta and Nagarajan 2013) problems that have been studied in the classical version of online selection can therefore be extended to the fully dynamic setting. Note that in the dynamic algorithms literature, *fully dynamic* algorithms are algorithms that deal with both adversarial insertions and deletions (Demetrescu et al. 2010). We could also interpret our model in a similar sense since elements arrive online (are inserted) according to an adversarial order, and cease to exist (are deleted) according to adversarially established activity times.

A successful approach to online selection problems is based on Online Contention Resolution Schemes (OCRSs) (Feldman, Svensson, and Zenklusen 2016). OCRSs use a priori information on the values of the elements to formulate a linear relaxation whose optimal fractional solution upper bounds the performance of the integral offline optimum. Then, an online rounding procedure is used to produce a solution whose value is as close as possible to the fractional relaxation solution's value, for any adversarial order of the input sequence. The OCRS approach allows to obtain good approximations of the expected optimal solution for linear and submodular objective functions. The existence of OCRSs for fully dynamic online selection problems is therefore a natural research question that we address in this work.

The OCRS approach is based on the availability of a priori information on weights and activity times. However, in real world scenarios, these might be missing or might be expensive to collect. Therefore, in the second part of our work, we study the fully dynamic online selection problem with partial information, where the main research question is whether the OCRS approach is still viable if a priori information on the weights is missing. In order to answer this question, we study a repeated version of the fully dynamic online selection problem, in which at each stage weights are unknown to the decision maker (i.e., no a priori information on weights is available) and chosen adversarially. The goal in this setting is the design of an online algorithm with performances (i.e., cumulative sum of weights of selected elements) close to that of the best static selection strategy in hindsight.

Our Contributions

First, we introduce the *fully dynamic online selection problem*, in which elements arrive following an adversarial ordering, and revealed one-by-one their weights and activity times at the time of arrival (i.e., *prophet model*), or after the element has been selected (i.e., *probing model*). Our model describes temporal packing constraints (i.e., downward-closed), where elements are active only within their activity time interval. The objective is to maximize the weight of the selected set of elements subject to temporal packing constraints. We provide two black-box reductions for adapting classical OCRS for online (non-dynamic) selection problems to the fully dynamic setting under full and partial information.

Blackbox reduction 1: from OCRS to temporal OCRS. Starting from a (b, c)-selectable greedy OCRS in the classical setting, we use it as a subroutine to build a (b, c)-selectable greedy OCRS in the more general temporal setting (see Algorithm 1 and Theorem 1). This means that competitive ratio guarantees in one setting determine the same guarantees in the other. Such a reduction implies the existence of algorithms with constant competitive ratio for online optimization problems with linear or submodular objective functions subject to matroid, matching, and knapsack constraints, for which we give explicit constructions. We also extend the framework to elements arriving in batches, which can have correlated weights or activity times within the batch, as described in the appendix of the full version of the paper.

Blackbox reduction 2: from temporal OCRS to no-α-regret algorithm. Following the recent work by Gergatsouli and Tzamos (2022) in the context of Pandora's box problems, we define the following extension of the problem to the partial-information setting. For each of the T stages, the algorithm is given in input a new instance of the fully dynamic online selection problem. Activity times are fixed beforehand and known to the algorithm, while weights are chosen by an adversary, and revealed only after the selection at the current stage has been completed. In such setting, we show that an α-competitive temporal OCRS can be exploited in the adversarial partial-information version of the problem, in order to build no-α-regret algorithms with polynomial per-iteration running time. Regret is measured with respect to the cumulative weights collected by the best fixed selection policy in hindsight. We study three different settings: in the first setting, we study the *full-feedback model* (i.e., the algorithm observes the entire utility function at the end of each stage). Then, we focus on the *semi-bandit-feedback model*, in which the algorithm only receives information on the weights of the elements it selects. In such setting, we provide a no-α-regret framework with $\tilde{O}(T^{1/2})$ upper bound on cumulative regret in the case in which we have a "white-box" OCRS (i.e., we know the exact procedure run within the OCRS, and we are able to simulate it ex-post). Moreover, we also provide a no-α-regret algorithm with $\tilde{O}(T^{2/3})$ regret upper bound for the case in which we only have oracle access to the OCRS (i.e., the OCRS is treated as a black-box, and the algorithm does not require knowledge about its internal procedures).

Related Work

In the first part of the paper, we deal with a setting where the algorithm has complete information over the input but is unaware of the order in which elements arrive. In this context, Contention resolution schemes (CRS) were introduced by Chekuri, Vondrák, and Zenklusen (2011) as a powerful rounding technique in the context of submodular maximization. The CRS framework was extended to online contention resolution schemes (OCRS) for online selection problems by Feldman, Svensson, and Zenklusen (2016), who provided constant competitive OCRSs for different problems, e.g. intersections of matroids, matchings, and prophet inequalities. We generalize the OCRS framework to a setting where elements are timed and cease to exist right after.

In the second part, we lift the complete knowledge assumption and work in an adversarial bandit setting, where at each stage the entire set of elements arrives, and we seek to select the "best" feasible subset. This is similar to the problem of *combinatorial bandits* (Cesa-Bianchi and Lugosi 2012), but unlike it, we aim to deal with combinatorial selection of *timed* elements. In this respect, *blocking bandits* (Basu et al. 2019) model situations where played arms are blocked for a specific number of stages. Despite their contextual (Basu et al. 2021), combinatorial (Atsidakou et al. 2021), and adversarial (Bishop et al. 2020) extensions, recent work on blocking bandits only addresses specific cases of the fully dynamic online selection problem (Dickerson et al. 2018), which we solve in entire generality, i.e. adversarially and for all packing constraints.

Our problem is also related to *sleeping bandits* (Kleinberg, Niculescu-Mizil, and Sharma 2010), in that the adversary decides which actions the algorithm can perform at each stage t. Nonetheless, a sleeping bandit adversary has to communicate all available actions to the algorithm before a stage starts, whereas our adversary sets arbitrary activity times for each element, choosing in what order elements arrive.

Preliminaries

Given a finite set $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq 2^{\mathcal{X}}$, let $\mathbf{1}_{\mathcal{Y}} \in \{0,1\}^{|X|}$ be the characteristic vector of set $\mathcal{X}$, and $\operatorname{co}\mathcal{X}$ be the convex hull of $\mathcal{X}$. We denote vectors by bold fonts. Given vector $\boldsymbol{x}$, we denote by x_i its i-th component. The set $\{1, 2, \ldots, n\}$, with $n \in \mathbb{N}_{>0}$, is compactly denoted as $[n]$. Given a set $\mathcal{X}$ and a scalar $\alpha \in \mathbb{R}$, let $\alpha\mathcal{X} := \{\alpha x : x \in \mathcal{X}\}$. Finally, given a discrete set $\mathcal{X}$, we denote by $\Delta^{\mathcal{X}}$ the $|\mathcal{X}|$-simplex.

We start by introducing a general *selection problem* in the standard (i.e., non-dynamic) case as studied by Kleinberg and Weinberg (2012) in the context of prophet inequalities. Let $\mathcal{E}$ be the ground set and let $m := |\mathcal{E}|$. Each element $e \in \mathcal{E}$ is characterized by a collection of parameters z_e. In general, z_e is a random variable drawn according to an element-specific distribution ζ_e, supported over the joint set of possible parameters. In the standard (i.e., non-dynamic) setting, z_e just encodes the *weight* associated to element e, that is $z_e = (w_e)$, for some $w_e \in [0,1]$.[1] In such case distributions ζ_e are supported over $[0,1]$. Random variables $\{z_e : e \in \mathcal{E}\}$ are independent, and z_e is distributed according to ζ_e. An *input sequence* is an ordered sequence of elements and weights such that every element in $\mathcal{E}$ occurs exactly once in the sequence. The order is specified by an arrival time s_e for each element e. Arrival times are such that $s_e \in [m]$ for all $e \in \mathcal{E}$, and for two distinct e, e' we have $s_e \neq s_{e'}$. The order of arrival of the elements is a priori unknown to the algorithm, and can be selected by an adversary. In the standard full-information setting the distributions ζ_e can be chosen by an adversary, but they are known to the algorithm a priori. We consider problems characterized by a family of *packing constraints*.

Definition 1 (Packing Constraint). *A family of constraints $\mathcal{F} = (\mathcal{E}, \mathcal{I})$, for ground set $\mathcal{E}$ and independence family $\mathcal{I} \subseteq 2^{\mathcal{E}}$, is said to be* packing *(i.e.,* downward-closed*) if, taken $A \in \mathcal{I}$, and $B \subseteq A$, then $B \in \mathcal{I}$.*

Elements of $\mathcal{I}$ are called *independent sets*. Such family of constraints is closed under intersection, and encompasses matroid, knapsack, and matching constraints.

Fractional LP formulation Even in the offline setting, in which the ordering of the input sequence $(s_e)_{e\in\mathcal{E}}$ is known beforehand, determining an independent set of maximum cumulative weight may be NP-hard in the worst-case (Feige 1998). Then, we consider the relaxation of the problem in which we look for an optimal *fractional solution*. The value of such solution is an upper bound to the value of the true offline optimum. Therefore, any algorithm guaranteeing a constant approximation to the offline fractional optimum immediately yields the same guarantees with respect to the offline optimum. Given a family of packing constraints $\mathcal{F} = (\mathcal{E}, \mathcal{I})$, in order to formulate the problem of computing the best fractional solution as a linear programming problem (LP) we introduce the notion of *packing constraint polytope* $\mathcal{P}_{\mathcal{F}} \subseteq [0,1]^m$ which is such that $\mathcal{P}_{\mathcal{F}} := \operatorname{co}(\{\mathbf{1}_S : S \in \mathcal{I}\})$. Given a non-negative submodular function $f : [0,1]^m \to \mathbb{R}_{\geq 0}$, and a family of packing constraints $\mathcal{F}$, an optimal *fractional* solution can be computed via the LP $\max_{\boldsymbol{x}\in\mathcal{P}_{\mathcal{F}}} f(\boldsymbol{x})$. If the goal is maximizing the cumulative sum of weights, the objective of the optimization problem is $\langle \boldsymbol{x}, \boldsymbol{w} \rangle$, where $\boldsymbol{w} := (w_1, \ldots, w_m) \in [0,1]^m$ is a vector specifying the weight of each element. If we assume access to a polynomial-time separation oracle for $\mathcal{P}_{\mathcal{F}}$ such LP yields an optimal fractional solution in polynomial time.

Online selection problem. In the online version of the problem, given a family of packing constraints $\mathcal{F}$, the goal is selecting an independent set whose cumulative weight is as large as possible. In such setting, the elements reveal one by one their realized z_e, following a fixed prespecified order unknown to the algorithm. Each time an element reveals z_e, the algorithm has to choose whether to select it or discard it, before the next element is revealed. Such decision is irrevocable. Computing the exact optimal solution to such online selection problems is intractable in general (Feige 1998), and the goal is usually to design approximation algorithms with good *competitive ratio*.[2] In the remainder of the section we describe one well-known framework for such objective.

Online contention resolution schemes. *Contention resolution schemes* were originally proposed by Chekuri, Vondrák, and Zenklusen (2011) in the context of submodular function maximization, and later extended to online selection problems by Feldman, Svensson, and Zenklusen (2016) under the name of *online contention resolution schemes* (OCRS). Given a fractional solution $\boldsymbol{x} \in \mathcal{P}_{\mathcal{F}}$, an OCRS is an online rounding procedure yielding an independent set in $\mathcal{I}$ guaranteeing a value close to that of $\boldsymbol{x}$. Let $R(\boldsymbol{x})$ be a random set containing each element e independently and with probability x_e. The set $R(\boldsymbol{x})$ may not be feasible according to constraints $\mathcal{F}$. An OCRS essentially provides a procedure to construct a good feasible approximation by starting from the random set $R(\boldsymbol{x})$. Formally,

Definition 2 (OCRS). *Given a point $\boldsymbol{x} \in \mathcal{P}_{\mathcal{F}}$ and the set of elements $R(\boldsymbol{x})$, elements $e \in \mathcal{E}$ reveal one by one whether they belong to $R(\boldsymbol{x})$ or not. An OCRS chooses irrevocably whether to select an element in $R(\boldsymbol{x})$ before the next element is revealed. An OCRS for $\mathcal{P}_{\mathcal{F}}$ is an online algorithm that selects $S \subseteq R(\boldsymbol{x})$ such that $\mathbf{1}_S \in \mathcal{P}_{\mathcal{F}}$.*

We will focus on *greedy OCRS*, which were defined by Feldman, Svensson, and Zenklusen (2016) as follows.

Definition 3 (Greedy OCRS). *Let $\mathcal{P}_{\mathcal{F}} \subseteq [0,1]^m$ be the feasibility polytope for constraint family $\mathcal{F}$. An OCRS π for $\mathcal{P}_{\mathcal{F}}$ is called a greedy OCRS if, for every ex-ante feasible solution $\boldsymbol{x} \in \mathcal{P}_{\mathcal{F}}$, it defines a packing subfamily of feasible sets $\mathcal{F}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}$, and an element e is selected upon arrival if, together with the set of already selected elements, the resulting set is in $\mathcal{F}_{\pi,\boldsymbol{x}}$.*

A greedy OCRS is *randomized* if, given $\boldsymbol{x}$, the choice of $\mathcal{F}_{\pi,\boldsymbol{x}}$ is randomized, and *deterministic* otherwise. For $b, c \in$

[1]This is for notational convenience. In the dynamic case z_e will contain other parameters in addition to weights.

[2]The *competitive ratio* is computed as the worst-case ratio between the value of the solution found by the algorithm and the value of an optimal solution.

$[0,1]$, we say that a greedy OCRS π is (b,c)-*selectable* if, for each $e \in \mathcal{E}$, and given $\boldsymbol{x} \in b\mathcal{P}_{\mathcal{F}}$ (i.e., belonging to a down-scaled version of $\mathcal{P}_{\mathcal{F}}$),

$$\Pr_{\pi,R(\boldsymbol{x})}\left[S \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{x}} \quad \forall S \subseteq R(\boldsymbol{x}), S \in \mathcal{F}_{\pi,\boldsymbol{x}}\right] \geq c.$$

Intuitively, this means that, with probability at least c, the random set $R(\boldsymbol{x})$ is such that an element e is selected no matter what other elements I of $R(x)$ have been selected so far, as long as $I \in \mathcal{F}_{\pi,\boldsymbol{x}}$. This guarantees that an element is selected with probability at least c against any adversary, which implies a bc competitive ratio with respect to the offline optimum (see the Appendix in the full version of the paper for further details). Now, we provide an example due to Feldman, Svensson, and Zenklusen (2016) of a feasibility constraint family where OCRSs guarantee a constant competitive ratio against the offline optimum. We will build on this example throughout the paper in order to provide intuition for the main concepts.

Example 1 (Theorem 2.7 in (Feldman, Svensson, and Zenklusen 2016))**.** *Given a graph $G = (\mathcal{V}, \mathcal{E})$, with $|\mathcal{E}| = m$ edges, we consider a matching feasibility polytope $\mathcal{P}_{\mathcal{F}} = \left\{\boldsymbol{x} \in [0,1]^m : \sum_{e\in\delta(u)} x_e \leq 1, \forall u \in \mathcal{V}\right\}$, where $\delta(u)$ denotes the set of all adjacent edges to $u \in \mathcal{V}$. Given $b \in [0,1]$, the OCRS takes as input $\boldsymbol{x} \in b\mathcal{P}_{\mathcal{F}}$, and samples each edge e with probability x_e to build $R(\boldsymbol{x})$. Then, it selects each edge $e \in R(\boldsymbol{x})$, upon its arrival, with probability $(1-e^{-x_e})/x_e$ only if it is feasible. Then, the probability to select any edge $e = (u,v)$ (conditioned on being sampled) is*

$$\begin{aligned}&\frac{1-e^{-x_e}}{x_e} \cdot \prod_{e' \in \delta(u)\cup\delta(v)\setminus\{e\}} e^{-x_{e'}} \\ &= \frac{1-e^{-x_e}}{x_e} \cdot e^{-\sum_{e'\in\delta(u)\cup\delta(v)\setminus\{e\}} x_{e'}} \geq \frac{1-e^{-x_e}}{x_e} \cdot e^{-2b} \\ &\geq e^{-2b},\end{aligned}$$

where the inequality follows from $x_e \in b\mathcal{P}_{\mathcal{F}}$, i.e., $\sum_{e'\in\delta(u)\setminus\{e\}} x_{e'} \leq b - x_e$, and similarly for $\delta(v)$. Note that in order to obtain an unconditional probability, we need to multiply the above by a factor x_e.

We remark that this example resembles closely our introductory motivating application, where financial transactions need to be assigned to reviewers upon their arrival. Moreover, Feldman, Svensson, and Zenklusen (2016) give explicit constructions of (b,c)-selectable greedy OCRSs for knapsack, matching, matroidal constraints, and their intersection. We include a discussion of their feasibility polytopes in the Appendix of the full version of the paper. Ezra et al. (2020) generalize the above online selection procedure to a setting where elements arrive in batches rather than one at a time; we provide a discussion of such setting in the Appendix of in the full version of the paper.

Fully Dynamic Online Selection

The *fully dynamic online selection problem* is characterized by the definition of *temporal packing constraints*. We generalize the online selection model (Section) by introducing an *activity time* $d_e \in [m]$ for each element. Element e arrives at time s_e and, if it is selected by the algorithm, it remains active up to time $s_e + d_e$ and "blocks" other elements from being selected. Elements arriving after that time can be selected by the algorithm. In this setting, each element $e \in \mathcal{E}$ is characterized by a tuple of attributes $z_e := (w_e, d_e)$. Let $\mathcal{F}^{\boldsymbol{d}} := (\mathcal{E}, \mathcal{I}^{\boldsymbol{d}})$ be the family of *temporal packing feasibility constraints* where elements block other elements in the same independent set according to activity time vector $\boldsymbol{d} = (d_e)_{e\in\mathcal{E}}$. The goal of fully dynamic online selection is selecting an independent set in $\mathcal{I}^{\boldsymbol{d}}$ whose cumulative weight is as large as possible (i.e., as close as possible to the offline optimum). We can naturally extend the expression for packing polytopes in the standard setting to the temporal one for every feasibility constraint family, by exploiting the following notion of *active elements*.

Definition 4 (Active Elements)**.** *For element $e \in \mathcal{E}$ and given $\{z_e\}_{e\in\mathcal{E}}$, we denote the set of* active elements *as $\mathcal{E}_e := \{e' \in \mathcal{E} : s_{e'} \leq s_e \leq s_{e'} + d_{e'}\}$.*[3]

In this setting, we don't need to select an independent set $S \in \mathcal{F}$, but, in a less restrictive way, we only require that for each incoming element we select a feasible subset of the set of active elements.

Definition 5 (Temporal packing constraint polytope)**.** *Given $\mathcal{F} = (\mathcal{E}, \mathcal{I})$, a* temporal packing constraint polytope *$\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}} \subseteq [0,1]^m$ is such that $\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}} := \text{co}\left(\{\mathbf{1}_S : S \cap \mathcal{E}_e \in \mathcal{I}, \forall e \in \mathcal{E}\}\right).$*

Observation 1. *For a fixed element e, the temporal polytope is the convex hull of the collection containing all the sets such that $S \cap \mathcal{E}_e$ is feasible. This needs to be true for all $e \in \mathcal{E}$, meaning that we can rewrite the polytope and the feasibility set as $\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}} = \text{co}\left(\bigcap_{e\in\mathcal{E}} \{\mathbf{1}_S : S \cap \mathcal{E}_e \in \mathcal{F}\}\right)$, and $\mathcal{I}^{\boldsymbol{d}} = \bigcap_{e\in\mathcal{E}} \{S : S \cap \mathcal{E}_e \in \mathcal{I}\}$. Moreover, when $\boldsymbol{d}$ and $\boldsymbol{d}'$ differ for at least one element e, that is $d_e < d'_e$, then $\mathcal{E}_e \subseteq \mathcal{E}'_e$. Then, $\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}} \supseteq \mathcal{P}^{\boldsymbol{d}'}_{\mathcal{F}}$, $\mathcal{I}^{\boldsymbol{d}} \supseteq \mathcal{I}^{\boldsymbol{d}'}$.*

We now extend Example 1 to account for activity times. In the Appendix of the full version of the paper, we also work out the reduction from standard to *temporal* packing constraints for a number of examples, including rank-1 matroids (single-choice), knapsack, and general matroid constraints.

Example 2. *We consider the temporal extension of the matching polytope presented in Example 1, that is*

$$\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}} = \left\{\boldsymbol{y} \in [0,1]^m : \sum_{e\in\delta(u)\cap\mathcal{E}_e} x_e \leq 1, \forall u \in V, \forall e \in \mathcal{E}\right\}.$$

Let us use the same OCRS as in the previous example, but where "feasibility" only concerns the subset of active edges in $\delta(u) \cup \delta(v)$. The probability to select an edge $e = (u,v)$ is

$$\frac{1-e^{-x_e}}{x_e} \cdot \prod_{e'\in\delta(u)\cup\delta(v)\cap\mathcal{E}_e\setminus\{e\}} e^{-x_{e'}} \geq \frac{1-e^{-x_e}}{x_e} \cdot e^{-2b} \geq e^{-2b},$$

which is obtained in a similar way to Example 1.

[3]Note that, since for distinct elements e, e', we have $s_{e'} \neq s_e$, we can equivalently define the set of active elements as $\mathcal{E}_e := \{e' \in \mathcal{E} : s_{e'} < s_e \leq s_{e'} + d_{e'}\} \cup \{e\}$.

Algorithm 1: Greedy OCRS Black-box Reduction

Input: Feasibility families $\mathcal{F}$ and $\mathcal{F}^{\boldsymbol{d}}$, polytopes $\mathcal{P}_{\mathcal{F}}$ and $\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$, OCRS π for $\mathcal{F}$, a point $\boldsymbol{x} \in b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$;
Initialize $S^{\boldsymbol{d}} \leftarrow \emptyset$;
Sample $R(\boldsymbol{x})$ such that $\Pr[e \in R(\boldsymbol{x})] = x_e$;
for $e \in \mathcal{E}$ **do**
  Upon arrival of element e, compute the set of currently active elements $\mathcal{E}_e$;
  if $(S^{\boldsymbol{d}} \cap \mathcal{E}_e) \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{y}}$ **then**
    Execute the original greedy OCRS $\pi(\boldsymbol{x})$;
    Update $S^{\boldsymbol{d}}$ accordingly;
  else
    Discard element e;
return set $S^{\boldsymbol{d}}$;

The above example suggests to look for a general reduction that maps an OCRS for the standard setting, to an OCRS for the temporal setting, while achieving at least the same competitive ratio.

OCRS for Fully Dynamic Online Selection

The first black-box reduction which we provide consists in showing that a (b,c)-selectable greedy OCRS for standard packing constraints implies the existence of a (b,c)-selectable greedy OCRS for temporal constraints. In particular, we show that the original greedy OCRS working for $\boldsymbol{x} \in b\mathcal{P}_{\mathcal{F}}$ can be used to construct another greedy OCRS for $\boldsymbol{y} \in b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$. To this end, Algorithm 1 provides a way of exploiting the original OCRS π in order to manage temporal constraints. For each element e, and given the induced subfamily of packing feasible sets $\mathcal{F}_{\pi,\boldsymbol{y}}$, the algorithm checks whether the set of previously selected elements $S^{\boldsymbol{d}}$ which are still active in time, together with the new element e, is feasible with respect to $\mathcal{F}_{\pi,\boldsymbol{y}}$. If that is the case, the algorithm calls the OCRS π. Then, if the OCRS π for input $\boldsymbol{y}$ decided to select the current element e, the algorithm adds it to $S^{\boldsymbol{d}}$, otherwise the set remains unaltered. We remark that such a procedure is agnostic to whether the original greedy OCRS is deterministic or randomized. We observe that, due to a larger feasibility constraint family, the number of independent sets have increased with respect to the standard setting. However, we show that this does not constitute a problem, and an equivalence between the two settings can be established through the use of Algorithm 1. The following result shows that Algorithm 1 yields a (b,c)-selectable greedy OCRS for temporal packing constraints.

Theorem 1. *Let $\mathcal{F}, \mathcal{F}^{\boldsymbol{d}}$ be the standard and temporal packing constraint families, respectively, and let their corresponding polytopes be $\mathcal{P}_{\mathcal{F}}$ and $\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$. Let $\boldsymbol{x} \in b\mathcal{P}_{\mathcal{F}}$ and $\boldsymbol{y} \in b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$, and consider a (b,c)-selectable greedy OCRS π for $\mathcal{F}_{\pi,\boldsymbol{x}}$. Then, Algorithm 1 equippend with π is a (b,c)-selectable greedy OCRS for $\mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{y}}$.*

Proof. Let us denote by $\hat{\pi}$ the procedure described in Algorithm 1. First, we show that $\hat{\pi}$ is a greedy OCRS for $\mathcal{F}^{\boldsymbol{d}}$.

Greedyness. It is clear from the setting and the construction that elements arrive one at a time, and that $\hat{\pi}$ irrevocably selects an incoming element only if it is feasible, and before seeing the next element. Indeed, in the *if* statement of Algorithm 1, we check that the active subset of the elements selected so far, together with the new arriving element e, is feasible against the subfamily $\mathcal{F}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}$. Constraint subfamily $\mathcal{F}_{\pi,\boldsymbol{x}}$ is induced by the original OCRS π, and point $\boldsymbol{x}$ belongs to the polytope $b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$. Note that we do not necessarily add element e to the running set $S^{\boldsymbol{d}}$, even though feasible, but act as the original greedy OCRS would have acted. All that is left to be shown is that such a procedure defines a subfamily of feasibility constraints $\mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}^{\boldsymbol{d}}$. By construction, on the arrival of each element e, we guarantee that $S^{\boldsymbol{d}}$ is a set such that its subset of active elements is feasible. This means that $S^{\boldsymbol{d}} \cap \mathcal{E}_e \in \mathcal{F}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}$. Then,

$$S^{\boldsymbol{d}} \in \mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}} := \bigcap_{e \in \mathcal{E}} \{S : S \cap \mathcal{E}_e \in \mathcal{F}_{\pi,\boldsymbol{x}}\}.$$

Finally, $\mathcal{F}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}$ implies that $\mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}} \subseteq \mathcal{F}^{\boldsymbol{d}}$, which shows that $\hat{\pi}$ is greedy. With the above, we can now turn to demonstrate (b,c)-selectability.

Selectability. Upon arrival of element $e \in \mathcal{E}$, let us consider S and $S^{\boldsymbol{d}}$ to be the sets of elements already selected by π and $\hat{\pi}$, respectively. By the way in which the constraint families are defined, and by construction of $\hat{\pi}$, we can observe that, given $\boldsymbol{x} \in b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$ and $\boldsymbol{y} \in b\mathcal{P}_{\mathcal{F}}$, for all $S \subseteq R(\boldsymbol{y})$ such that $S \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{y}}$, there always exists a set $S^{\boldsymbol{d}} \subseteq R(\boldsymbol{x})$ such that $(S^{\boldsymbol{d}} \cap \mathcal{E}_e) \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{x}}$. This establishes an injection between the selected set under standard constraints, and its counterpart under temporal constraints. We observe that, for all $e \in \mathcal{E}$ and $\boldsymbol{x} \in b\mathcal{P}^{\boldsymbol{d}}_{\mathcal{F}}$,

$$\Pr\big[S^{\boldsymbol{d}} \cup \{e\} \in \mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}} \quad \forall S^{\boldsymbol{d}} \subseteq R(\boldsymbol{x}), S^{\boldsymbol{d}} \in \mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}}\big] = \Pr\big[(S^{\boldsymbol{d}} \cap \mathcal{E}_e) \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{x}} \ \forall S^{\boldsymbol{d}} \subseteq R(\boldsymbol{x}), S^{\boldsymbol{d}} \cap \mathcal{E}_e \in \mathcal{F}^{\boldsymbol{d}}_{\pi,\boldsymbol{x}}\big].$$

Hence, since for greedy OCRS π and $\boldsymbol{y} \in b\mathcal{P}_{\mathcal{F}}$, we have that $\Pr[S \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{y}} \ \forall S \subseteq R(\boldsymbol{y}), S \in \mathcal{F}_{\pi,\boldsymbol{y}}] \geq c$, we can conclude by the injection above that

$$\Pr\big[(S^{\boldsymbol{d}} \cap \mathcal{E}_e) \cup \{e\} \in \mathcal{F}_{\pi,\boldsymbol{x}} \quad \forall S^{\boldsymbol{d}} \subseteq R(\boldsymbol{x}), S^{\boldsymbol{d}} \cap \mathcal{E}_e \in \mathcal{F}_{\pi,\boldsymbol{x}}\big] \geq c.$$

The theorem follows. □

We remark that the above reduction is agnostic to the weight scale, i.e., we need not assume that $w_e \in [0,1]$ for all $e \in E$. In order to further motivate the significance of Algorithm 1 and Theorem 1, we explicitly reduce the standard setting to the fully dynamic one in the full version of the paper for single-choice, and provide a general recipe for all packing constraints.

Fully Dynamic Online Selection under Partial Information

In this section, we study the case in which the decision-maker has to act under partial information. In particular, we focus on the following online sequential extension of the full-information problem: at each stage $t \in [T]$, a decision maker faces a new instance of the fully dynamic online selection problem. An unknown vector of weights $\boldsymbol{w}_t \in [0,1]^{|E|}$ is chosen by an adversary at each stage t, while feasibility set $\mathcal{F}^{\boldsymbol{d}}$ is known and fixed across all T stages. This setting is analogous to the one recently studied by Gergatsouli and Tzamos (2022) in the context of *Pandora's box problems*. A crucial difference with the online selection problem with full-information studied in Section is that, at each step t, the decision maker has to decide whether to select or discard an element before observing its weight. In particular, at each t, the decision maker takes an action $\boldsymbol{a}_t := \mathbf{1}_{S_t^{\boldsymbol{d}}}$, where $S_t^{\boldsymbol{d}} \in \mathcal{F}^{\boldsymbol{d}}$ is the feasible set selected at stage t. The choice of $\boldsymbol{a}_t$ is made before observing $\boldsymbol{w}_t$. The objective of maximizing the cumulative sum of weights is encoded in the reward function $f : [0,1]^{2m} \ni (\boldsymbol{a}, \boldsymbol{w}) \mapsto \langle \boldsymbol{a}, \boldsymbol{w} \rangle \in [0,1]$, which is the reward obtained by playing $\boldsymbol{a}$ with weights $\boldsymbol{w} = (w_e)_{e \in \mathcal{E}}$. [4]

In this setting, we can think of $\mathcal{F}^{\boldsymbol{d}}$ as the set of *super-arms* in a combinatorial online optimization problem. Our goal is designing online algorithms which have a performance *close* to that of the best fixed super-arm in hindsight.[5] In the analysis, as it is customary when the online optimization problem has an NP-hard offline counterpart, we resort to the notion of α-regret. In particular, given a set of feasible actions $\mathcal{X}$, we define an algorithm's α-regret up to time T as

$$\text{Regret}_\alpha(T) := \alpha \max_{\boldsymbol{x} \in \mathcal{X}} \left\{ \sum_{t=1}^{T} f(\boldsymbol{x}, \boldsymbol{w}_t) \right\} - \mathbb{E}\left[\sum_{t=1}^{T} f(\boldsymbol{x}_t, \boldsymbol{w}_t) \right],$$

where $\alpha \in (0,1]$ and $\boldsymbol{x}_t$ is the strategy output by the online algorithm at time t. We say that an algorithm has the *no-α-regret* property if $\text{Regret}_\alpha(T)/T \to 0$ for $T \to \infty$.

The main result of the section is providing a black-box reduction that yields a no-α-regret algorithm for any fully dynamic online selection problem admitting a temporal OCRS. We provide no-α-regret frameworks for three scenarios:

- *full-feedback model*: after selecting $\boldsymbol{a}_t$ the decision-maker observes the exact reward function $f(\cdot, \boldsymbol{w}_t)$.
- *semi-bandit feedback with white-box OCRS*: after taking a decision at time t, the algorithm observes $w_{t,e}$ for each element $e \in S_t^{\boldsymbol{d}}$ (i.e., each element selected at t). Moreover, the decision-maker has exact knowledge of the procedure employed by the OCRS, which can be easily simulated.
- *semi-bandit feedback with oracle access to the OCRS*: the decision maker has semi-bandit feedback and the OCRS is given as a black-box which can be queried once per step t.

Full-feedback Setting

In this setting, after selecting $\boldsymbol{a}_t$, the decision-maker gets to observe the reward function $f(\cdot, \boldsymbol{w}_t)$. In order to achieve performance close to that of the best fixed super-harm in hindsight the idea is to employ the α-competitive OCRS designed in Section by feeding it with a fractional solution $\boldsymbol{x}_t$ computed by considering the weights selected by the adversary up to time $t-1$.[6]

Algorithm 2: FULL-FEEDBACK ALGORITHM

Input: T, $\mathcal{F}^{\boldsymbol{d}}$, temporal OCRS $\hat{\pi}$, subroutine RM
Initialize RM for strategy space $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$
for $t \in [T]$ **do**
  $\boldsymbol{x}_t \leftarrow$ RM.RECOMMEND()
  $\boldsymbol{a}_t \leftarrow$ execute OCRS $\hat{\pi}$ with input $\boldsymbol{x}_t$
  Play $\boldsymbol{a}_t$, and subsequently observe $f(\cdot, \boldsymbol{w}_t)$
  RM.UPDATE($f(\cdot, \boldsymbol{w}_t)$)

Let us assume to have at our disposal a no-α-regret algorithm for decision space $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$. We denote such regret minimizer as RM, and we assume it offers two basic operations: i) RM.RECOMMEND() returns a vector in $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$; ii) RM.UPDATE($f(\cdot, \boldsymbol{w})$) updates the internal state of the regret minimizer using feedback received by the environment in the form of a reward function $f(\cdot, \boldsymbol{w})$. Notice that the availability of such component is not enough to solve our problem since at each t we can only play a super-arm $\boldsymbol{a}_t \in \{0,1\}^m$ feasible for $\mathcal{F}^{\boldsymbol{d}}$, and not the strategy $\boldsymbol{x}_t \in \mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}} \subseteq [0,1]^m$ returned by RM. The decision-maker can exploit the subroutine RM together with a temporal greedy OCRS $\hat{\pi}$ by following Algorithm 2. We can show that, if the algorithm employs a regret minimizer for $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$ with a sublinear cumulative regret upper bound of $\mathfrak{R}^T$, the following result holds.

Theorem 2. *Given a regret minimizer* RM *for decision space $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$ with cumulative regret upper bound $\mathfrak{R}^T$, and an α-competitive temporal greedy OCRS, Algorithm 2 provides*

$$\alpha \max_{S \in \mathcal{I}^{\boldsymbol{d}}} \sum_{t=1}^{T} f(\mathbf{1}_S, \boldsymbol{w}_t) - \mathbb{E}\left[\sum_{t=1}^{T} f(\boldsymbol{a}_t, \boldsymbol{w}_t) \right] \leq \mathfrak{R}^T.$$

Since we are assuming the existence of a polynomial-time separation oracle for the set $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$, then the LP $\arg\max_{\boldsymbol{x} \in \mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}} f(\boldsymbol{x}, \boldsymbol{w})$ can be solved in polynomial time for any $\boldsymbol{w}$. Therefore, we can instantiate a regret minimizer for $\mathcal{P}_{\mathcal{F}}^{\boldsymbol{d}}$ by using, for example, *follow-the-regularised-leader* which yields $\mathfrak{R}^T \leq \tilde{O}(m\sqrt{T})$ (Orabona 2019).

Semi-Bandit Feedback with White-Box OCRS

In this setting, given a temporal OCRS $\hat{\pi}$, it is enough to show that we can compute the probability that a certain super-arm $\boldsymbol{a}$ is selected by $\hat{\pi}$ given a certain order of arrivals at stage t and a vector of weights $\boldsymbol{w}$. If that is the case, we can build a no-α-regret algorithm with regret upper bound of $\tilde{O}(m\sqrt{T})$ by employing Algorithm 2 and by instantiating the regret minimizer RM as the *online stochastic mirror descent* (OSMD) framework by Audibert, Bubeck,

[4]The analysis can be easily extended to arbitrary functions linear in both terms.

[5]As we argue in the Appendix of the full version of the paper it's not possible to be competitive with respect to more powerful benchmarks without sacrificing the computational efficiency (*i.e.*, without having an exponential per-iteration running time).

[6]We remark that a (b,c)-selectable OCRS yields a bc competitive ratio. In the following, we let $\alpha := bc$.

and Lugosi (2014). We observe that the regret bound obtained is this way is tight in the semi-bandit setting (Audibert, Bubeck, and Lugosi 2014). Let $q_t(e)$ be the probability with which our algorithm selects element e at time t. Then, we can equip OSMD with the following unbiased estimator of the vector of weights: $\hat{w}_{t,e} := w_{t,e} a_{t,e} / q_t(e)$. [7] In order to compute $q_t(\cdot)$ we need to have observed the order of arrival at stage t, the weights corresponding to super-arm $\boldsymbol{a}_t$, and we need to be able to compute the probability with which the OCRS selected e at t. This the reason for which we talk about "white-box" OCRS, as we need to simulate ex post the procedure followed by the OCRS in order to compute $q_t(\cdot)$. When we know the procedure followed by the OCRS, we can always compute $q_t(e)$ for any element e selected at stage t, since at the end of stage t we know the order of arrival, weights for selected elements, and the initial fractional solution $\boldsymbol{x}_t$. We provide further intuition as for how to compute such probabilities through the running example of matching constraints.

Example 3. *Consider Algorithm 2 initialized with the OCRS of Example 1. Given stage t, we can safely limit our attention to selected edges (i.e., elements e such that $a_{t,e} = 1$). Indeed, all other edges will either be unfeasible (which implies that the probability of selecting them is 0), or they were not selected despite being feasible. Consider an arbitrary element e among those selected. Conditioned on the past choices up to element e, we know that $e \in \boldsymbol{a}_t$ will be feasible with certainty, and thus the (unconditional) probability it is selected is simply $q_t(e) = 1 - e^{-y_{t,e}}$.*

Semi-Bandit Feedback and Oracle Access to OCRS

As in the previous case, at each stage t the decision maker can only observe the weights associated to each edge selected by $\boldsymbol{a}_t$. Therefore, they have no counterfactual information on their reward had they selected a different feasible set. On top of that, we assume that the OCRS is given as a black-box, and therefore we cannot compute ex post the probabilities $q_t(e)$ for selected elements. However, we show that it is possible to tackle this setting by exploiting a reduction from the semi-bandit feedback setting to the full-information feedback one. In doing so, we follow the approach first proposed by Awerbuch and Kleinberg (2008). The idea is to split the time horizon T into a given number of equally-sized blocks. Each block allows the decision maker to simulate a single stage of the full information setting. We denote the number of blocks by Z, and each block $\tau \in [Z]$ is composed by a sequence of consecutive stages I_τ. Algorithm 3 describes the main steps of our procedure. In particular, the algorithm employs a procedure RM, an algorithm for the full feedback setting as the one described in the previous section, that exposes an interface with the two operation of a traditional regret minimizer. During each block τ, the full-information subroutine is used to compute a vector $\boldsymbol{x}_\tau$. Then, in most stages of the window I_τ, the decision $\boldsymbol{a}_t$ is computed by feeding $\boldsymbol{x}_\tau$ to the OCRS. A few stages are chosen uniformly at random to estimate utilities provided by other feasible sets (*i.e.*, exploration phase). After the execution of all the stages in the window I_τ, the algorithm computes estimated reward functions and uses them to update the full-information regret minimizer.

Algorithm 3: SEMI-BANDIT-FEEDBACK ALGORITHM WITH ORACLE ACCESS TO OCRS

```
Input: T, F^d, temporal OCRS π̂, full-feedback
algorithm RM for decision space P^d_F
Let Z be initialized as in Theorem 3, and initialize
  RM appropriately
for τ = 1, ..., Z do
    I_τ ← {(τ − 1)T/Z + 1, ..., τT/Z}
    Choose a random permutation p : [m] → E, and
      t_1, ..., t_m stages at random from I_τ
    x_τ ← RM.RECOMMEND()
    for t = (τ − 1)T/Z + 1, ..., τT/Z do
        if t = t_j for some j ∈ [m] then
            x_t ← 1_{S^d} for a feasible set S^d
              containing p(j)
        else
            x_t ← x_τ
        Play a_t obtained from the OCRS π̂ executed
          with fractional solution x_t
    Compute estimators f̃_τ(e) of
      f_τ(e) := 1/|I_τ| Σ_{t∈I_τ} f(1_e, w_t) for each e ∈ E
    RM.UPDATE(f̃_τ(·))
```

Let $p : [m] \to \mathcal{E}$ be a random permutation of elements in $\mathcal{E}$. Then, for each $e \in \mathcal{E}$, by letting j be the index such that $p(j) = e$ in the current block τ, an unbiased estimator $\tilde{f}_\tau(e)$ of $f_\tau(e) := \frac{1}{|I_\tau|} \sum_{t \in I_\tau} f(\mathbf{1}_e, \boldsymbol{w}_t)$ can be easily obtained by setting $\tilde{f}_\tau(e) := f(\mathbf{1}_e, \boldsymbol{w}_{t_j})$. Then, it is possible to show that our algorithm provides the following guarantees.

Theorem 3. *Given a temporal packing feasibility set $\mathcal{F}^{\boldsymbol{d}}$, and an α-competitive OCRS $\hat{\pi}$, let $Z = T^{2/3}$, and the full feedback subroutine* RM *be defined as per Theorem 2. Then Algorithm 3 guarantees that*

$$\alpha \max_{S \in \mathcal{I}^{\boldsymbol{d}}} \sum_{t=1}^{T} f(\mathbf{1}_S, \boldsymbol{w}_t) - \mathbb{E}\left[\sum_{t=1}^{T} f(\boldsymbol{a}_t, \boldsymbol{w}_t)\right] \leq \tilde{O}(T^{2/3}).$$

Conclusion and Future Work

In this paper we introduce fully dynamic online selection problems in which selected items affect the combinatorial constraints during their activity times. We presented a generalization of the OCRS approach that provides near optimal competitive ratios in the full-information model, and no-α-regret algorithms with polynomial per-iteration running time with both full- and semi-bandit feedback. Our framework opens various future research directions. For example, it would be particularly interesting to understand whether a variation of Algorithms 2 and 3 can be extended to the case in which the adversary changes the constraint family at each stage. Moreover, the study of the bandit-feedback model remains open, and no regret bound is known for that setting.

[7] We observe that $\hat{w}_{t,e}$ is equal to 0 when e has not been selected at stage t because, in that case, $a_{t,e} = 0$.

Acknowledgements

The authors of Sapienza are supported by the Meta Research grant on "Fairness and Mechanism Design", the ERC Advanced Grant 788893 AMDROMA "Algorithmic and Mechanism Design Research in Online Markets", the MIUR PRIN project ALGADIMAR "Algorithms, Games, and Digital Markets".

References

Atsidakou, A.; Papadigenopoulos, O.; Basu, S.; Caramanis, C.; and Shakkottai, S. 2021. Combinatorial Blocking Bandits with Stochastic Delays. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, 404–413.

Audibert, J.-Y.; Bubeck, S.; and Lugosi, G. 2014. Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1): 31–45.

Awerbuch, B.; and Kleinberg, R. 2008. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1): 97–114.

Basu, S.; Papadigenopoulos, O.; Caramanis, C.; and Shakkottai, S. 2021. Contextual Blocking Bandits. In *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, 271–279.

Basu, S.; Sen, R.; Sanghavi, S.; and Shakkottai, S. 2019. Blocking Bandits. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Bishop, N.; Chan, H.; Mandal, D.; and Tran-Thanh, L. 2020. Adversarial Blocking Bandits. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Cesa-Bianchi, N.; and Lugosi, G. 2012. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5): 1404–1422.

Chawla, S.; Hartline, J. D.; Malec, D. L.; and Sivan, B. 2010. Multi-Parameter Mechanism Design and Sequential Posted Pricing. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, 311–320. New York, NY, USA: Association for Computing Machinery. ISBN 9781450300506.

Chekuri, C.; Vondrák, J.; and Zenklusen, R. 2011. Submodular Function Maximization via the Multilinear Relaxation and Contention Resolution Schemes. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, 783–792. New York, NY, USA: Association for Computing Machinery. ISBN 9781450306911.

Demetrescu, C.; Eppstein, D.; Galil, Z.; and Italiano, G. F. 2010. *Dynamic Graph Algorithms*, 9. Chapman & Hall/CRC, 2 edition. ISBN 9781584888222.

Dickerson, J.; Sankararaman, K.; Srinivasan, A.; and Xu, P. 2018. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Ezra, T.; Feldman, M.; Gravin, N.; and Tang, Z. G. 2020. Online Stochastic Max-Weight Matching: Prophet Inequality for Vertex and Edge Arrival Models. In *EC'20*, 769–787.

Feige, U. 1998. A Threshold of Ln n for Approximating Set Cover. *J. ACM*, 45(4): 634–652.

Feldman, M.; Svensson, O.; and Zenklusen, R. 2016. Online Contention Resolution Schemes. In Krauthgamer, R., ed., *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, 1014–1033. SIAM.

Gergatsouli, E.; and Tzamos, C. 2022. Online Learning for Min Sum Set Cover and Pandora's Box. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 7382–7403.

Gupta, A.; and Nagarajan, V. 2013. A Stochastic Probing Problem with Applications. In *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization*, IPCO'13, 205–216. Berlin, Heidelberg: Springer-Verlag. ISBN 9783642366932.

Hajiaghayi, M. T.; Kleinberg, R.; and Sandholm, T. 2007. Automated Online Mechanism Design and Prophet Inequalities. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, 58–65. AAAI Press. ISBN 9781577353232.

Kleinberg, R.; Niculescu-Mizil, A.; and Sharma, Y. 2010. Regret Bounds for Sleeping Experts and Bandits. *Mach. Learn.*, 80(2–3): 245–272.

Kleinberg, R.; and Weinberg, S. M. 2012. Matroid prophet inequalities. In *STOC'12*, 123–136.

Orabona, F. 2019. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*.