Online Learning for Min Sum Set Cover and Pandora's Box

Evangelia Gergatsouli [1]   Christos Tzamos [1]

Abstract

Two central problems in Stochastic Optimization are MIN SUM SET COVER and PANDORA'S BOX. In PANDORA'S BOX, we are presented with n boxes, each containing an unknown value and the goal is to open the boxes in some order to minimize the sum of the search cost and the smallest value found. Given a distribution of value vectors, we are asked to identify a near-optimal search order. MIN SUM SET COVER corresponds to the case where values are either 0 or infinity. In this work, we study the case where the value vectors are not drawn from a distribution but are presented to a learner in an online fashion. We present a computationally efficient algorithm that is constant-competitive against the cost of the optimal search order. We extend our results to a bandit setting where only the values of the boxes opened are revealed to the learner after every round. We also generalize our results to other commonly studied variants of PANDORA'S BOX and MIN SUM SET COVER that involve selecting more than a single value subject to a matroid constraint.

1. Introduction

One of the fundamental problems in stochastic optimization is PANDORA'S BOX problem, first introduced by Weitzman in (Weitzman, 1979). The problem asks to select among n alternatives, called boxes, one with a low value. In the stochastic version of the problem, it is assumed that values in the boxes are drawn from a known distribution and the actual realization of any box can be revealed at a cost after inspection.

The goal is to design an algorithm that efficiently searches

among the n alternatives to find a low value while also paying a low inspection cost. We thus aim to minimize the sum of the search costs the algorithm pays and the value of the alternative(s) it chooses in the end. In the standard version of PANDORA'S BOX a single value must be chosen, but we also consider common generalizations that require k distinct alternatives to be chosen, or alternatives that form a matroid basis.

While most of the literature has focused on the stochastic case, where there is a known distribution of values given in advance, we instead consider an *online* version of the problem played over T rounds, where in each round a different realization of values in the boxes is adversarially chosen. The goal of the learner is to pick a good strategy of opening boxes in every round that guarantees low regret compared to choosing in hindsight the optimal policy for the T rounds from a restricted family of policies.

In this work, we mainly consider policies that fix the order in which boxes are explored but have free arbitrary stopping rules. Such policies are called *partially-adaptive* and are known to be optimal in many cases, most notably in the stochastic version where the values of the boxes are drawn independently. Such policies are also optimal in the special case of the PANDORA'S BOX problem where the values in the boxes are either 0 or ∞. This case is known as the MIN SUM SET COVER problem (MSSC) and is a commonly studied problem in the area of approximation algorithms.

1.1. Our Results

Our work presents a simple but powerful framework for designing online learning algorithms for PANDORA'S BOX, MSSC and other related problems. Our framework yields approximately low-regret algorithms for these problems through a three step process:

1. We first obtain convex relaxations of the instances of every round.

2. We then apply online-convex optimization to obtain good fractional solutions to the relaxed instances achieving low regret.

3. We finally round the fractional solutions to integral solutions for the original instances at a small multi-

---

[1]Department of Computer Sciences, University of Wisconsin-Madison, Madison, WI, USA. Correspondence to: Evangelia Gergatsouli <evagerg@cs.wisc.edu>, Christos Tzamos <tzamos@wisc.edu>.

plicative loss.

Through this framework, we obtain a

- 9.22-approximate no-regret algorithm for the problem of selecting 1 box.

- $O(1)$-approximate no-regret algorithm for the problem of selecting k boxes.

- $O(\log k)$-approximate no-regret algorithm for the problem of selecting a rank k matroid basis.

We start by presenting these results in the **full information** setting (section 3) where the values of all boxes are revealed after each round, once the algorithm has made its choices.

A key contribution of our work is to further extend these results to a more-realistic **bandit setting** (section 4). In this setting, the algorithm only observes the values for the boxes it explored in each round and can only use this information to update its strategy for future rounds. In each round there is also the option of obtaining the full information by paying a price. We show that even under this more pessimistic setting we can obtain approximately no-regret algorithm with the same approximation guarantees as above.

We also provide stronger regret guarantees against more restricted classes of algorithms for the PANDORA'S BOX and MSSC problems that are non-adaptive (section 5).

All the algorithms we develop in this paper are computationally efficient. As such, the approximation guarantees given above are approximately tight since it is NP-hard to improve on these beyond small constants even when competing with the simpler non-adaptive benchmark. In particular, it was shown in (Feige et al., 2004) that even the special case of MSSC is APX-hard and cannot be approximated within a smaller factor than 4. It is an interesting open question to what extent these bounds can be improved with unlimited computational power. While in the stochastic version, this would trivialize the problem, in the online setting the obtained approximation factors may still be necessary information theoretically.

1.2. Comparison with Previous Work

Our work is closely related to the work of (Chawla et al., 2020). In that work, the authors study a stochastic version of PANDORA'S BOX with an arbitrarily correlated distribution and aim to approximate the optimal partially adaptive strategies. We directly extend all the results of (Chawla et al., 2020) in the online non-stochastic setting, where we are required at each round to solve an instance of the problem.

Another very related paper is the work of (Fotakis et al., 2020) that also studies the online learning problem but focuses specifically on the MIN SUM SET COVER problem

and its generalization (GMSSC) that asks to select k alternatives instead of one. Our work significantly improves their results in several ways.

- We provide a simpler algorithm based on online convex optimization that does not rely on calculating gradients. We immediately obtain all our results through the powerful framework that we develop.

- This allows us to study more complex constraints like matroid rank constraints as well as study the more general PANDORA'S BOX. It is challenging to extend the results of (Fotakis et al., 2020) to such settings while keeping the required gradient computation task computationally tractable.

- Finally, we extend their results to a more natural bandit setting, where after each round we only have information about the alternatives that we explored rather than the whole instance.

In another recent work similar to ours, Esfandiari et al. (Esfandiari et al., 2019) consider a Multi-armed bandit version of PANDORA'S BOX problem which however greatly differs with ours in the following ways.

- In their setting each box has a type, and the algorithm is required to pick one box **per type**, while in our case the game is independent in each round.

- Their benchmark is a "prophet" who can choose the maximum reward per type of box, at the end of T rounds.

- The decision to pick a box is irrevocable[1] and they only consider threshold policies, as they relate the problem to prophet inequalities (see surveys (Hill & Kertz, 1992; Lucier, 2017; Correa et al., 2018) for more details on prophet inequalities).

1.3. Related Work

We model our search problem using PANDORA'S BOX, which was first introduced by Weitzman in the Economics literature (Weitzman, 1979). Since then, there has been a long line of research studying PANDORA'S BOX and its variants e.g. where boxes can be selected without inspection (Doval, 2018; Beyhaghi & Kleinberg, 2019), there is correlation between the boxes (Chawla et al., 2020), the boxes have to be inspected in a specific order (Boodaghians et al., 2020) or boxes are inspected in an online manner (Esfandiari et al., 2019). Some work is also done in the generalized setting where more information can be obtained

---

[1] The algorithm decides when seeing a box whether to select it or not, and cannot "go back" and select the maximum value seen.

for a price (Charikar et al., 2000; Gupta & Kumar, 2001; Chen et al., 2015b;a). Finally a long line of research considers more complex combinatorial constraints like budget constraints (Goel et al., 2006), packing constraints (Gupta & Nagarajan, 2013), matroid constraints (Adamczyk et al., 2016), maximizing a submodular function (Gupta et al., 2016; 2017), an approach via Markov chains (Gupta et al., 2019) and various packing and covering constraints for both minimization and maximization problems (Singla, 2018). In the special case of MSSC, the line of work was initiated by (Feige et al., 2004), and continued with improvements and generalizations to more complex constraints (Azar et al., 2009; Munagala et al., 2005; Bansal et al., 2010; Skutella & Williamson, 2011).

On the other hand, our work advances a recent line of research on the foundations of data-driven algorithm design, started by Gupta and Roughgarden (Gupta & Roughgarden, 2017), and continued by (Balcan et al., 2017; 2018a;b; Kleinberg et al., 2017; Weisz et al., 2018; Alabi et al., 2019), where they study parameterized families of algorithms in order to learn parameters to optimize the expected runtime or performance of the algorithm with respect to the underlying distribution. Similar work was done before (Gupta & Roughgarden, 2017) on self-improving algorithms (Ailon et al., 2006; Clarkson et al., 2010).

Furthermore, our results directly simplify and generalize the results of (Fotakis et al., 2020) in the case of partial feedback. Related to the partial feedback setting, (Flaxman et al., 2005) consider single bandit feedback and (Agarwal et al., 2010) consider multi-point bandit feedback. Both these works focus on finding good estimators for the gradient in order to run a gradient descent-like algorithm. For more pointers to the online convex optimization literature, we refer the reader to the survey by Shalev-Shwartz (Shalev-Shwartz, 2012) and the initial primal-dual analysis of the *Follow the Regularized Leader* family of algorithms by (Shalev-Shwartz & Singer, 2007).

2. Preliminaries

We evaluate the performance of our algorithms using *average regret*. We define the **average regret** of an algorithm $\mathcal{A}$ against a benchmark OPT, over a time horizon T as

$$\text{Regret}_{\text{OPT}}(\mathcal{A}, T) = \frac{1}{T} \sum_{t=1}^{T} \left(\mathcal{A}(t) - \text{OPT}(t) \right) \quad (1)$$

where $\mathcal{A}(t)$ and $\text{OPT}(t)$ is the cost at round t of $\mathcal{A}$ and OPT respectively. We similarly define the average α-**approximate regret** against a benchmark OPT as

$$\alpha\text{-Regret}_{\text{OPT}}(\mathcal{A}, T) = \frac{1}{T} \sum_{t=1}^{T} \left(\mathcal{A}(t) - \alpha \text{OPT}(t) \right). \quad (2)$$

We say that an algorithm $\mathcal{A}$ is **no regret** if $\text{Regret}_{\text{OPT}}(\mathcal{A}, T) = o(1)$. Similarly, we say that $\mathcal{A}$ is α-**approximate no regret** if $\alpha\text{-Regret}_{\text{OPT}}(\mathcal{A}, T) = o(1)$. Observe the we are always competing with an oblivious adversary, that selects the one option that minimizes the total loss over all rounds.

2.1. Problem Definitions

In PANDORA'S BOX we are given a set $\mathcal{B}$ of n boxes with unknown costs and a set of possible scenarios that determine these costs. In each round $t \in [T]$, an adversary chooses the instantiation of the costs in the boxes, called a *scenario*. Formally, a scenario at time t is a vector $\boldsymbol{c}(t) \in \mathbb{R}^n$ for any $t \in [T]$, where c_i^s denotes the cost for box i when scenario s is instantiated. Note that without loss of generality, we can assume that $c_i \leq n$, since if some is more than n we can ignore them, and if all are above n we automatically get a 2 approximation[2].

The goal of the algorithm at every round is to choose a box of small cost while spending as little time as possible gathering information. The algorithm cannot directly observe the instantiated scenario, however, it is allowed to "open" boxes one at a time. When opening a box, the algorithm observes the cost inside the box. In total, we want to minimize the regret over T rounds, relative to the optimal algorithm.

Formally, let $\mathcal{P}_t$ and c_i^t be the set of boxes opened and the cost of the box selected respectively by the algorithm at round $t \in [T]$. The cost of the algorithm $\mathcal{A}$ at round t is $\mathcal{A}(t) = \min_{i \in \mathcal{P}_t} c_i^t + |\mathcal{P}_t|$ and the goal is to minimize regret $\text{Regret}_{\text{OPT}}(\mathcal{A}, T)$.

Any algorithm can be described by a pair (σ, τ), where σ is a permutation of the boxes representing the order in which they are opened, and τ is a stopping rule – the time at which the algorithm stops opening and returns the minimum cost it has seen so far. Observe that in its full generality, an algorithm may choose the next box to open and the stopping time as a function of the identities and costs of the previous opened boxes.

Different Benchmarks. As observed in (Chawla et al., 2020), optimizing over the class of all such algorithms is intractable, therefore simpler benchmarks are considered.

- **The Non-adaptive Benchmark (NA)**: in this case the adversary chooses all the T scenarios about to come, and selects a fixed set of boxes to open, which is the same in every round. In this case, the $\text{OPT}(t)$ term in the regret does not depend on t.

---

[2]Since opening all boxes to find the minimum value costs us at most $n + \min_{i \in \mathcal{B}} c_i$, and the optimal also pays at least n

- **The Partially-adaptive Benchmark (PA)**: in this case, the adversary can have a different set of boxes to open in each round, which can depend on the algorithm's choices in rounds $1, \ldots, t-1$.

An important special case. A special case of PANDORA'S BOX is the MIN SUM SET COVER problem (MSSC). In this problem, the costs inside the boxes are either 0 or ∞. We say a scenario is *covered* or *satisfied* if, in our solution, we have opened a box that has value 0 for this scenario.

General feasibility constraints. We also study two more complex extensions of the problem. In the first one we are required to select exactly k boxes for some $k \geq 1$, and in the second, the algorithm is required to select a basis of a given matroid. We design **partially-adaptive** strategies that are approximately no-regret, for the different constraints and benchmarks described in this section.

2.2. Relaxations

2.2.1. SCENARIO - AWARE RELAXATION

Observe that the class of partially-adaptive strategies is still too large and complex, since the stopping rule can arbitrarily depend on the costs observed in boxes upon opening. One of the main contributions of (Chawla et al., 2020), which we are using in this work too, is that they showed it is enough to design a strategy that chooses an ordering of the boxes and performs well, assuming that we know when to stop. This relaxation of partially-adaptive, called *scenario-aware partially-adaptive* (SPA), greatly diminishes the space of strategies to consider, and makes it possible to design competitive algorithms, at the cost of an extra constant factor. This is formally stated in the lemma below. The proof can be found in (Chawla et al., 2020) and it is based on a generalization of ski-rental (Karlin et al., 1990).

Lemma 2.1 (Simplification of Theorem 3.4 from (Chawla et al., 2020)). *For a polynomial, in the number of boxes, α-approximate algorithm for scenario-aware partially adaptive strategies, there exists a polynomial time algorithm that is a $\frac{e}{e-1}\alpha$-approximation partially-adaptive strategy.*

2.2.2. FRACTIONAL RELAXATION AND ROUNDING

This first relaxation allows us to only focus on designing efficient SPA strategies which only require optimizing over the permutation of boxes. However both MSSC and PANDORA'S BOX are non-convex problems. We tackle this issue by using a convex relaxation of the problems, given by their linear programming formulation.

Definition 1 (Convex Relaxation). *Let Π be a minimization problem over a domain X with $g : X \to \mathbb{R}$ as its objective*

function, we say that a function $\overline{g} : \overline{X} \to \mathbb{R}$ is a convex relaxation of g, if

1. *The function $\overline{g}$ and its domain $\overline{X}$ are convex.*

2. *$X \subseteq \overline{X}$ and for any $x \in X$, $\overline{g}(x) \leq g(x)$.*

Using this definition, for our partially-adaptive benchmark we relax the domain $\overline{X} = \{x \in [0,1]^{n \times n} : \sum_i x_{it} = 1 \text{ and } \sum_t x_{it} = 1\}$ to be the set of doubly stochastic $n \times n$ matrices. We use a convex relaxation $\overline{g}^s$ similar to the one from the generalized min-sum set cover problem in (Bansal et al., 2010) and (Skutella & Williamson, 2011), but *scenario dependent*; for a given scenario s, the relaxation $\overline{g}^s$ changes. We denote by $\mathcal{T}$ the set of n time steps, by x_{it} the indicator variable for whether box i is opened at time t, and by z_{it}^s the indicator of whether box i is selected for scenario s at time t. We define the relaxation $\overline{g}^s(\boldsymbol{x})$ as

$$\min_{z \geq 0} \sum_{i \in \mathcal{B}, t \in \mathcal{T}} (t + c_i^s) z_{it}^s \qquad \text{(Relaxation-SPA)}$$

$$\text{s.t.} \sum_{t \in \mathcal{T}, i \in \mathcal{B}} z_{it}^s = 1,$$

$$z_{it}^s \leq x_{it}, \qquad i \in \mathcal{B}, t \in \mathcal{T}.$$

Similarly, we also relax the problem when we are required to pick k boxes (Relaxation-SPA-k) and when we are required to pick a matroid basis (Relaxation-SPA-matroid).

Leveraging the results of (Chawla et al., 2020), in sections C.1, C.2 and C.3 of the appendix, we show how to use a rounding that does not depend on the scenario chosen in order to get an approximately optimal integer solution, given one for the relaxation. Specifically, we define the notion of α-approximate rounding.

Definition 2 (α-approximate rounding). *Let Π be a minimization problem over a domain X with $f : X \to \mathbb{R}$ as its objective function and a convex relaxation $\overline{f} : \overline{X} \to \mathbb{R}$. Let $\overline{x} \in \overline{X}$ be a solution to Π with cost $\overline{f}(\overline{x})$. Then an α-approximate rounding is a an algorithm that given $\overline{x}$ produces a solution $x \in X$ with cost*

$$f(x) \leq \alpha \overline{f}(\overline{x})$$

3. Full Information Setting

We begin by presenting a general technique for approaching PANDORA'S BOX type of problems via Online Convex Optimization (OCO). Initially we observe, in the following theorem, that we can combine

1. a rounding algorithm with good approximation guarantees,

2. an online minimization algorithm with good regret guarantees

to obtain an algorithm with good regret guarantee.

Theorem 3.1. *Let Π be a minimization problem over a domain X and $\overline{\Pi}$ be the convex relaxation of Π over convex domain $\overline{X} \supseteq X$.*

If there exists an α-approximate rounding algorithm $\mathcal{A} : \overline{X} \to X$ for any feasible solution $\overline{x} \in \overline{X}$ to a feasible solution $x \in X$ then, any online minimization algorithm for $\overline{\Pi}$ that achieves regret $Regret(T)$ against a benchmark OPT, gives α-approximate regret $\alpha Regret(T)$ for Π.

Proof of Theorem 3.1. Let $f_1, ..., f_T$ be the online sequence of functions presented in problem Π, in each round $t \in [T]$, and let $\overline{f}_1, ..., \overline{f}_T$ be their convex relaxations in $\overline{\Pi}$.

Let $\overline{x}_t \in \overline{X}$ be the solution the online convex optimization algorithm gives at each round $t \in [T]$ for problem $\overline{\Pi}$. Calculating the total expected cost of Π, for all time steps $t \in [T]$ we have that

$$
\mathbb{E}\left[\sum_{t=1}^{T} f_t\big(\mathcal{A}(\overline{x}_t)\big)\right] \leq \alpha \sum_{t=1}^{T} \overline{f}_t(\overline{x}_t)
$$

$$
\leq \alpha \left(Regret(T) + \min_{x \in \overline{X}} \sum_{t=1}^{T} \overline{f}_t(x)\right)
$$

$$
\leq \alpha \left(Regret(T) + \min_{x \in X} \sum_{t=1}^{T} f_t(x)\right).
$$

By rearranging the terms, we get the theorem. $\square$

Given this theorem, in the following sections we show (1) how to design an algorithm with a low regret guarantee for PANDORA'S BOX (Theorem 3.3) and (2) how to obtain rounding algorithms with good approximation guarantees, using the results of (Chawla et al., 2020).

3.1. Applications to PANDORA'S BOX and MSSC

Applying Theorem 3.1 to our problems, in their initial non-convex form, we are required to pick an integer permutation of boxes. The relaxations, for the different benchmarks and constraints, are shown in Relaxation-SPA, Relaxation-SPA-k and Relaxation-SPA-matroid.

We denote by $\overline{g}^s(\boldsymbol{x})$ the objective function of the scenario aware relaxation of the setting we are trying to solve e.g for selecting 1 box we have Relaxation-SPA. Denote by $\overline{X} = [0,1]^{n \times n}$ the solution space. We can view this problem as an online convex optimization one as follows.

1. At every time step t we pick a vector $\boldsymbol{x}_t \in \overline{X}$, where $\overline{X}$ is a convex set.

2. The adversary picks a scenario $s \in \mathcal{S}$ and therefore a function $f^s : \overline{X} \to \mathbb{R}$ where $f^s = \overline{g}^s$ and we incur

---

Algorithm 1: Algorithm $\mathcal{A}$ for the full information case.

Input: $\Pi = (\mathcal{F}, \text{OPT})$: the problem to solve, $\mathcal{A}_\Pi$: the rounding algorithm for Π

1   Denote by $f^s(\boldsymbol{x}) =$ fractional objective function
2   Select regularizer $U(\boldsymbol{x})$ according to Theorem 3.3
3   $\overline{X} =$ space of fractional solutions
4   **for** *Each round $t \in [T]$* **do**
5   Set $\boldsymbol{x}_t = \min_{\boldsymbol{x} \in \overline{X}} \sum_{\tau=1}^{t-1} f^{s_\tau}(\boldsymbol{x}) + U(\boldsymbol{x})$
6   Round $\boldsymbol{x}_t$ to $\boldsymbol{x}_t^{\text{int}}$ according to $\mathcal{A}_\Pi$
7   Receive loss $f^s(\boldsymbol{x}_t^{\text{int}})$
8   **end**

---

loss $f^s(\boldsymbol{x}_t) = \overline{g}^s(\boldsymbol{x}_t)$. Note that f^s is convex in all cases (Relaxation-SPA, Relaxation-SPA-k, Relaxation-SPA-matroid).

3. We observe the function f^s for all points $\boldsymbol{x} \in \overline{X}$.

A family of algorithms that can be applied to solve this problem is called *Follow The Regularized Leader (FTRL)*. These algorithms work by picking, at every step, the solution that would have performed best so far while also adding a regularization term for stability. For the *FTRL* family of algorithms we have the following guarantees.

Theorem 3.2 (Theorem 2.11 from (Shalev-Shwartz, 2012))**.** *Let $f_1, \ldots, f_T$ be a sequence of convex functions such that each f_t is L-Lipschitz with respect to some norm. Assume that FTRL is run on the sequence with a regularization function U which is η-strongly-convex with respect to the same norm. Then, for all $u \in C$ we have that $Regret(FTRL, T) \cdot T \leq U_{\max} - U_{\min} + TL^2\eta$*

Our algorithm works similarly to *FTRL*, while additionally rounding the fractional solution, in each step, to an integer one. The algorithm is formally described in Algorithm 1, and we show how to choose the regularizer $U(\boldsymbol{x})$ in Theorem 3.3.

We show the guarantees of our algorithm above using Theorem 3.2 which provides regret guarantees for *FTRL*. The proof of Theorem 3.3 is deferred to section A of the appendix.

Theorem 3.3. *The average regret of Algorithm 1 is*

$$
Regret_{PA}(\mathcal{A}, T) \leq 2n\sqrt{\frac{\log n}{T}}
$$

achieved by setting $U(\boldsymbol{x}) = \left(\sum_{i=1}^{n} \sum_{t=1}^{n} x_{it} \log x_{it}\right)/\eta$ as the regularization function, and $\eta = \sqrt{\frac{\log n}{T}}$.

Finally, using Theorem 3.1 we get Corollary 3.3.1 for competing with the partially-adaptive benchmark for all different

feasibility constraints (choose 1, choose k or choose a matroid basis), where we use the results of Corollary C.0.1, to obtain the guarantees for the rounding algorithms.

Corollary 3.3.1 (Competing against PA, full information). *In the full information setting, Algorithm 1 is*

- *9.22-approximate no regret for **choosing** 1 **box***

- $O(1)$-*approximate no regret for **choosing** k **boxes***

- $O(\log k)$-*approximate no regret for **choosing a matroid basis***

Remark 3.3.1. *In the special case of MSSC, our approach obtains the tight 4-approximation of the offline case (Feige et al., 2004). The details of this are deferred to section C.1 of the Appendix. This result improves on the previous work (Fotakis et al., 2020) who obtain a 11.713-approximation.*

4. Bandit Setting

Moving on to a bandit setting for our problem, where we do not observe the whole function after each step. Specifically, after choosing $\boldsymbol{x}_t \in \overline{X}$ in each round t, we only observe a loss $f^s(\boldsymbol{x}_t)$ at the point $\boldsymbol{x}_t$ we chose to play and not for every $\boldsymbol{x} \in \overline{X}$. This difference prevents us from directly using any online convex optimization algorithm, as in the full information setting of section 3. However, observe that if we decide to open all n boxes, this is equivalent to observing the function f^s for all $\boldsymbol{x} \in \overline{X}$, since we learn the cost of all permutations.

We exploit this similarity by randomizing between running *FTRL* and paying n to open all boxes. Specifically we split $[T]$ into T/k intervals and choose a time, uniformly at random in each one, when we are going to open all boxes n and thus observe the function on all inputs. This process is formally described in Algorithm 2, and we show the following guarantees.

Theorem 4.1. *The average regret for Algorithm 2, for* $k = \left(\frac{n}{2L + \sqrt{\log n}}\right)^{2/3} T^{1/3}$ *and loss functions that are L-Lipschitz is*

$$\mathbb{E}\left[Regret_{PA}(\mathcal{A}_{PA}, T)\right] \le 2\left(2L \log n + n\right)^{2/3} \cdot n^{1/3} \cdot T^{-1/3}.$$

To analyze the regret of Algorithm 2 and prove Theorem 4.1, we consider the regret of two related settings.

1. In the first setting, we consider a full-information online learner that observes at each round t a single function sampled uniformly among the k functions of the corresponding interval $\mathcal{I}_t$. We call this setting *random costs*.

Algorithm 2: $\mathcal{A}_{PA}$ minimizing regret against PA

1. Get parameter k from Theorem 4.1
2. Select regularizer $U(\boldsymbol{x})$ according to Theorem 4.1
3. Split the times $[T]$ into T/k intervals $\mathcal{I}_1 \ldots, \mathcal{I}_{T/k}$
4. $\mathcal{R} \leftarrow \emptyset$ // Random times for each $\mathcal{I}_i$
5. **for** *Every interval $\mathcal{I}_i$* **do**
6.   Pick a t_p uniformly in $\mathcal{I}_i$
7.   **for** *All times $t \in \mathcal{I}_i$* **do**
8.   **if** $t = t_p$ **then**
9. $\mathcal{R} \leftarrow \mathcal{R} \cup \{t_p\}$
10.   Open all boxes
11.   Get feedback $f^{s_{t_p}}$
12.   **else**
13. $\boldsymbol{x}_t \leftarrow \text{argmin}_{\boldsymbol{x} \in \overline{X}} \sum_{\tau \in \mathcal{R}} f^{s_\tau}(\boldsymbol{x}) + U(\boldsymbol{x})$
14.   **end**
15.   **end**
16. **end**

---

2. In the second setting, we again consider a full-information online learner that observes at each round t a single function which is the average of the k functions in the corresponding interval $\mathcal{I}_t$. We call this setting *average costs*.

The following lemma, shows that any online algorithm for the random cost setting yields low regret even for the average costs setting.

Lemma 4.2. *Any online strategy for the* random costs *setting with expected average regret $R(T)$ gives expected average regret at most $R(T) + n/\sqrt{kT}$ for the equivalent average costs setting.*

Proof of Lemma 4.2. Denote by $\overline{f}_t = \frac{1}{k} \sum_{i=1}^{k} f_{t_i}$ the cost function corresponding to the average costs setting and by $f_t^r = f_{t_i}$ where $i \sim U([k])$ the corresponding cost function for the random costs setting. Let $x^* = \text{argmin}_{\boldsymbol{x} \in \overline{X}} \sum_{t=1}^{T/k} \overline{f}_t(\boldsymbol{x})$ be the minimizer of the $\overline{f}_t$ over the T/k rounds.

We also use $X_t = \overline{f}_t(\boldsymbol{x}_t) - f_t^r(\boldsymbol{x}_t)$, to denote the difference in costs between the two settings for each interval (where $\boldsymbol{x}_t$ is the action taken at each interval t by the random costs strategy). Observe that this is a random variable depending on the random choice of time in each interval. We have that

$$\mathbb{E}\left[\sum_{t=1}^{T/k} |X_t|\right] \le \left(\mathbb{E}\left[\left(\sum_{t=1}^{T/k} X_t\right)^2\right]\right)^{1/2}$$

$$= \left(\mathbb{E}\left[\sum_{t=1}^{T/k} X_t^2\right]\right)^{1/2}$$

$$\leq n\sqrt{\frac{T}{k}}.$$

The two inequalities follow by Jensen's inequality and the fact that X_t's are bounded by n. The equality is because the random variables X_t are martingales, i.e. $\mathbb{E}\left[X_t | X_1, ..., X_{t-1}\right] = 0$, as the choice of the function at time t is independent of the chosen point x_t.

We now look at the average regret of the strategy x_t for the average cost setting. We have that

$$\frac{1}{T}\mathbb{E}\left[\sum_t \overline{f}_t(x_t)\right] - R(T) - \frac{n}{\sqrt{kT}}$$

$$\leq \frac{1}{T}\mathbb{E}\left[\sum_t f_t^r(x_t)\right] - R(T)$$

$$\leq \frac{1}{T}\mathbb{E}\left[\min_x \sum_t f_t^r(x)\right]$$

$$\leq \frac{1}{T}\mathbb{E}\left[\sum_t f_t^r(x^*)\right]$$

$$= \sum_t \overline{f}_t(x^*)$$

which implies the required regret bound.

□

Given this lemma, we are now ready to show Theorem 4.1.

Proof of Theorem 4.1. To establish the result, we note that the regret of our algorithm is equal to the regret achievable in the average cost setting multiplied by k plus nT/k since we pay n for opening all boxes once in each of the T/k intervals. Using Lemma 4.2, it suffices to bound the regret in the random costs setting. Let $U(x) : [0,1]^{n \times n} \to \mathbb{R}$ be an η/n-strongly convex regularizer used in the *FTRL* algorithm. We are using $U(x) = \left(\sum_{i=1}^n \sum_{t=1}^n x_{it} \log x_{it}\right)/\eta$, which is η/n-strongly convex from Lemma A.1 and is at most $(n \log n)/\eta$ as we observed in corollary 3.3.1. Then from Theorem 3.2, we get that the average regret for the corresponding random costs setting is $2L\sqrt{\frac{\log n}{kT}}$.

Using Lemma 4.2, we get that the total average regret $R(T)$ of our algorithm is

$$R(T) \leq k \cdot 2L\sqrt{\frac{\log n}{kT}} + k \cdot n/\sqrt{kT} + \frac{n}{k}.$$

Setting $k = \left(\frac{n}{2L\log n + n}\right)^{2/3} T^{1/3}$ the theorem follows. □

Finally, using Theorem 4.1 we can get the same guarantees as the full-information setting, using the α-approximate rounding for each case (Corollary C.0.1).

Corollary 4.2.1 (Competing against PA, bandit). *In the bandit setting, Algorithm 2 is*

- *9.22-approximate no regret for* **choosing** 1 **box**

- $O(1)$-*approximate no regret for* **choosing** k **boxes**

- $O(\log k)$-*approximate no regret for* **choosing a matroid basis**

5. Competing with the Non-adaptive

We switch gears towards a different benchmark, that of the non-adaptive strategies. Similarly to the partially adaptive benchmark, here we we first present the linear programming for the non-adaptive benchmark as a function $\overline{f} : [0,1]^n \to \mathbb{R}$ with $\overline{f}(x)$ equal to

$$\min_{z \geq 0} \quad \sum_{i \in \mathcal{B}} x_i + \frac{1}{|\mathcal{S}|}\sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_i^s z_i^s \qquad \text{(LP-NA)}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{B}} z_i^s = 1, \qquad\qquad \forall s \in \mathcal{S}$$

$$z_i^s \leq x_i \qquad\qquad \forall i \in \mathcal{B}, s \in \mathcal{S}$$

where x_i is an indicator variable for whether box i is opened and z_i^s indicates whether box i is assigned to scenario s.

Note that the algorithms we provided for the partially-adaptive case cannot be directly applied since the objective functions of LP-NA, LP-NA-k and LP-NA-matroid are not n-Lipschitz. To achieve good regret bounds in this case, we design an algorithm that randomizes over an "explore" and an "exploit" step, similarly to (Alabi et al., 2019), while remembering the LP structure of the problem given constraints $\mathcal{F}$. Observe that there is a "global" optimal linear program (which is either LP-NA, LP-NA-k or LP-NA-matroid depending on the constraints $\mathcal{F}$) defined over all rounds T. Getting a new instance in each round is equivalent to receiving a new (hidden) set of constraints. We first describe two functions utilised by the algorithm in order to find a feasible fractional solution to the LP and to round it.

1. *Ellipsoid*$(k, \mathcal{LP})$: finds and returns a feasible solution to $\mathcal{LP}$ of cost at most k. By starting from a low k value and doubling at each step, lines 10-13 result in us finding a fractional solution within 2 every time.

2. *Round*$(S_t, \mathcal{F})$: rounds the fractional feasible solution S_t using the algorithm corresponding to $\mathcal{F}$. The rounding algorithms are presented in section D of the appendix. For selecting 1 box we have Algorithm 7, for selecting k boxes Algorithm 8 and for selecting a matroid basis Algorithm 9.

The algorithm works in two steps; in the "explore" step (line 7) opening all boxes results in us exactly learning

Algorithm 3: Algorithm $\mathcal{A}_{\mathcal{F}}$ for minimizing regret vs NA

---

1 **Input**: set of constraints $\mathcal{F}$
2 $\mathcal{LP} \leftarrow$ LP-NA or LP-NA-k or LP-NA-matroid (according to $\mathcal{F}$)
3 $\mathcal{C}_1 \leftarrow \emptyset$ // Constraints of LP
4 **for** *round* $t \in [T]$ **do**
5   draw $c \in U[0,1]$
6   **if** $c > p_t$ **then**
7   Open all n boxes, inducing new constraint c_{new}
8          $\mathcal{C}_{t+1} \leftarrow \mathcal{C}_t \cup \{c_{new}\}$
9          $k \leftarrow 1$
10   **repeat**
11              $(\boldsymbol{x}, \boldsymbol{z}) \leftarrow$ **Ellipsoid** $(k, \mathcal{LP})$
12              $k \leftarrow 2k$
13   **until** $(\boldsymbol{x}, \boldsymbol{z})$ *is feasible*;
14   **else**
15          $S_t \leftarrow S_{t-1}$
16          $\pi \leftarrow$ **Round**$(S_t, \mathcal{F})$
17   Open boxes according to order π
18   **end**
19 **end**

---

the hidden constraint of the current round, by paying n. The "exploit" step uses the information learned from the previous rounds to open boxes and choose one.

Observe that the cost of Algorithm 3 comes from three different cases, depending on what the result of the flip of the coin c is in each round.

1. If $c > p_t$, we and pay n for opening all boxes.

2. If $c < p_t$ and we pay cost proportional to the LP (we have a feasible solution).

3. If $c < p_t$ and we pay cost proportional to n (we did not have a feasible solution).

We bound term 3 using mistake bound, and then continue by bounding terms 1 and 2 to get the bound on total regret.

5.1. Bounding the mistakes

We start by formally defining what is *mistake bound* of an algorithm.

Definition 3 (Mistake Bound of Algorithm $\mathcal{A}$)**.** *Let $\mathcal{A}$ be an algorithm that solves problem Π and runs in $t \in [T]$ rounds with input x_t in each one. Then we define $\mathcal{A}$'s mistake bound*

as

$$\mathcal{M}(\mathcal{A}, T) = \mathbb{E}\left[\sum_{t=1}^{T} \mathbb{1}\{x_t \text{ not feasible for } \Pi\}\right]$$

where the expectation is taken over the algorithm's choices.

The main contribution in this section is the following lemma, that bounds the number of mistakes.

Lemma 5.1. *Algorithm 3 has mistake bound*

$$\mathcal{M}(\mathcal{A}_{\mathcal{F}}, T) \leq O(n^2 \sqrt{T}).$$

The mistake bound applies to all the different constraints $\mathcal{F}$ we consider. To achieve this, we leverage the fact that the ellipsoid algorithm, running on the optimal LP corresponding to the constraints $\mathcal{F}$, needs polynomial in n time to find a solution. The proof works by showing that every time, with probability p_t, we make progress towards the solution, and since the ellipsoid in total makes polynomial in n steps we also cannot make too many mistakes. The proof of Lemma 5.1 is deferred to section B of the appendix.

5.2. Regret for different constraints

Moving on to show regret guarantees of Algorithm 3 for the different types of constraints. We start off with the special case where we are required to pick one box, but all the costs inside the boxes are either 0 or ∞, and then generalize this to arbitrary costs and more complex constraints.

Theorem 5.2 (Regret for $0/\infty$)**.** *Algorithm 3, with $p_t = 1/\sqrt{T}$ has the following average regret, when $\mathcal{F} = \{Select\ 1\ box\}$ and $c_i \in \{0, \infty\}$.*

$$\mathbb{E}\left[Regret_{NA}(\mathcal{A}_{\mathcal{F}}, T)\right] \leq OPT + O\left(\frac{n^2}{\sqrt{T}}\right).$$

Proof of Theorem 5.2. Denote by M the mistake bound term, bounded above in Lemma 5.1. We calculate the total average regret

$$\mathbb{E}[\text{Regret}_{NA}(\mathcal{A}_{\mathcal{F}}, T)] + \text{OPT} = \frac{1}{T}\left(M + \sum_{t=1}^{T} \mathbb{E}\left[|S_t|\right]\right)$$

$$= \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + (1 - p_t)\mathbb{E}\left[|S_t|\right]\right)$$

$$\leq \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + 2\text{OPT}\right)$$

$$\leq M + 2\text{OPT} + n \sum_{t=1}^{T} p_t$$

$$\leq 2\text{OPT} + O\left(\frac{n^2}{\sqrt{T}}\right)$$

where initially we summed up the total regret of Algorithm 3 where the first term is the mistake bound from Lemma 5.1. Then we used the fact that $\text{OPT}_t \leq \text{OPT}$ and the solution found by the ellipsoid is within 2, and in the last line we used Since $\sum_{t=1}^{T} p_t \leq \sqrt{T}$ from (Alabi et al., 2019). Finally, subtracting OPT from both sides we get the theorem. □

Generalizing this to arbitrary values $c_i \in \mathbb{R}$, we show that when we are given a β approximation algorithm we get the following guarantees, depending on the approximation factor.

Theorem 5.3. *If there exists a partially adaptive algorithm $\mathcal{A}_{\mathcal{F}}$ that is β-competitive against the non-adaptive optimal, for the problem with constraints $\mathcal{F}$, then Algorithm 3, with $p_t = 1/\sqrt{T}$ has the following regret.*

$$\mathbb{E}\left[Regret_{NA}(\mathcal{A}_{\mathcal{F}}, T)\right] \leq 2\beta OPT + O\left(\frac{n^2}{\sqrt{T}}\right).$$

The proof follows similarly to the $0/\infty$ case, and is deferred to section B of the appendix. Combining the different guarantees against the non-adaptive benchmark with Theorem 5.3 we get the following corollary.

Corollary 5.3.1 (Competing against NA, bandit setting). *In the bandit setting, when competing with the non-adaptive benchmark, Algorithm 3 is*

- 3.16-*approximate no regret for **choosing** 1 **box** (using Theorem D.1)*

- 12.64-*approximate no regret for **choosing** k **boxes** (using Theorem D.3)*

- $O(\log k)$-*approximate no regret for **choosing a matroid basis** (using Theorem D.5)*

References

Adamczyk, M., Sviridenko, M., and Ward, J. Submodular stochastic probing on matroids. *Math. Oper. Res.*, 41(3):1022–1038, 2016. doi: 10.1287/moor.2015.0766. URL https://doi.org/10.1287/moor.2015.0766.

Agarwal, A., Dekel, O., and Xiao, L. Optimal algorithms for online convex optimization with multi-point bandit feedback. In Kalai, A. T. and Mohri, M. (eds.), *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*, pp. 28–40. Omnipress, 2010. URL http://colt2010.haifa.il.ibm.com/papers/COLT2010proceedings.pdf#page=36.

Ailon, N., Chazelle, B., Comandur, S., and Liu, D. Self-improving algorithms. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*, pp. 261–270, 2006. URL http://dl.acm.org/citation.cfm?id=1109557.1109587.

Alabi, D., Kalai, A. T., Ligett, K., Musco, C., Tzamos, C., and Vitercik, E. Learning to prune: Speeding up repeated computations. In Beygelzimer, A. and Hsu, D. (eds.), *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, volume 99 of *Proceedings of Machine Learning Research*, pp. 30–33. PMLR, 2019. URL http://proceedings.mlr.press/v99/alabi19a.html.

Azar, Y., Gamzu, I., and Yin, X. Multiple intents re-ranking. In Mitzenmacher, M. (ed.), *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pp. 669–678. ACM, 2009. doi: 10.1145/1536414.1536505. URL https://doi.org/10.1145/1536414.1536505.

Balcan, M., Nagarajan, V., Vitercik, E., and White, C. Learning-theoretic foundations of algorithm configuration for combinatorial partitioning problems. In *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, pp. 213–274, 2017. URL http://proceedings.mlr.press/v65/balcan17a.html.

Balcan, M., Dick, T., Sandholm, T., and Vitercik, E. Learning to branch. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pp. 353–362, 2018a. URL http://proceedings.mlr.press/v80/balcan18a.html.

Balcan, M., Dick, T., and Vitercik, E. Dispersion for data-driven algorithm design, online learning, and private optimization. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pp. 603–614, 2018b. doi: 10.1109/FOCS.2018.00064. URL https://doi.org/10.1109/FOCS.2018.00064.

Bansal, N., Gupta, A., and Krishnaswamy, R. A constant factor approximation algorithm for generalized min-sum set cover. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pp. 1539–1545, 2010. doi: 10.1137/1.9781611973075.125. URL https://doi.org/10.1137/1.9781611973075.125.

Beyhaghi, H. and Kleinberg, R. Pandora's problem with nonobligatory inspection. In Karlin, A., Immorlica, N., and Johari, R. (eds.), *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019,*

Phoenix, AZ, USA, June 24-28, 2019, pp. 131–132. ACM, 2019. doi: 10.1145/3328526.3329626. URL https://doi.org/10.1145/3328526.3329626.

Boodaghians, S., Fusco, F., Lazos, P., and Leonardi, S. Pandora's box problem with order constraints. In Biró, P., Hartline, J. D., Ostrovsky, M., and Procaccia, A. D. (eds.), *EC '20: The 21st ACM Conference on Economics and Computation, Virtual Event, Hungary, July 13-17, 2020*, pp. 439–458. ACM, 2020. doi: 10.1145/3391403.3399501. URL https://doi.org/10.1145/3391403.3399501.

Charikar, M., Fagin, R., Guruswami, V., Kleinberg, J. M., Raghavan, P., and Sahai, A. Query strategies for priced information (extended abstract). In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pp. 582–591, 2000. doi: 10.1145/335305.335382. URL https://doi.org/10.1145/335305.335382.

Chawla, S., Gergatsouli, E., Teng, Y., Tzamos, C., and Zhang, R. Pandora's box with correlations: Learning and approximation. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pp. 1214–1225. IEEE, 2020. doi: 10.1109/FOCS46700.2020.00116. URL https://doi.org/10.1109/FOCS46700.2020.00116.

Chen, Y., Hassani, S. H., Karbasi, A., and Krause, A. Sequential information maximization: When is greedy near-optimal? In *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, pp. 338–363, 2015a. URL http://proceedings.mlr.press/v40/Chen15b.html.

Chen, Y., Javdani, S., Karbasi, A., Bagnell, J. A., Srinivasa, S. S., and Krause, A. Submodular surrogates for value of information. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pp. 3511–3518, 2015b. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9841.

Clarkson, K. L., Mulzer, W., and Seshadhri, C. Self-improving algorithms for convex hulls. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pp. 1546–1565, 2010. doi: 10.1137/1.9781611973075.126. URL https://doi.org/10.1137/1.9781611973075.126.

Correa, J. R., Foncea, P., Hoeksma, R., Oosterwijk, T., and Vredeveld, T. Recent developments in prophet inequalities. *SIGecom Exch.*, 17(1):61–70, 2018. doi: 10.1145/3331033.3331039. URL https://doi.org/10.1145/3331033.3331039.

Doval, L. Whether or not to open pandora's box. *J. Econ. Theory*, 175:127–158, 2018. doi: 10.1016/j.jet.2018.01.005. URL https://doi.org/10.1016/j.jet.2018.01.005.

Esfandiari, H., Hajiaghayi, M. T., Lucier, B., and Mitzenmacher, M. Online pandora's boxes and bandits. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 1885–1892. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33011885. URL https://doi.org/10.1609/aaai.v33i01.33011885.

Feige, U., Lovász, L., and Tetali, P. Approximating min sum set cover. *Algorithmica*, 40(4):219–234, 2004. doi: 10.1007/s00453-004-1110-5. URL https://doi.org/10.1007/s00453-004-1110-5.

Flaxman, A., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pp. 385–394. SIAM, 2005. URL http://dl.acm.org/citation.cfm?id=1070432.1070486.

Fotakis, D., Lianeas, T., Piliouras, G., and Skoulakis, S. Efficient online learning of optimal rankings: Dimensionality reduction via gradient descent. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Goel, A., Guha, S., and Munagala, K. Asking the right questions: model-driven optimization using probes. In *Proceedings of the Twenty-Fifth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 26-28, 2006, Chicago, Illinois, USA*, pp. 203–212, 2006. doi: 10.1145/1142351.1142380. URL https://doi.org/10.1145/1142351.1142380.

Grötschel, M., Lovász, L., and Schrijver, A. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. ISBN 978-3-642-97883-8. doi: 10.1007/978-3-642-97881-4. URL https://doi.org/10.1007/978-3-642-97881-4.

Gupta, A. and Kumar, A. Sorting and selection with structured costs. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pp. 416–425, 2001. doi: 10.1109/SFCS.2001.959916. URL https://doi.org/10.1109/SFCS.2001.959916.

Gupta, A. and Nagarajan, V. A stochastic probing problem with applications. In *Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, pp. 205–216, 2013. doi: 10.1007/978-3-642-36694-9_18. URL https://doi.org/10.1007/978-3-642-36694-9_18.

Gupta, A., Nagarajan, V., and Singla, S. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pp. 1731–1747, 2016. doi: 10.1137/1.9781611974331.ch120. URL https://doi.org/10.1137/1.9781611974331.ch120.

Gupta, A., Nagarajan, V., and Singla, S. Adaptivity gaps for stochastic probing: Submodular and XOS functions. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pp. 1688–1702, 2017. doi: 10.1137/1.9781611974782.111. URL https://doi.org/10.1137/1.9781611974782.111.

Gupta, A., Jiang, H., Scully, Z., and Singla, S. The markovian price of information. In *Integer Programming and Combinatorial Optimization - 20th International Conference, IPCO 2019, Ann Arbor, MI, USA, May 22-24, 2019, Proceedings*, pp. 233–246, 2019. doi: 10.1007/978-3-030-17953-3_18. URL https://doi.org/10.1007/978-3-030-17953-3_18.

Gupta, R. and Roughgarden, T. A PAC approach to application-specific algorithm selection. *SIAM J. Comput.*, 46(3):992–1017, 2017. doi: 10.1137/15M1050276. URL https://doi.org/10.1137/15M1050276.

Hill, T. P. and Kertz, R. P. A survey of prophet inequalities in optimal stopping theory. *CONTEMPORARY MATHEMATICS*, 125, 1992.

Karlin, A. R., Manasse, M. S., McGeoch, L. A., and Owicki, S. S. Competitive randomized algorithms for non-uniform problems. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1990, San Francisco, California, USA.*, pp. 301–309, 1990. URL http://dl.acm.org/citation.cfm?id=320176.320216.

Kleinberg, R., Leyton-Brown, K., and Lucier, B. Efficiency through procrastination: Approximately optimal algorithm configuration with runtime guarantees. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 2023–2031, 2017. doi: 10.24963/ijcai.2017/281. URL https://doi.org/10.24963/ijcai.2017/281.

Lucier, B. An economic view of prophet inequalities. *SIGecom Exch.*, 16(1):24–47, 2017. doi: 10.1145/3144722.3144725. URL https://doi.org/10.1145/3144722.3144725.

Munagala, K., Babu, S., Motwani, R., and Widom, J. The pipelined set cover problem. In Eiter, T. and Libkin, L. (eds.), *Database Theory - ICDT 2005, 10th International Conference, Edinburgh, UK, January 5-7, 2005, Proceedings*, volume 3363 of *Lecture Notes in Computer Science*, pp. 83–98. Springer, 2005. doi: 10.1007/978-3-540-30570-5_6. URL https://doi.org/10.1007/978-3-540-30570-5_6.

Shalev-Shwartz, S. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 4(2):107–194, 2012. doi: 10.1561/2200000018. URL https://doi.org/10.1561/2200000018.

Shalev-Shwartz, S. and Singer, Y. A primal-dual perspective of online learning algorithms. *Mach. Learn.*, 69(2-3):115–142, 2007. doi: 10.1007/s10994-007-5014-x. URL https://doi.org/10.1007/s10994-007-5014-x.

Singla, S. The price of information in combinatorial optimization. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pp. 2523–2532, 2018. doi: 10.1137/1.9781611975031.161. URL https://doi.org/10.1137/1.9781611975031.161.

Skutella, M. and Williamson, D. P. A note on the generalized min-sum set cover problem. *Oper. Res. Lett.*, 39(6):433–436, 2011. doi: 10.1016/j.orl.2011.08.002. URL https://doi.org/10.1016/j.orl.2011.08.002.

Weisz, G., Gyorgy, A., and Szepesvari, C. Leapsandbounds: A method for approximately optimal algorithm configuration. In *International Conference on Machine Learning*, pp. 5257–5265, 2018.

Weitzman, M. L. Optimal Search for the Best Alternative. *Econometrica*, 47(3):641–654, May 1979.

A. Proofs from section 3

The following lemma shows the strong convexity of the regularizer used in our *FTRL* algorithms.

Lemma A.1 (Convexity of Regularizer). *The following function is $1/n$-strongly convex with respect to the ℓ_1-norm.*

$$U(\boldsymbol{x}) = \sum_{i=1}^{n} \sum_{t=1}^{n} x_{it} \log x_{it}$$

for a doubly-stochastic matrix $x \in [0,1]^{n \times n}$

Proof. Since $U(\boldsymbol{x})$ is twice continuously differentiable we calculate $\nabla^2 U(\boldsymbol{x})$, which is a $n \times n$ diagonal matrix since

$$\frac{\vartheta U(\boldsymbol{x})}{\vartheta x_{kt} \vartheta x_{ij}} = \begin{cases} 1/x_{ij} & \text{If } i = k \text{ and } j = t \\ 0 & \text{Else} \end{cases}$$

We show that $\boldsymbol{z} \nabla^2 U(\boldsymbol{x}) \boldsymbol{z} \geq ||\boldsymbol{z}||_1^2$ for all $\boldsymbol{x} \in \mathbb{R}^{n^2}$. We make the following mapping of the variables for each x_{ij} we map it to p_k where $k = (i-1)n + j$. We have that

$$\begin{aligned} \boldsymbol{z} \nabla^2 U(\boldsymbol{x}) \boldsymbol{z} &= \sum_{i=1}^{n^2} \frac{(z_i)^2}{p_i} \\ &= \frac{1}{n} \left(\sum_{i=1}^{n^2} p_i \right) \sum_{i=1}^{n^2} \frac{(z_i)^2}{p_i} \\ &\geq \frac{1}{n} \left(\sum_{i=1}^{n^2} \sqrt{p_i} \frac{|z_i|}{\sqrt{p_i}} \right)^2 \\ &= \frac{1}{n} ||\boldsymbol{z}||_1^2. \end{aligned}$$

where in the second line we used that x_{ij}'s are a doubly stochastic matrix, and then Cauchy-Schwartz inequality. □

Theorem 3.3. *The average regret of Algorithm 1 is*

$$Regret_{PA}(\mathcal{A}, T) \leq 2n \sqrt{\frac{\log n}{T}}$$

achieved by setting $U(\boldsymbol{x}) = \left(\sum_{i=1}^{n} \sum_{t=1}^{n} x_{it} \log x_{it} \right) / \eta$ as the regularization function, and $\eta = \sqrt{\frac{\log n}{T}}$.

Proof. Initially observe that by setting $x_{ij} = 1/n$ we get $U_{\max} - U_{\min} = (n \log n)/\eta$, since we get the maximum entropy when the values are all equal. Additionally, from Lemma A.1 we have that $U(\boldsymbol{x})$ is η/n-strongly convex. Observing also that the functions in all cases are n-Lipschitz and using Theorem 3.2 we obtain the guarantee of the theorem, by setting $\eta = \frac{\sqrt{\log n}}{\sqrt{T}}$. □

B. Proofs from section 5

Before moving to the formal proof of Lemma 5.1, we recall the following lemma about the ellipsoid algorithm, bounding the number of steps it takes to find a feasible solution.

Lemma B.1 (Lemma 3.1.36 from (Grötschel et al., 1988)). *Given a full dimensional polytope $P = \{x : Cx \leq d\}$, for $x \in \mathbb{R}^n$, and let $\langle C, d \rangle$ be the encoding length of C and d. If the initial ellipsoid is $E_0 = E(R^2 I, 0)^3$ where $R = \sqrt{n} 2^{\langle C, d \rangle - n^2}$ the ellipsoid algorithm finds a feasible solution after $O(n^2 \langle C, d \rangle)$ steps.*

---

[3] $E(R, Z)$ indicates a ball of radius R and center Z.

Using the lemma above, we can now prove Lemma 5.1, which we also restate below.

Lemma 5.1. *Algorithm 3 has mistake bound*

$$\mathcal{M}(\mathcal{A}_\mathcal{F}, T) \leq O(n^2 \sqrt{T}).$$

Proof. Our analysis follows similarly to Theorem 3.2 of (Alabi et al., 2019). Initially observe that the only time we make a mistake is in the case with probability $(1 - p_t)$ if the LP solution is not feasible. Denote by $\mathcal{C}^*$ the set of the constraints of $\mathcal{LP}$ as defined in Algorithm 3, and by $\mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \ldots \subseteq \mathcal{C}_t$ the constraint set for every round of the algorithm, for all $t \in [T]$. We also denote by $N_T(c)$ the number of times a constraint c was not in $\mathcal{C}_t$ for some time t but was part of $\mathcal{LP}$. Formally $N_T(c) = |\{c \in \mathcal{C}^*, c \notin \mathcal{C}_t, \}|$ for a constraint $c \in \mathcal{C}^*$ and any $t \in [T]$. We can bound the mistake bound of Algorithm 3 as follows

$$\mathcal{M}(\mathcal{A}, T) \leq \sum_{c \in \mathcal{C}^*} \mathbb{E}\left[N_T(c)\right].$$

Let $t_c \in [T]$ be the round that constraint c is added to the algorithm's constraint set for the first time, and let S_c be the set of ℓ rounds in which we made a mistake because of this constraint. Observe that $\{S_1, S_2, \ldots S_\ell\} = S_c \subseteq \{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_{t_c}\}$. We calculate the probability that $N_T(c)$ is incremented on round k of S

$$\mathbf{Pr}\left[N_T(c) \text{ incremented on round } k\right] = \prod_{i=1}^{k} (1 - p_i) \leq (1 - p)^k,$$

since in order to make a mistake, we ended up on line 14 of the algorithm. Therefore

$$\mathbb{E}\left[N_T(c)\right] \leq \sum_{i=1}^{T} (1 - p)^i = \frac{(1 - p)(1 - (1 - p))^T}{p}.$$

However in our case, every time a constraint is added to $\mathcal{C}_t$, one step of the ellipsoid algorithm is run, for the $\mathcal{LP}$. Using Lemma B.1 and observing that in our case $\langle C, d \rangle = O(1)$ the total times this step can happen is at most $O(n^2)$, giving us the result of the lemma by setting $p = 1/\sqrt{T}$. □

Theorem 5.3. *If there exists a partially adaptive algorithm $\mathcal{A}_\mathcal{F}$ that is β-competitive against the non-adaptive optimal, for the problem with constraints $\mathcal{F}$, then Algorithm 3, with $p_t = 1/\sqrt{T}$ has the following regret.*

$$\mathbb{E}\left[Regret_{NA}(\mathcal{A}_\mathcal{F}, T)\right] \leq 2\beta OPT + O\left(\frac{n^2}{\sqrt{T}}\right).$$

Proof of Theorem 5.3. Denote by M be the mistake bound term, bounded in Lemma 5.1. Calculating the total average regret we get

$$
\begin{aligned}
\mathbb{E}\left[\text{Regret}_{NA}(\mathcal{A}_\mathcal{F}, T)\right] + \text{OPT} &= \frac{1}{T}\left(M + \sum_{t=1}^{T} \mathbb{E}\left[S_t\right]\right) && \text{Definition} \\
&= \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t(n + \min_{i \in \mathcal{F}} c_i) + (1 - p_t)\mathbb{E}\left[S_t\right]\right) && \text{Algorithm 3} \\
&\leq \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + p_t \min_{i \in \mathcal{F}} c_i + 2\beta\text{OPT}_t\right) && \mathcal{A}_\mathcal{F}, \text{ and ellipsoid's loss} \\
&\leq (2\beta + 1)\text{OPT} + \frac{1}{T}\left(M + n \sum_{t=1}^{T} p_t\right) && \min_{i \in \mathcal{F}} c_i \leq \text{OPT}_t \leq \text{OPT} \\
&\leq (2\beta + 1)\text{OPT} + \frac{n}{\sqrt{T}} && \sum_{t=1}^{T} p_t \leq \sqrt{T} \text{ from (Alabi et al., 2019)} \\
&\leq (2\beta + 1)\text{OPT} + O\left(\frac{n^2}{\sqrt{T}}\right) && \text{From Lemma 5.1.}
\end{aligned}
$$

Therefore, subtracting OPT from both sides we get the theorem. □

C. Rounding against Partially-adaptive

In this section we show how using the rounding algorithms presented in (Chawla et al., 2020), we obtain α-approximate rounding for our convex relaxations. We emphasize the fact that these algorithms convert a fractional solution of the relaxed problem, to an integer solution of comparable cost, without needing to know the scenario. Formally we show the following guarantees.

Corollary C.0.1 (Rounding against PA). *Given a fractional solution $\overline{x}$ of cost $\overline{f}(\overline{x})$ there exists an α-approximate rounding where*

- *Selecting 1 box: $\alpha = 9.22$ (Using Lemma C.1)*

- *Selecting k boxes: $\alpha = O(1)$ (Using Lemma C.2)*

- *Selecting a matroid basis: $\alpha = O(\log k)$ (Using Lemma C.4)*

In order to obtain the results of this corollary, we combine the ski-rental Lemma 2.1 with Lemmas C.1, C.2 and C.4, for each different case of constraints. Observe that, as we show in the following sections, the part where the fractional solution given is rounded to an integer permutation **does not depend on the scenario realized**. Summarizing the rounding framework of the algorithms is as follows.

1. Receive fractional solution $\overline{x}, \overline{z}$.

2. Use rounding 4, 5, 6, depending on the constraints, to obtain an (integer) permutation π of the boxes.

3. Start opening boxes in the order of π.

4. Use ski-rental to decide the stopping time.

C.1. Rounding against Partially-adaptive for Choosing 1 Box

The convex relaxation for this case (Relaxation-SPA) is also given in section 2.2.2, but we repeat it here for convenience.

$$\min_{z \geq 0} \quad \sum_{i \in \mathcal{B}, t \in \mathcal{T}} (t + c_i^s) z_{it}^s \qquad \text{(Relaxation-SPA)}$$

$$\text{s.t.} \quad \sum_{t \in \mathcal{T}, i \in \mathcal{B}} z_{it}^s = 1,$$

$$z_{it}^s \leq x_{it}, \qquad\qquad i \in \mathcal{B}, t \in \mathcal{T}.$$

Our main lemma in this case, shows how to obtain a constant competitive partially adaptive strategy, when given a scenario-aware solution.

Lemma C.1. *Given a scenario-aware fractional solution $\overline{x}$ of cost $\overline{f}(\overline{x})$ there exists an efficient partially-adaptive strategy x with cost at most $9.22\overline{f}(\overline{x})$.*

Proof. We explicitly describe the rounding procedure, in order to highlight its independence of the scenario realized. For the rest of the proof we fix an (unknown) realized scenario s. Starting from our (fractional) solution $\overline{x}$ of cost $\overline{f}^s = \overline{f}_o^s + \overline{f}_c^s$, where $\overline{f}_o^s$ and $\overline{f}_c^s$ are the opening and values[4] cost respectively, we use the reduction in Theorem 5.2 in (Chawla et al., 2020) to obtain a transformed fractional solution $\overline{x}'$ of cost $\overline{f'}^s = \overline{f'}_o^s + \overline{f'}_c^s$. For this transformed solution, (Chawla et al., 2020) in Lemma 5.1 showed that

$$\overline{f'}_o^s \leq \left(\frac{\alpha}{\alpha - 1}\right)^2 \overline{f}_o^s \tag{3}$$

for the opening cost and

$$\overline{f'}_c^s \leq \alpha \overline{f}_c^s \tag{4}$$

---

[4]Cost incurred by the value found inside the box.

for the cost incured by the value inside the box chosen. To achieve this, the initial variables $\overline{x}_{it}$ are scaled by a factor depending on α to obtain $\overline{x}'_{it}$. For the remainder of the proof, we assume this scaling happened at the beginning, and abusing notation we denote by $\overline{x}$ the scaled variables. This is without loss of generality since, at the end of the proof, we are taking into account the loss in cost incurred by the scaling (Inequalities 3 and 4). The rounding process is shown in Algorithm 4.

Algorithm 4: Scenario aware, α-approximate rounding for 1 box

Data: Fractional solution x with cost $\overline{f}$, set $\alpha = 3 + 2\sqrt{2}$

`/* Part 1:  Scenario-independent rounding                                      */`

1   $\sigma :=$ for every $t = 1, \dots, n$, repeat **twice**: open each box i w.p. $q_{it} = \frac{\sum_{t' \le t} \overline{x}_{it'}}{t}$.

2

`/* Part 2:  Scenario-dependent stopping time                                   */`

3   Given scenario s, calculate z^s and $\overline{f}^s_c$

4   $\tau_s :=$ If box i is opened and has value $c^s_i \le \alpha \overline{f}^s_c$ then select it.

The ratio of the opening cost of the integer to the fractional solution is bounded by

$$\frac{f^s_o}{\overline{f'}^s_o} \le \frac{2\sum_{t=1}^{\infty} \prod_{k=1}^{t-1} \left(1 - \frac{\sum_{i \in A, t' \le k} z^s_{it'}}{k}\right)^2}{\sum_i t \cdot z^s_{it}} \qquad \text{Since } z^s_{it} \le x_{it}$$

$$\le \frac{2\sum_{t=1}^{\infty} \exp\left(-2\sum_{k=1}^{t-1} \frac{\sum_{t' \le k} z^s_{it'}}{k}\right)}{\sum_i t \cdot z^s_{it}} \qquad \text{Using that } 1 + x \le e^x$$

Observe that $h(z) = \log \frac{f^s_o}{\overline{f'}^s_o} = \log f^s_o - \log \overline{f}^s_o$ is a convex function since the first part is LogSumExp, and $\log \overline{f}^s_o$ is the negation of a concave function. That means $h(z)$ obtains the maximum value in the boundary of the domain, therefore at the integer points where $z^s_{it} = 1$ iff $t = \ell$ for some $\ell \in [n]$, otherwise $z^s_{it} = 0$. Using this fact we obtain

$$\frac{f^s_o}{\overline{f'}^s_o} \le \frac{2\ell + 2\sum_{t=\ell+1}^{\infty} \exp\left(-2\sum_{k=\ell}^{t-1} \frac{1}{k}\right)}{\ell} \qquad \text{Using that } z^s_{it} = 1 \text{ iff } t = \ell$$

$$= \frac{2\ell + 2\sum_{t=\ell+1}^{\infty} \exp\left(H_{t-1} - H_{\ell-1}\right)}{\ell} \qquad H_t \text{ is the } t\text{'th harmonic number}$$

$$\le \frac{2\ell + 2\sum_{t=\ell+1}^{\infty} \left(\frac{\ell}{t}\right)^2}{\ell} \qquad \text{Since } H_{t-1} - H_{\ell-1} \ge \int_\ell^t \frac{1}{x} dx = \log t - \log \ell$$

$$\le \frac{2\ell + 2\ell^2 \int_\ell^{\infty} \frac{1}{t^2} dt}{\ell} \qquad \text{Since } t^{-2} \le x^{-2} \text{ for } x \in [t-1, t]$$

$$= 4.$$

Combining with equation 3, we get that $f^s_o \le 4\left(\frac{\alpha}{\alpha-1}\right)^2 \overline{f}^s_o$. Recall that for the values cost, inequality (4) holds, therefore requiring that $4\left(\frac{\alpha}{\alpha-1}\right)^2 = \alpha$, we have the lemma for $\alpha = 3 + 2\sqrt{2}$. $\qquad \square$

Corollary C.1.1. *For the case of MSSC, when the costs inside the boxes are either 0 or ∞, the rounding of Lemma C.1 obtains a 4-approximation, improving the 11.473 of (Fotakis et al., 2020).*

C.2. Rounding against Partially-adaptive for Choosing k Boxes

In this case we are required to pick k boxes instead of one. Similarly to the one box case, we relax the domain $\overline{X} = \{x \in [0,1]^{n \times n} : \sum_i x_{it} = 1 \text{ and } \sum_t x_{it} = 1\}$, to be the set of doubly stochastic matrices. We define the relaxation $\overline{g}^s(x)$ as

$$\min_{y \ge 0, z \ge 0} \qquad \sum_{t \in \mathcal{T}} (1 - y^s_t) \quad + \sum_{i \in \mathcal{B}, t \in \mathcal{T}} c^s_i z^s_{it} \qquad \text{(Relaxation-SPA-k)}$$

$$
\begin{array}{rrll}
\text{subject to} & z_{it}^s & \leq & x_{it}, & \forall i \in \mathcal{B}, t \in \mathcal{T} \\
& \sum_{t' \leq t, i \notin A} z_{it'}^s & \geq & (k - |A|)y_t^s, & \forall A \subseteq \mathcal{B}, t \in \mathcal{T} \qquad (5)
\end{array}
$$

Our main lemma in this case, shows how to obtain a constant competitive partially adaptive strategy, when given a scenario-aware solution, in the case we are required to select k items.

Lemma C.2. *Given a scenario-aware fractional solution $\overline{z}^s, \overline{x}$ of cost $\overline{g}^s(\overline{x})$ there exists an efficient partially-adaptive strategy x with cost at most $O(1)\overline{g}^s(\overline{x})$.*

Proof. We follow exactly the steps of the proof of Theorem 6.2 from (Chawla et al., 2020), but here we highlight two important properties of it.

- The rounding part that decides the permutation (Algorithm 5) does **not** depend on the scenario realised, despite the algorithm being scenario-aware.

- The proof does not use the fact that the initial solution is an *optimal* LP solution. Therefore, the guarantee is given against **any** feasible fractional solution.

The rounding process is shown in Algorithm 5.

Algorithm 5: Scenario aware, α-approximate rounding for k-coverage from (Chawla et al., 2020)

Data: Solution $\boldsymbol{x}$ to Relaxation-SPA-k. Set $\alpha = 8$

```
/* Part 1:  Scenario-independent rounding                                */
```
1   $\sigma :=$ For each phase $\ell = 1, 2, \ldots$, open each box i independently with probability $q_{i\ell} = \min\left(\alpha \sum_{t \leq 2^\ell} x_{it}, 1\right)$.

2

```
/* Part 2:  Scenario-dependent stopping time                            */
```
3   $\tau_s :=$
4   Given scenario s, calculate $\boldsymbol{y}^s, \boldsymbol{z}^s$
5   Define $t_s^* = \max\{t : y_t^s \leq 1/2\}$.
6   **if** $2^\ell \geq t_s^*$ **then**
7   For each opened box i, select it with probability $\min\left(\frac{\alpha \sum_{t \leq 2^\ell} z_{it}^s}{q_{i\ell}}, 1\right)$.
8   Stop when we have selected k boxes in total.
9   **end**

Assume that we are given a fractional solution $(\overline{\boldsymbol{x}}, \overline{\boldsymbol{y}}^s, \overline{\boldsymbol{z}}^s)$, where $\overline{x}_{it}$ is the fraction that box i is opened at time t, $\overline{z}_{it}^s$ is the fraction box i is chosen for scenario s at time t, and $\overline{y}_t^s$ is the fraction scenario s is "covered" at time t, where covered means that there are k boxes selected for this scenario[5]. Denote by $\overline{f}_o^s$ (f_o^s) and $\overline{f}_c^s$ (f_c^s) the fractional (rounded) costs for scenario s due to opening and selecting boxes respectively. Denote also by t_s^* the last time step that $y_t^s \leq 1/2$ and observe that

$$
\overline{f}_o^s \geq \frac{t_s^*}{2}. \qquad (6)
$$

Fix a realized scenario s and denote by $\ell_0 = \lceil \log t_s^* \rceil$. Using that for each box i the probability that it is selected in phase $\ell \geq \ell_0$ is $\min(1, 8 \sum_{t' \leq 2^\ell} z_{it'}^s)$, we use the following lemma from (Bansal et al., 2010) that still holds in our case ; the proof of the lemma only uses constraint 5 and a Chernoff bound.

Lemma C.3 (Lemma 5.1 in (Bansal et al., 2010))**.** *If each box i is selected w.p. at least $\min(1, 8 \sum_{t' \leq t} z_{it'}^s)$ for $t \geq t_s^*$, then with probability at least $1 - e^{-9/8}$, at least k different boxes are selected.*

---
[5]We use the variables y_t^s for convenience, they are not required since $y_t^s = \sum_{t' < t, i \in \mathcal{B}} z_{it}^s$.

Similarly to (Chawla et al., 2020), let $\gamma = e^{-9/8}$ and $\mathcal{B}_j$ be the set of boxes selected at phase j. Since the number of boxes opened in a phase is independent of the event that the algorithm reaches that phase prior to covering scenario s the expected inspection cost is

$$
\begin{aligned}
\mathbb{E}\left[f_o^s \text{ after phase } \ell_0\right] &= \sum_{\ell=\ell_0}^{\infty} \mathbb{E}\left[f_o^s \text{ in phase } \ell\right] \cdot \mathbf{Pr}\left[\text{reach phase } \ell\right] \\
&\leq \sum_{\ell=\ell_0}^{\infty} \sum_{i\in\mathcal{B}} \alpha \sum_{t'\leq 2^\ell} x_{it'} \cdot \prod_{j=\ell_0}^{\ell-1} \mathbf{Pr}\left[|\mathcal{B}_j| \leq k\right] \\
&\leq \sum_{\ell=\ell_0}^{\infty} 2^\ell \alpha \cdot \gamma^{\ell-\ell_0} && \text{Lemma C.3, } x_{it} \text{ doubly stochastic} \\
&= \frac{2^{\ell_0}\alpha}{1-2\gamma} < \frac{2t_s^*\alpha}{1-2\gamma} \leq \frac{4\alpha\overline{f}_o^s}{1-2\gamma}. && \ell_0 = \lceil \log t_s^* \rceil \text{ and ineq. (6)}
\end{aligned}
$$

Observe that the expected opening cost at each phase ℓ is at most $\alpha 2^\ell$, therefore the expected opening cost before phase ℓ_0 is at most $\sum_{\ell<\ell_0} \alpha 2^\ell < 2^{\ell_0}\alpha < 2t_s^*\alpha \leq 4\alpha\overline{f}_o^s$. Putting it all together, the total expected opening cost of the algorithm for scenario s is

$$
f_o^s \leq 4\alpha\overline{f}_o^s + \frac{4\alpha\overline{f}_o^s}{1-2\gamma} < 123.25\overline{f}_o^s.
$$

To bound the cost of our algorithm, we find the expected total value of any phase ℓ, conditioned on selecting at least k distinct boxes in this phase.

$$
\begin{aligned}
\mathbb{E}[\text{cost in phase } \ell | \text{at least } k \text{ boxes are selected in phase } \ell] \\
\leq \frac{\mathbb{E}\left[\text{cost in phase } \ell\right]}{\mathbf{Pr}\left[\text{at least } k \text{ boxes are selected in phase } \ell\right]} \\
\leq \frac{1}{1-\gamma}\mathbb{E}\left[\text{cost in phase } \ell\right] \\
\leq \frac{1}{1-\gamma}\sum_{i\in\mathcal{B}} \alpha \sum_{t\leq 2^\ell} z_{it}^s c_i^s = \frac{1}{1-\gamma}\alpha\overline{f}_c^s < 11.85\overline{f}_c^s.
\end{aligned}
$$

The third line follows by Lemma C.3 and the last line by the definition of $\overline{f}_c^s$. Notice that the upper bound does not depend on the phase ℓ, so the same upper bound holds for f_c^s. Thus the total cost contributed from scenario s in our algorithm is

$$
f^s = f_o^s + f_c^s < 123.25\overline{f}_o^s + 11.85\overline{f}_c^s \leq 123.25\overline{f}^s,
$$

which gives us the lemma.

$\square$

C.3. Rounding against Partially-adaptive for Choosing a Matroid Basis

Similarly to the k boxes case, we relax the domain $\overline{X} = \{x \in [0,1]^{n\times n} : \sum_i x_{it} = 1 \text{ and } \sum_t x_{it} = 1\}$, to be the set of doubly stochastic matrices. Let $r(A)$ for any set $A \subseteq \mathcal{B}$ denote the rank of this set. We define $\overline{g}^s(x)$ as

$$
\min_{y\geq 0, z\geq 0} \sum_{t\in\mathcal{T}} (1-y_t^s) \quad + \sum_{i\in\mathcal{B},t\in\mathcal{T}} c_i^s z_{it}^s \qquad \text{(Relaxation-SPA-matroid)}
$$

$$
\begin{aligned}
\text{subject to} \qquad \sum_{t\in\mathcal{T},i\in A} z_{it}^s &\leq r(A), & \forall A \subseteq \mathcal{B} && (7) \\
z_{it}^s &\leq x_{it}, & \forall i \in \mathcal{B}, t \in \mathcal{T} && \\
\sum_{i\notin A}\sum_{t'\leq t} z_{it'}^s &\geq (r([n]) - r(A))y_t^s, & \forall A \subseteq \mathcal{B}, t \in \mathcal{T} && (8)
\end{aligned}
$$

Our main lemma in this case, shows how to obtain a constant $\log k$-competitive partially adaptive strategy, when given a scenario-aware solution, in the case we are required to select a matroid basis.

Lemma C.4. *Given a scenario-aware fractional solution $\overline{z}^s, \overline{x}$ of cost $\overline{g}^s(\overline{x})$ there exists an efficient partially-adaptive strategy z_s with cost at most $O(\log k)\overline{g}^s(\overline{x})$.*

Proof of Lemma C.4. We follow exactly the steps of the proof of Lemma 6.4 from (Chawla et al., 2020), but similarly to the k items case we highlight the same important properties; (1) the rounding that decides the permutation **does not depend on the scenario** (2) the proof does not use the fact that the initial solution given is optimal in any way. The rounding process is shown in Algorithm 6.

Algorithm 6: Scenario aware, $O(\log n)$-approximate rounding, matroids from (Chawla et al., 2020)

Data: Fractional solution $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ for scenario s, $\alpha = 64$.

```
/* Part 1:  Scenario-independent rounding                              */
```
1   $\sigma :=$ for every $t = 1, \ldots, n$, open each box i independently with probability $q_{it} = \min\left\{\alpha \ln k \frac{\sum_{t' \leq t} x_{it'}}{t}, 1\right\}$.

2

```
/* Part 2:  Scenario-dependent stopping time.                          */
```
3   $\tau_s :=$
4   Given scenario s, calculate $\boldsymbol{y}, \boldsymbol{z}$
5   Let $t_s^* = \min\{t : y_t^s \leq 1/2\}$.
6   **if** $t > t_s^*$ **then**
7   For each opened box i, select it with probability $\min\left\{\frac{\alpha \ln k \sum_{t' \leq t} z_{it'}^s}{t q_{it}}, 1\right\}$.
8   Stop when we find a base of the matroid.
9   **end**

Denote by $(\overline{x}, \overline{y}^s, \overline{z}^s)$ the fractional scenario-aware solution given to us, where $\overline{x}_{it}$ is the fraction that box i is opened at time t, $\overline{z}_{it}^s$ is the fraction box i is chosen for scenario s at time t, and $\overline{y}_t^s$ is the fraction scenario s is "covered" at time t, where covered means that there is a matroid basis selected for this scenario. Denote also by $\overline{f}_o^s(f_o^s), \overline{f}_c^s(f_c^s)$ and the fractional costs for scenario s due to opening and selecting boxes for the fractional (integral) solution respectively.

In scenario s, let phase ℓ be when $t \in (2^{\ell-1}t_s^*, 2^\ell t_s^*]$. We divide the time after t_s^* into exponentially increasing phases, while in each phase we prove that our success probability is a constant. The following lemma gives an upper bound for the opening cost needed in each phase to get a full rank base of the matroid, and still holds in our case, since only uses the constraints of a feasible solution.

Lemma C.5 (Lemma 6.6 from (Chawla et al., 2020)). *In phase ℓ, the expected number of steps needed to select a set of full rank is at most $(4 + 2^{\ell+2}/\alpha)t_s^*$.*

Define $\mathcal{X}$ to be the random variable indicating number of steps needed to build a full rank subset. The probability that we build a full rank basis within some phase $\ell \geq 6$ is

$$\mathbf{Pr}\left[\mathcal{X} \leq 2^{\ell-1}t_s^*\right] \geq 1 - \frac{\mathbb{E}[\mathcal{X}]}{2^{\ell-1}t_s^*} \geq 1 - \frac{1}{2^{\ell-1}t_s^*}(4 + 2^{\ell+2}/\alpha)t_s^* = 1 - 2^{3-\ell} - \frac{8}{\alpha} \geq \frac{3}{4}, \tag{9}$$

where we used Markov's inequality for the first inequality and Lemma C.5 for the second inequality. To calculate the total inspection cost, we sum up the contribution of all phases.

$$\mathbb{E}\left[f_o^s \text{ after phase } 6\right] = \sum_{\ell=6}^{\infty} \mathbb{E}\left[f_o^s \text{ at phase } \ell\right] \cdot \mathbf{Pr}\left[\text{ALG reaches phase } \ell\right]$$

$$\leq \sum_{\ell=6}^{\infty} \sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \sum_{i \in \mathcal{B}} \alpha \ln k \cdot \frac{\sum_{t' \leq t} x_{it'}}{t} \left(\frac{1}{4}\right)^{\ell-6} \qquad \text{Algorithm 6}$$

$$\leq \sum_{\ell=6}^{\infty} 2^{\ell-1} t_s^* \alpha \ln k \cdot \left(\frac{1}{4}\right)^{\ell-6} \qquad\qquad x_{it} \text{ doubly stochastic}$$

$$= \frac{128\alpha \ln k t_s^*}{3} \leq \frac{256 c \ln k \overline{f}_o^s}{3}. \qquad\qquad \text{Since } t_s^* \leq 2\overline{f}_o^s$$

Since the expected opening cost at each step is $\alpha \ln k$ and there are $2^5 t_s^* \leq 64 \overline{f}_o^s$ steps before phase 6, we have

$$f_o^s \leq \alpha \ln k \cdot 64\overline{f}_o^s + \frac{256\alpha \ln k \overline{f}_o^s}{3} = O(\log k)\overline{f}_o^s.$$

Similarly to the k-coverage case, to bound the cost of our algorithm, we find the expected total cost of any phase $\ell \geq 6$, conditioned on boxes forming a full rank base are selected in this phase.

$$\mathbf{E}[f_c^s \text{ in phase } \ell | \text{full rank base selected in phase } \ell]$$

$$\leq \frac{\mathbb{E}\left[f_c^s \text{ in phase } \ell\right]}{\mathbf{Pr}\left[\text{full rank base selected in phase } \ell\right]}$$

$$\leq \frac{1}{3/4}\mathbb{E}\left[f_c^s \text{ in phase } \ell\right]$$

$$\leq \frac{1}{3/4}\sum_{i \in \mathcal{B}}\sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \alpha \ln k \frac{\sum_{t' \leq t} z_{it'}^s c_i^s}{t}$$

$$\leq \frac{1}{3/4}\sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \alpha \ln k \sum_{i \in \mathcal{B}} \frac{\sum_{t' \in \mathcal{T}} z_{it'}^s c_i^s}{2^{\ell-1}t_s^*}$$

$$= \frac{1}{3/4}\alpha \ln k \overline{f}_c^s = O(\log k)\overline{f}_c^s.$$

Such upper bound of conditional expectation does not depend on ℓ, thus also gives the same upper bound for f_c^s. Therefore $f^s = f_o^s + f_c^s \leq O(\log k)(\overline{f}_o^s + \overline{f}_c^s) = O(\log k)\overline{f}^s$.

$\square$

D. Linear Programs & Roundings against NA

D.1. Competing with the non-adaptive for choosing 1 box

The linear program for this case (LP-NA) is already given in the preliminaries section. The result in this case is a $e/(e-1)$-approximate partially adaptive strategy, given in (Chawla et al., 2020) is formally restated below, and the rounding algorithm is presented in Algorithm 7.

Theorem D.1 (Theorem 4.2 from (Chawla et al., 2020))**.** *There exists an efficient partially adaptive algorithm with cost at most $e/(e-1)$ times the total cost of the optimal non-adaptive strategy.*

Algorithm 7: SPA vs NA from(Chawla et al., 2020)

Input: Solution $\boldsymbol{x}, \boldsymbol{z}$ to program (LP-NA); scenario s

1   $\sigma :=$ For $t \geq 1$, select and open box i with probability $\frac{x_i}{\sum_{i \in \mathcal{B}} x_i}$.

2   $\tau_s :=$ If box i is opened at step t, select the box and stop with probability $\frac{z_i^s}{x_i}$.

D.2. Competing with the non-adaptive benchmark for choosing k boxes

We move on to consider the case where we are required to pick k distinct boxes at every round. Similarly to the one box case, we define the optimal non-adaptive strategy that can be expressed by a linear program. We start by showing how to perform the rounding step of line 16 of Algorithm 3 in the case we have to select k boxes. The guarantees are given in

Theorem D.3 and the rounding is presented in Algorithm 8. This extends the results of (Chawla et al., 2020) for the case of selecting k items against the non-adaptive.

Lemma D.2. *There exists a scenario-aware partially adaptive 4-competitive algorithm to the optimal non-adaptive algorithm for picking k boxes.*

Combining this lemma with Theorem 3.4 from (Chawla et al., 2020) we get Theorem D.3.

Theorem D.3. *We can efficiently find a partially-adaptive strategy for optimal search with k options that is $4e/(e-1)$-competitive against the optimal non-adaptive strategy.*

Before presenting the proof for Lemma D.2, we formulate our problem as a linear program as follows. The formulation is the same as LP-NA, we introduce constraints 10, since we need to pick k boxes instead of 1.

$$\text{minimize} \quad \sum_{i \in \mathcal{B}} x_i \quad + \quad \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_i^s z_i^s \qquad\qquad \text{(LP-NA-k)}$$

$$\text{subject to} \quad \sum_{i \in \mathcal{B}} z_i^s \quad = \quad k, \qquad\qquad \forall s \in \mathcal{S} \qquad\qquad (10)$$

$$z_i^s \quad \leq \quad x_i, \qquad\qquad \forall i \in \mathcal{B}, s \in \mathcal{S}$$

$$x_i, z_i^s \quad \in \quad [0,1] \qquad\qquad \forall i \in \mathcal{B}, s \in \mathcal{S}$$

Denote by $\text{OPT}_p = \sum_{i \in \mathcal{B}} x_i$ and $\text{OPT}_c = 1/|\mathcal{S}| \sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_i^s z_i^s$ to be the optimal opening cost and selected boxes' costs, and respectively ALG_p and ALG_c the algorithm's costs.

Algorithm 8: SPA vs NA, k-coverage

Input: Solution x, z to above LP-NA-k, scenario s. We set $\beta = 1/100, \alpha = 1/4950$

1 Denote by $\mathcal{X}_{\text{low}} = \{i : x_i < 1/\beta\}$ and $X = \sum_{i \in \mathcal{X}_{\text{low}}} x_i$

2 $\sigma :=$ open all boxes that $x_i \geq 1/\beta$, from $\mathcal{X}_{\text{low}}$ select each box i w.p. $\frac{x_i}{X}$

3

4 Denote by k' and OPT_c' the values of OPT_c and k restricted in the set $\mathcal{X}_{\text{low}}$

5 $\tau_s :=$ select all boxes that $z_i^s \geq 1/\beta$

6   Discard all boxes i that $c_i > \alpha \text{OPT}_c'/k'$

7   From the rest select box i with probability $\frac{x_i}{X}$

8   Stop when we have selected k boxes in total.

Proof of Lemma D.2. Let (x, z) be the solution to (LP-NA-k), for some scenario $s \in \mathcal{S}$. We round this solution through the following steps, bounding the extra cost occurred at every step. Let $\beta > 1$ be a constant to be set later.

- **Step 1**: open all boxes i with $x_i \geq 1/\beta$, select all that $z_i^s \geq 1/\beta$. This step only incurs at most $\beta(\text{OPT}_p + \text{OPT}_c)$ cost. The algorithm's value cost is $\text{ALG}_c = \sum_{i:z_i^s \geq 1/\beta} c_i$ while $\text{OPT}_c = \sum_i z_i^s c_i \geq \sum_{i:z_i^s \geq 1/\beta} c_i z_i^s \geq 1/\beta \sum_{i:z_i^s \geq 1/\beta} c_i = 1/\beta \text{ALG}_c$. A similar argument holds for the opening cost.

- **Step 2**: let $\mathcal{X}_{\text{low}} = \{i : x_i < 1/\beta\}$, and denote by OPT_c' and k' the new values for OPT_c and k restricted on the set $\mathcal{X}_{\text{low}}$ and by $X = \sum_{i \in \mathcal{X}_{\text{low}}} x_i$.

  - **Step 2a**: convert values to either 0 or ∞ by setting $c_i = \infty$ for every box i such that $c_i > \alpha \text{OPT}_c'/k'$ and denote by $\mathcal{L}_s = \{i : c_i \leq \alpha \text{OPT}_c'/k'\}$.
  - **Step 2b**: select every box with probability $\frac{x_i}{X}$, choose a box only if it is in $\mathcal{X}_{\text{low}}$. Observe that the probability of choosing the j'th box from L_s given that we already have chosen $j - 1$ is

$$\mathbf{Pr}\left[\text{choose } j\text{'th}|\text{have chosen } j-1\right] \geq \frac{\sum_{i \in L_s} x_i - j/\beta}{X} \qquad\qquad \text{Since } x_i \leq 1/\beta \text{ for all } x_i \in \mathcal{X}_{\text{low}}$$

$$\geq \frac{\sum_{i \in L_s} z_i^s - j/\beta}{X} \qquad\qquad \text{From LP constraint}$$

$$\geq \frac{(1-1/\alpha)k' - j/\beta}{\text{OPT}'_p} \qquad\qquad \text{From Markov's Inequality}$$

$$\geq \frac{(1-1/\alpha)k' - k'/\beta}{\text{OPT}'_p} \qquad\qquad \text{Since } j \leq k'$$

$$\geq \frac{(\alpha\beta - \beta - \alpha)k'}{\alpha\beta\text{OPT}'_p}$$

Therefore the expected time until we choose k' boxes is

$$\mathbb{E}\left[\text{ALG}_p\right] = \sum_{j=1}^{k'} \frac{1}{\mathbf{Pr}\left[\text{choose } j\text{'th}|\text{have chosen } j-1\right]}$$

$$\leq \sum_{j=1}^{k'} \alpha\beta \frac{\text{OPT}'_p}{(\alpha\beta - \alpha - \beta)k'}$$

$$= \alpha\beta \frac{\text{OPT}'_p}{\alpha\beta - \alpha - \beta}$$

Observe also that since all values selected are are $c_i \leq \alpha\text{OPT}'_c/k'$, we incur value cost $\text{ALG}_c \leq \alpha\text{OPT}'_c$.

Putting all the steps together, we get $\text{ALG} \leq \left(\beta + \frac{\alpha\beta}{\alpha\beta - \alpha - \beta}\right)\text{OPT}_p + (\beta + \alpha)\text{OPT}_c \leq 4\text{OPT}$, when setting $a = 2\beta/(\beta - 1)$ and $\beta = 1/100$ $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad\square$

D.3. Competing with the non-adaptive benchmark for choosing a matroid basis

In this section $\mathcal{F}$ requires us to select a basis of a given matroid. More specifically, assuming that boxes have an underlying matroid structure we seek to find a basis of size k with the minimum cost and the minimum query time. Let $r(A)$ denote the rank of the set $A \subseteq \mathcal{B}$. Using the linear program of the k-items case, we replace the constraints to ensure that ensure that we select at most $r(A)$ number of elements for every set and that whatever set A of boxes is already chosen, there still enough elements to cover the rank constraint. The guarantees for this case are given in Theorem D.5 and the rounding presented in Algorithm 9. This case also extends the results of (Chawla et al., 2020).

Lemma D.4. *There exists a scenario-aware partially-adaptive $O(\log k)$-approximate algorithm to the optimal non-adaptive algorithm for picking a matroid basis of rank k.*

Combining this lemma with Theorem 3.4 from (Chawla et al., 2020) we get Theorem D.5.

Theorem D.5. *We can efficiently find a partially-adaptive strategy for optimal search over a matroid of rank k that is $O(logk)$-competitive against the optimal non-adaptive strategy.*

In order to present the proof for Lemma D.4, we are using the LP formulation of the problem with a matroid constraint, as shown below. Let $r(A)$ denote the rank of the set $A \subseteq \mathcal{B}$. The difference with LP-NA-k is that we replace constraint 10 with constraint11 which ensures we select at most $r(A)$ number of elements for every set and constraint (12) ensures that whatever set A of boxes is already chosen, there still enough elements to cover the rank constraint.

$$\text{minimize} \quad \sum_{i \in \mathcal{B}} x_i \quad + \quad \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_i^s z_i^s \qquad\qquad \text{(LP-NA-matroid)}$$

$$\text{subject to} \quad \sum_{i \in \mathcal{B}} z_i^s \quad \leq \quad r(A), \qquad\qquad \forall s \in \mathcal{S}, A \subseteq \mathcal{B} \qquad\qquad (11)$$

$$\sum_{i \in A} z_i^s \quad \geq \quad r([n]) - r(A) \quad \forall A \subseteq \mathcal{B}, \forall s \in \mathcal{S} \qquad\qquad (12)$$

$$z_i^s \quad \leq \quad x_i, \qquad\qquad \forall i \in \mathcal{B}, s \in \mathcal{S} \qquad\qquad (13)$$

$$x_i, z_i^s \quad \in \quad [0,1] \qquad\qquad \forall i \in \mathcal{B}, s \in \mathcal{S}$$

Similarly to the case for k items, denote by $\text{OPT}_p = \sum_{i \in \mathcal{B}} x_i$ and $\text{OPT}_c = 1/|\mathcal{S}| \sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_i^s z_i^s$, and ALG_p, ALG_c the respective algorithm's costs.

Algorithm 9: SPA vs NA, matroid

Input: Solution x, z to above LP-NA-matroid, scenario s. We set $\beta = 1/100, \alpha = 1/4950$

1 Denote by $\mathcal{X}_{\text{low}} = \{i : x_i < 1/\beta\}$ and $X = \sum_{i \in \mathcal{X}_{\text{low}}} x_i$

2 $\sigma :=$ open all boxes that $x_i \geq 1/\beta$, from $\mathcal{X}_{\text{low}}$ select each box i w.p. $\frac{x_i}{X}$

3

4 Denote by k^j and OPT_c^j the values of OPT_c and k restricted in the set $\mathcal{X}_{\text{low}}$ when j boxes are selected.

5 $\tau_s :=$ select all boxes that $z_i^s \geq 1/\beta$

6   Discard all boxes i that $c_i > \alpha \text{OPT}_c^j / k^j$

7   From the rest select box i with probability $\frac{x_i}{X}$

8   Stop when we have selected k boxes in total.

Proof of Lemma D.4. Similarly to Lemma D.2, let (x, z) be the solution to LP-NA-matroid, for some scenario $s \in \mathcal{S}$. We round this solution through the following process. Let $\beta > 1$ be a constant to be set later.

- **Step 1**: open all boxes i with $x_i \geq 1/\beta$, select all that $z_i^s \geq 1/\beta$. This step only incurs at most $\beta(\text{OPT}_p + \text{OPT}_c)$ cost.

- **Step 2**: let $\mathcal{X}_{\text{low}} = \{i : x_i < 1/\beta\}$. Denote by OPT_c' and k' the new values of OPT_c and k restricted on $\mathcal{X}_{\text{low}}$. At every step, after having selected j boxes, we restrict our search to the set of low cost boxes $\mathcal{L}_s^j = \{i : v_i \leq \alpha \text{OPT}_c^j / k^j\}$ where OPT_c^j and k^j are the new values for OPT_c and k after having selected $k^j = j$ boxes.

  - **Step 2a**: Convert values to either 0 or ∞ by setting $v_i = \infty$ for every box i such that $v_i > \alpha \text{OPT}_c^j / k^j$.
  - **Step 2b**: Select every box with probability $\frac{x_i}{X}$, choose a box only if it is in $\mathcal{X}_{\text{low}}$. Observe that the probability of choosing the j'th box from L_s given that we already have chosen $j - 1$ is

$$\textbf{Pr} \left[\text{choose } j\text{'th} | \text{have chosen } j - 1\right] \geq \frac{\sum_{i \in L_s^{j-1}} x_i}{X}$$

$$\geq \frac{\sum_{i \in L_s^{j-1}} z_i^s}{X} \qquad \text{From LP constraint (13)}$$

$$\geq \frac{k - (k - j)}{X} \qquad \text{From LP constraint (12)}$$

$$= \frac{j}{\text{OPT}_p'}$$

Therefore the expected time until we choose k' boxes is

$$\mathbb{E}\left[\text{ALG}_c\right] = \sum_{j=1}^{k'} \frac{1}{\textbf{Pr}\left[\text{choose } j\text{'th} | \text{have chosen } j - 1\right]}$$

$$\leq \text{OPT}_p' \sum_{j=1}^{k'} \frac{1}{j}$$

$$\leq \log k \cdot \text{OPT}_p$$

Observe also that every time we choose a value from the set $\mathcal{L}_s^j$, therefore the total cost incurred by the selected values is

$$\text{ALG}_v \leq \sum_{i=1}^{k'} \alpha \frac{\text{OPT}_c^i}{k_i} \leq \sum_{i=1}^{k'} \frac{\text{OPT}_c}{i} \leq \alpha \log k \cdot \text{OPT}_c$$

Putting all the steps together, we get $\text{ALG} \leq O(\log k)\text{OPT}$ $\square$