DATA DRIVEN SEARCH WITH COSTLY INFORMATION: WHEN TO OPEN PANDORA'S BOX

by

Evangelia Gergatsouli

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2024

Date of final oral examination: 04/19/2024

To my mum, my grandma Evangelia, and all the women trying to overcome sometimes impossible hurdles.

Στη μαμά μου, τη γιαγιά Ευαγγελία, και όλες τις γυναίκες που προσπαθούν να ξεπεράσουν κάποιες φορές αδύνατα εμπόδια.

ACKNOWLEDGMENTS

First of all I would like to thank Christos for being my advisor and collaborator throughout this PhD. His enthusiasm for tackling new problems combined with his ability to come up with new ideas were definitely crucial in shaping my own approach to research as well. I also really thank him for giving me the space to explore on my own and work with other people as well, helping me become a independent researcher. A big thanks also to Shuchi, who became a collaborator from the very beginning; a big chunk of the papers in this thesis also include her. Despite not being officially a coadvisor, she was really close to one and I'm grateful for all the advice and suggestions, academic or not. I would also like to thank Ilias Diakonikolas for being in my Qual, Prelim and defense committees, and Eric Bach for being in my final defense committee. I would also like to thank Paul Dutting for submitting a reference letter for me.

Throughout these years I was very lucky to meet and collaborate with people outside UW, and I wanted to thank them individually. Themis Gouleakis, for an (unfortunately remote) internship at Max-Planck at the beginning of the pandemic. Okke Schrijvers for a really fantastic internship at Meta, for his really useful work advice and suggestions and for being a great mentor, even after the internship ended. Finally, Jose Correa for an amazing visit at Universidad de Chile and for welcoming me in one of the most collaborative labs I've ever been to. Also everyone else at U de Chile that made this visit so fun: Victor, Jose V, Bruno, Dana, Andres, Alexandros, Uli and Jannik. Special thanks to Andres and Alexandros for nudging me to apply for this visit and to Anna Me. for her invaluable advice on learning spanish!

A big thanks also goes to my collaborators for discussing all these problems together, some of which are part of this thesis; Alexia Atsidakou, Constantine Caramanis, Shuchi Chawla, Jose Correa, Paul Dutting, Dimitris Fotakis, Themis Gouleakis, Brendan Lucier, Jeremy Mcmahan, Orestis Papadigenopoulos, Nikolas Patris, Charis Pipis, Rojin Rezvan, Miltos Stouras, Yifeng Teng, Alexandros Tsigonias-Dimitriadis, Christos Tzamos, Bruno Ziliotto, Ruimin Zhang. I want to especially mention again Orestis; apart from being a great collaborator, he has also been a friend, always ready to offer advice, and someone I was always looking forward to grabbing a beer with, either in Austin or in any city around the world that we happened to coincide. He is really missed.

To the people that I bothered with my job-market questions, I truly thank them for their time and advice; Greg, Orestis P, Orestis V and Manolis.

Apart from the academic side, I would like to thank all the amazing people around me, old and new; this would have been an infinitely worse experience without them. First of all an enormoos thanks to two people without whom I probably wouldn't be writing these acknowledgments right now. Giorgos K (who despite being 8 timezones away was always available) and Rojin, who were there (remotely or physically) in a not-so-fun period, I am really grateful for that. To Rojin for being one of my closest friends almost from the beginning, and being there through highs and lows, I'm grateful for that. I would have never imagined, when I talked to her on skype right before she joined UW (showing her the snow, like an excited little kid), that we would still be friends 5 years later. To Angeliki, for becoming a close friend from her initial days in Madison, I'm really grateful for all the advice, our many discussions over tsipouro and our fun trips exploring WI (and the salar!). To Erfan, for also becoming a close friend towards the end, for his support, unwavering optimism and for always helping me see the bright side of things, I really need that sometimes! I really enjoyed everything I did with all you guys, and I really hope we'll be closer together and there will be much more of that fun stuff to come!

To the **polis Council, Elena, Emad, Erfan, Luke, Mehrdad, Ramin, Rojin and Soroosh for making the pandemic such a (surprisingly) fun time, it would definitely not have been the same without you all! Bengisu, Erfan and Alex for an amazingly fun summer exploring CA. Everyone else in Madison ; Gokcan, Bengisu, Zack, Kartik, Luke, Soroosh, Natalia, Jesse, Ahmet and Greg for all the fun stuff we did throughout the years; I loved spending every minute of it with all of you and I hope we'll still be able to hang out in the years to come! The CA guys, Eirini, Yeganeh, Amirata and Negin for their hospitality and all the fun stuff we did during my visits there!

To everyone back home or scattered around the world that I unfortunately don't get to see nearly as often as I want: Giorgos K, Alexandros, Maria, Anna B, Dafni, Iliana, Angeliki K, Sevi, Eirini K, Anna O, Eirini T, Manos; thank you for all the (little) time we were able to spend together these 6 years, and our small fun trips around Greece (or CA, Germany and Chile with some of you), during my visits there. Finally my parents, my grandma Evangelia, my sister, my godparents and my two amazing, beautiful cats (probably the cutest ones!). I really really wish we could all be closer!

As a closing (yet important) note, I wanted to acknowledge that University of

Wisconsin-Madison, as many other institutions around the US, is built on land forcefully taken from people living in this area far before Wisconsin became a state. The land that the main UW - Madison campus is built on (called Teejop (day-JOPE)) used to belong to the Ho-Chunk people, who alongside many other Native Nations in WI, were forced to abandon their land.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Data-driven algorithms for combinatorial optimization problem is a brand new area that emerged in the intersection of theoretical computer science and machine learning these past years. It manages to combine modern machine learning techniques and the use of (the abundance of) training data with the formal mathematical guarantees of traditional algorithm design.

In this thesis we study a natural costly search problem; having some preliminary (potentially inaccurate/uncertain) input data for n different options, information about this uncertain input can be obtained at extra monetary or computational overhead. Our goal is to find the best option for us (i.e. the cheapest) without spending too many resources obtaining accurate information. Determining the optimal manner for acquiring information has now become an online decision-making problem: each piece of information obtained by the algorithm can affect whether and which piece to acquire next

Pandora's Box problem (Weitzman, 1979), an old economics paradigm, and its extensions capture optimization problems with stochastic input where the algorithm can obtain instantiations of input random variables at some cost. Learning exploration policies given input distributions has been a main challenge of machine learning frequently associated with Reinforcement Learning, however, most of the applied methods for combinatorial algorithms usually have no theoretical guarantees or they optimize only over a parameterized family of algorithms. But even in non-parametric settings we usually assume full knowledge of the input distribution, which is not the case in practice.

In contrast to the previous work, this thesis focuses on two main themes; (1) learning policies for *non-parametric* benchmarks (2) robust policies for unknown or noisy settings, both through the lens of Pandora's Box.

1 INTRODUCTION

Traditional algorithm design focuses on solving the various combinatorial optimization problems in *worst case instances* and in a *static* manner; the algorithm takes the input at the beginning, decides on a process of solving the problem and executes it. Modern problems however could be evolving rapidly, thus requiring algorithms that re-evaluate their input and potentially change their initial process, while at the same time being able to utilize the new data that came up, instead of pessimistically assume worst case input. This area is known as **data-driven algorithms design** and largely lies in the intersection of traditional theoretical Computer Science areas like Online and Approximation algorithms, Stochastic Optimization as well as Machine Learning and Data Science.

There are many real life scenarios where we are called to make a decision, while only having partial information. We can, however, spend more time or resources learning more about our options in order to make a better decision. Such problems are modeled by a problem from economics called Pandora's Box, first formulated by Weitzman (1979). More concretely, we can think of our different options as n boxes, each of which contains a price which follows some known distribution. In order to see the price drawn we need to pay an *opening* cost for this box. In the our goal is to pick a box such that we minimize our total exploration cost plus the price of the box we chose. This is a very well-studied problem in the economics literature, and to better motivate the problem we include some economics-inspired examples and applications.

- **Housing Market**: Imagine you are looking to buy a house, and have gone through many listing online that mention a price for the house and contain photos and videos of the property. However, it would be prudent to visit the house in person, thus spending time and gas, to verify that the house is as described in the listing. It could be the case, that some damages were not disclosed to you, therefore the actual price is higher than the one listed. It would be unreasonable to visit every house that is listed for sale online, but *how do we choose which ones to visit and which one to buy in the end*?

- **Candidate Search**: In another common scenario, a company is looking to hire an employee for an open position. They receive a multitude of CVs, and are planning on interviewing the candidates that look like the best fit for the open position. However, interviews are costly; some company employee has to spend time with the interviewees and flying the candidate out to the company location costs actual money. *How should the company choose the candidates to interview, and the one to hire in the end*?

- **Consumer Search**: In the simplest scenario that we all face daily, imagine we are looking to buy a gift for our friend, and we know this item is sold in a bunch of shops. Despite having some prior information on the shops (e.g. some of them are notoriously expensive, but some might usually sell cheaper products), we need to spend time to visit the shop and inquire about our gift. Since we do not have the time to visit every shop to find the cheapest option, *how do we decide which shops to visit and when to stop searching*?

The optimal policy for this problem is given by a simple and elegant greedy algorithm, described in Weitzman (1979). For this policy to work Weitzman made the assumption that the different options are **independent of each other** ; an assumption that largely does not hold in practice, as we highlight in the examples mentioned above.

- **Housing Market**: there could be local geographical phenomena that affect close by houses the same way. For example a neighborhood's proximity to water (e.g. lake, sea or river) might imply that all nearby houses have a higher chance of water or mold damage being present.
- **Candidate Search**: similarly, candidates from the same school are affected the same way. Having taken the same classes with the same professors, means that their background has a high chance of being good for both or bad for both.
- **Consumer Search**: different shops use different suppliers which could affect their prices. For example if a supplier is running low on a part, all the shops using this supplier will probably have a shortage and sell the items that use this part at a higher price.

Wanting to solve a problem as close to real life situations as possible, **in this thesis we tackle PANDORA's BOX with correlations from a data-driven perspective**. We address two main themes on data driven algorithm design. First we design algorithms for non-parametric benchmarks; most of the literature in data-driven algorithms focuses on fixing algorithms with specific parameters that are then learned. Our work, on the other hand, does not restrict the algorithm to use specific parameters. The second theme explored in this thesis is that of robustness to the prior information given. It is not always possible to obtain accurate and complete information on the problem, therefore we studied what happens when prior information is fully unknown, and how we can incorporate the information obtained during the game in our future decisions.

To summarize, in this thesis we address two main problems through the lens of the Correlated Pandora's Box problem;

- Part I: Designing policies for *non-parametric* benchmarks.
- Part II: Learning policies with partial or imperfect access to the prior distribution.

1.0.1 Related Work

The economic theory of search, a subfield of microeconomics, is a long , initiated by Stigler (1961). Since then there have has been a vast amount of work on subjects related to the economic theory of search (Stigler, 1961; Rothschild, 1978; Wilde, 1980; Weitzman, 1979), we mention some examples of subtopics here. Job/labor markets (McCall, 1970; Mortensen, 1986; Miller, 1984) study the economics of job searching (from the perspective of the company or the applicants), often trying to model how people of different backgrounds/ages behave while searching for a job (Holzer, 1988). The housing market (Quan and Quigley, 1991) that study how the buyer and seller decide on their prices (to offer their property at and to buy a potential property respectively). Consumer information search (Ratchford, 1982; Simonson et al., 1988; Moorthy et al., 1997) that study how consumers behave when on the market for a product; how priors affect their behavior, what happens with uncertain priors and when information acquisition is costly for them. In general the works coming from economics focus more on the modeling aspect of the problem at hand, compared to the computer science work that focuses on the algorithmic aspect. For a more complete survey on the economic theory of search and its applications we refer the reader to the book by McCall and McCall (2007).

The problem that models all these scenarios is Pandora's Box, first formulated in (Weitzman, 1979). The reservation values proposed by Weitzman are a special case of Gittins index (Gittins and Jones, 1974) who came up with them independently. There are various proofs of this (Whittle, 1980; Weber, 1992), with a simple alternative proof proposed by (Tsitsiklis, 1994). We refer the reader to the book (Gittins et al., 2011) for more information on Gittins indices. This line of work is also closely connected to Markov chains literature, as observed in Dumitriu et al. (2003).

Inspired by Weitzman's seminal work on Pandora's Box (Weitzman, 1979), there has been a renewed interest in the computer science community to study variants of the original problem. Initially, alternative models of inspection were first considered, with the main work done on non-obligatory inspection (Guha et al., 2008; Beyhaghi and Kleinberg, 2019; Doval, 2018; Fu et al., 2023; Beyhaghi and Cai, 2023a) where a box can be selected without being opened. Similar to this, there have been variants where boxes can be partially open (Aouad et al., 2020) or being gradually and continuously opened while updating the prior (Ke and Villas-Boas, 2019). Kleinberg et al. (2016) presents an alternative simpler proof of Weitzman's reservation values, with an interesting intuitive interpretation. Our work, presented also as part of this thesis, tackled the version where the prior distributions are correlated (Chawla et al., 2020, 2023; Gergatsouli and Tzamos, 2024). In the online setting (Gergatsouli and Tzamos, 2022; Guo et al., 2021; Gatmiry et al., 2024; Atsidakou et al., 2024) the problem is studied from a regret minimization perspective. Close to the online setting Esfandiari et al. (2019) study a variant where there is a distribution over the costs, and draws connections to the bandit literature. A setting where the boxes have combinatorial cost is studied in Berger et al. (2023), while Bechtel et al. (2022) studies the variant where the search is delegated. In (Boodaghians et al., 2020) we no longer have control over the order, which is fixed and given to us at the beginning. Finally models where the ability to go back and pick the best option seen is challenged are also considered in Karni and Schwartz (1977) where recall happens with some probability, while in later works the problem were the algorithm has to commit and cannot turn back to pick the best option seen Fu et al. (2018); Segev and Singla (2021) are studied, giving a PTAS and an EPTAS respectively. For a more detailed overview of the variants of Pandora's Box in the computer science literature we refer the reader to the recent survey by Beyhaghi and Cai (2023b).

Finally, all the Pandora's Box-related problems fall under the wider *price of information* literature where the goal is to solve a task that requires obtaining some costly information (Gupta and Kumar, 2001; Charikar et al., 2000; Chen et al., 2015b,a) or studies the structure of approximately optimal rules for several combinatorial problems in this costly information setting (Singla, 2018; Gupta et al., 2019; Ma and Tzamos, 2023; Drygala et al., 2023).

1.1 Our Contributions

In Part I we explored various non-parametric benchmarks for PANDORA'S BOX. Our aim is to learn search policies that do not depend on any algorithm parameters whatsoever, and give provable theoretical performance guarantees. A searching policy consists of two main components;

1. which box to open/explore next (also called *exploration or opening order*),
2. when to stop (also called *stopping rule*).

We study three different family of strategies, that differ in the amount of adaptivity on the components (1) and (2) of their policy they are allowed. We started from the most restrictive *Non-Adaptive* benchmark, moving on to the *Partially-Adaptive* and finally tackling the most general *Fully-Adaptive* benchmark. We summarized their main differences in Table 1.1 and explained in more detail in the paragraphs that follow. For a pictorial depiction of the relation between the different benchmarks we included Figure 1.1.

	Adapt Stopping time	**Adapt Exploration Order**
Non-Adaptive	No	No
Partially-Adaptive	Yes	No
Fully-Adaptive	Yes	Yes

Table 1.1: Non-parametric benchmarks comparison.

The Fully-Adaptive benchmark This is the most general benchmark; in a *Fully-Adaptive* policy the order of exploration and the stopping rule can arbitrarily depend on the values realized so far.

The Partially-Adaptive benchmark Motivated by the structure of the optimal policy in the original solution (Weitzman, 1979) we also considered the benchmark of *Partially-Adaptive* policies. In such policies, the exploration order has to be fixed beforehand, but the stopping rule can arbitrarily depend on the values realized.

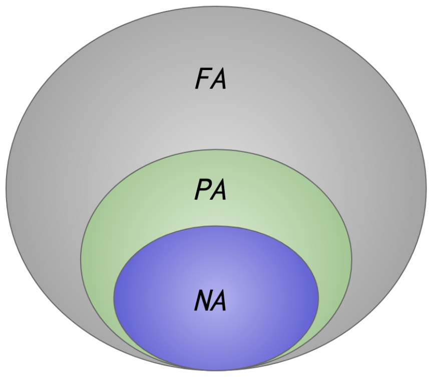

Figure 1.1: The picture shows the relation between the benchmarks. The non-adaptive family is a subset of the Partially-Adaptive which is a subset of the Fully-Adaptive.

The Non-Adaptive benchmark This is the most restricted benchmark that allows for no adaptivity at all. Both the order of exploration and the stopping time have to be fixed a priori (before any boxes are opened) and cannot depend on the prices seen inside the boxes explored during the execution of the algorithm. Since this is a relatively weak benchmark, we will only present some results as a warmup to our main results for the Partially-Adaptive benchmark, in Section 3.A.1.

1.1.1 Competing with the Partially-Adaptive

We begin by describing our initial approach for competing with the Partially-Adaptive benchmark in Chapter 3, published in Chawla et al. (2020). This approach works by separating the two decisions of the policy: (1) deciding the exploration order and (2) deciding the stopping time, where each step incurs a constant factor loss in approximation (Figure 1.2). This LP-based algorithm will result in a 9.22-approximation to the optimal Partially-Adaptive.

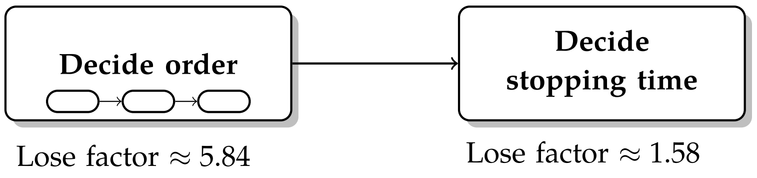

Figure 1.2: Initial approach vs Partially-Adaptive.

In Chapter 4 we simplify our initial approach by directly generalizing the reservation values policy described by Weitzman (1979). We describe two versions of this approach that differ in the update of our prior distribution beliefs and give an almost tight (4.42) approximation. We summarized the results against the Partially-Adaptive benchmark in Figure 1.3.

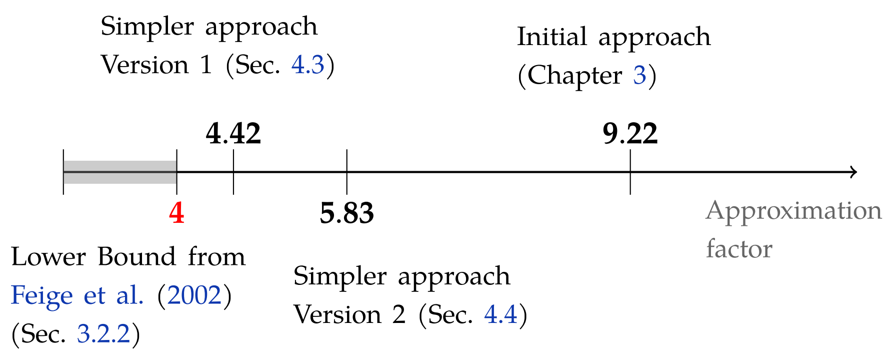

Figure 1.3: Landscape of results for PANDORA's BOX vs Partially-Adaptive

1.1.1.1 More complex constraints

Following our study on the traditional PANDORA's BOX objective of selecting only one option, we also extend our results to more complex constraints, summarized in table 1.2. We study the following constraints;

1. k items: for this constraint we are required to select k boxes instead of one at the end of the algorithm. The results are presented in Section 3.3.1,
2. matroid of rank k: in this case the boxes form a matroid and we are required to pick a matroid basis of size k. The results are presented in Section 3.3.2.

	Result
k items	$O(1)$ (Theorem 3.12)
matroid basis	$\Theta(\log k)$ (Theorems 3.17 and 3.15)

Table 1.2: Results for complex constraints

1.1.2 Competing with the Fully-Adaptive

We continue by studying the more general Fully-Adaptive benchmark. We show via a series of reductions how our problem is connected to two other well studied problems; Min Sum Set Cover and Optimal Decision Tree. The reductions given in Sections 5.3

and 5.4 imply a series of approximation ratios and lower bounds summarized in Figure 1.4 and explained in detail in Section 5.2.

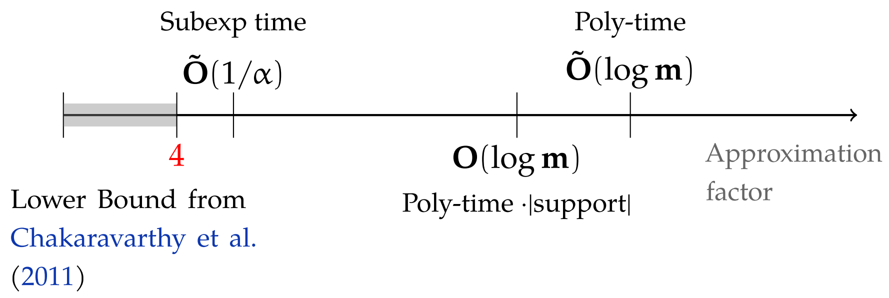
Subexp time
$\tilde{\mathbf{O}}(1/\alpha)$
Poly-time
$\tilde{\mathbf{O}}(\log \mathbf{m})$
4
$\mathbf{O}(\log \mathbf{m})$
Poly-time ·|support|
Approximation factor
Lower Bound from Chakaravarthy et al. (2011)

Figure 1.4: Landscape of results for PANDORA's Box vs Fully-Adaptive

In Part II we tackle three different regimes of partial knowledge on the prior distribution. From the most pessimistic one where we have no information on the prior distributions, to the intermediate one where we are provided with noisy information to the one where we directly get samples from the prior distributions.

1.1.3 Sample Access

We begin by studying the case where we do not have full access on the prior distribution, but we can only get samples from it (Chapter 7). We first show that against the Fully-Adaptive benchmark we have no hope of obtaining any meaningful approximation. Then in Section 7.1 we revisit our previous results against the Partially-Adaptive benchmark when only given sample access to the joint prior distribution. We show how our results extend to this setting, and maintain the approximation guarantees within $(1+\varepsilon)$ when getting $\text{poly}(n, 1/\varepsilon, \log 1/\delta)$ number of samples. The results are summarized in Table 1.3.

	Approximation Factor
Choose 1 box	$(1+\varepsilon)4.42$ (Sec. 7.1.2), $(1+\varepsilon)9.22$ (Sec. 7.1.1)
Choose k boxes	$(1+\varepsilon)O(1)$ (Sec. 7.1.1)
Choose matroid basis	$(1+\varepsilon)O(\log k)$ (Sec. 7.1.1)

Table 1.3: Approximation factors when selecting $\text{poly}(n, 1/\varepsilon, \log 1/\delta)$ samples. Results hold w.p. $1-\delta$.

1.1.4 Unknown Distributions

In Chapter 8 we switch to a worst case model for our knowledge of the joint prior distribution; we are given no information at all for the prior distribution. We tackle the online version of Pandora's Box where we have to find an algorithm for each of T days with the goal of minimizing regret (i.e. our average distance from the optimal solution of each round). There are two settings arising in this problem, depending on the amount of information we see after the game of each day ends.

- *Full information setting*: at the end of each day we see all the prices inside the boxes.
- *Bandit setting*: at the end of each day we only see the prices in the boxes we chose to open.

We begin in Section 8.1 with the case where the prices inside the boxes can be adversarially selected. In this setting we give approximate no regret algorithms for both the bandit and the full information setting. Our algorithm first relaxes the initial integer problem a continuous convex one, then uses any online convex optimization algorithm to solve it, and finally rounds the solution given any rounding algorithm with good guarantees. The process is summarized in Figure 1.5. The final rounding part, is possible through our previous results of Chapter 3, where we presented LP-based rounding schemes for Pandora's Box even with complex constraints.

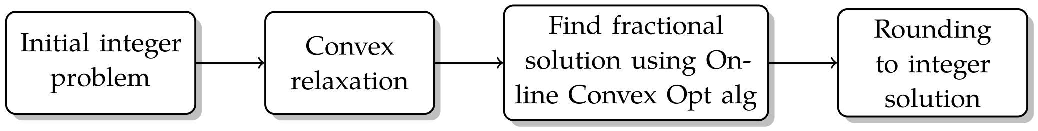
Initial integer problem
Convex relaxation
Find fractional solution using Online Convex Opt alg
Rounding to integer solution

Figure 1.5: High level overview of our algorithm for online Pandora's Box with adversarial prices.

Using the framework above, we are able to obtain the following results, presented formally in Corollary 8.7 for the full information setting & Corollary 8.11 for the bandit setting.

- 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 note that for the full information setting the regret factor is $O(T^{-1/2})$ (Theorem 8.6) while for the Bandit setting the regret factor is $O(T^{-1/3})$ (Theorem 8.9).

1.1.4.1 Adding Context

Following this, in Section 8.2 we consider adding the extra information of *context* at the beginning of every round. The adversary in this case can select the distributions at each round, and we still have no knowledge of them. We require however that a *realizability assumption* holds; There exists an optimal vector $\boldsymbol{w}^*$ and a function h that is used by the optimal to calculate the reservation values of the boxes for each round. As we explain in Section 8.2 this is a minimal assumption for the problem to be meaningful. Our framework essentially reduces our initial PANDORA's Box problem with context to a simpler online regression one, and then used Weitzman's strategy to solve it (see Figure 1.6).

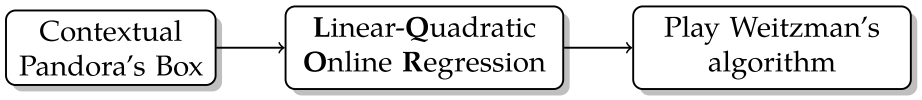

Figure 1.6: High level overview of our algorithm for Contextual PANDORA's BOX.

Our final result is informally presented below, while the formal definition is given in Theorem 8.17.

Main Theorem (Informal Theorem 8.17)**.** *Given an oracle that achieves expected regret* $r(T)$ *for Linear-Quadratic Online Regression, our Algorithm achieves* $O(\sqrt{Tr(T)})$ *regret for the* CONTEXTUAL PANDORA'S BOX *problem. In particular, if the regret* $r(T)$ *is sublinear in* T*, our algorithm achieves sublinear regret.*

1.1.5 Noisy Data

Finally, in Chapter 9, we switch to an intermediate model, where we have access to noisy data on our distribution. In Section 9.1 we tackle PANDORA's Box against the Fully Adaptive optimal, when we have access to a mixture of m product distributions and connect it to the noisy version of Optimal Decision Tree. This will give us a $O(1)$-approximation Dynamic Programming algorithm.

1.2 Bibliographic Notes

The results presented in Part I are based on joint work with[1] Shuchi Chawla, Jeremy McMahan, Yifeng Teng, Christos Tzamos, and Ruimin Zhang. Specifically, Chapter 3 is based on Chawla et al. (2020), Chapter 4 is based on Gergatsouli and Tzamos (2024), Chapter 5 is based on Chawla et al. (2023).

The results presented in Part II are based on joint work with Alexia Atsidakou, Constantine Caramanis, Shuchi Chawla, Jeremy McMahan, Orestis Papadigenopoulos, Yifeng Teng, Christos Tzamos and Ruimin Zhang. Specifically, Chapter 7 is based on Chawla et al. (2020), Chapter 8 is based on Gergatsouli and Tzamos (2022) and Atsidakou et al. (2024).

[1]All co-authors listed in alphabetical order.

Part I

Non-parametric Benchmarks

2 INTRODUCTION & OVERVIEW OF RESULTS

The goal of this part of the thesis is to study the different non-parametric benchmarks that we can use for PANDORA'S BOX. We are studying three different benchmarks, also briefly mentioned in Section 1.1, which we define here more formally. Our aim is to learn search policies that do not depend on any algorithm parameters whatsoever, and give provable theoretical performance guarantees. Recall that an algorithm for PANDORA'S BOX consists of two parts

1. Which box to open/explore next (also called *exploration order*)
2. When to stop (also called *stopping rule*)

The instance of the problem is a set of n boxes $\mathcal{B}$, a cost $c_i \in \mathbb{R}^+$ for each box $i \in \mathcal{B}$, and a joint distribution $\mathcal{D}$. This distribution can be thought of as a collection of vectors of size n, called *scenarios*. Each of these scenarios happens with some known probability. We can think of the distribution as the example in Table 2.1 below, where p_i is the probability of scenario i happening.

	Box 1	**Box 2**	**Box 3**	...	**Box n**
$p_1 \to$ **Scenario** 1:	42	13	15		24
$p_2 \to$ **Scenario** 2:	0	24	94	$\dots$	2
$\vdots$			$\vdots$		
$p_m \to$ **Scenario** m:	31	15	9	$\dots$	2

Table 2.1: Example of the joint distribution $\mathcal{D}$, with n boxes, and m scenarios.

As already mentioned in Section 1.1 in our work, we study three different family of strategies, that differ in the amount of adaptivity on the components (1) and (2) of their policy they are allowed. We summarized their main differences in Table 1.1 of Section 1.1 and in Figure 1.1 for a pictorial representation of their power. We explain the benchmarks here in more technical details.

The Fully-Adaptive benchmark The most general benchmark allows the optimal policy to arbitrarily adapt both the exploration order and the stopping time, based on what the prices revealed inside the boxes are. In more concrete terms, this is an optimal that sees the same information as we do, but has infinite computation power to compute the following exponential dynamic program. We denote by $\mathcal{O}$ the set of opened boxes. The dynamic program at each step has to decide whether to stop and pick the best price found (i.e. $\min_{i \in \mathcal{O}} v_i$) or continue and pay the next box c_j and

$$\mathrm{OPT}(\mathcal{O}) = \begin{cases} \min_{i \in \mathcal{O}} v_i & \text{if } \overline{\mathcal{O}} == \emptyset \\ \min_{j \notin \mathcal{O}} \left(\min_{i \in \mathcal{O}} v_i, c_j + \mathrm{OPT}(\mathcal{O} \cup \{j\}) \right) & \text{else} \end{cases} \tag{2.1}$$

Observe that $\mathcal{O}$ can take 2^n different values, and at every step we have a computation of n, therefore our DP has size $O(n2^n)$.

The Partially-Adaptive benchmark This benchmark is motivated by the structure of the optimal policy in the original solution (Weitzman, 1979). In a Partially-Adaptive policy, the exploration order has to be fixed beforehand, but the stopping rule can arbitrarily depend on the values realized. The optimal therefore is minimizing

$$\mathbf{E}_{\mathcal{D}} \left[\min_{i \in \mathcal{O}} v_i + \sum_{i \in \mathcal{O}} c_i \right],$$

where $\mathcal{D}$ is the joint distribution over all boxes, and c_i is the cost of box i. Another way to think about this benchmark is that it consists of a permutation π (i.e. a fixed ordering of boxes) and a stopping rule τ.

TheNon-Adaptive benchmark This is the most restricted benchmark that allows for no adaptations at all. Both the order of exploration and the stopping time have to be fixed a priori (before any boxes are opened) and cannot depend on the prices seen inside the boxes explored during the execution of the algorithm.

2.1 Related work

The seminal work of Gupta and Roughgarden Gupta and Roughgarden (2017) introduced the problem of algorithm selection in a distributional learning setting focusing on the number of samples required to learn an approximately optimal algorithm. A long line of recent research extends this framework to efficient sample-based optimization over parameterized classes of algorithms Ailon et al. (2006); Clarkson et al. (2010); Gupta and Roughgarden (2017); Balcan et al. (2017, 2018a,b); Kleinberg et al. (2017); Weisz et al. (2018); Alabi et al. (2019). In contrast to these results our work studies optimization over larger, non-parametric classes of algorithms, indeed any polynomial time (partially-adaptive) algorithm. a more flexible design space aims to speed-up computations for polynomially solvable problems. A related work of Alabi et al. Alabi et al. (2019), also considers speeding up computations for polynomially solvable problems. In their model, they design competitive algorithms with the non-adaptive optimum in finding the best solution but they crucially require that the answer is always unique. In comparison, our model allows trading off computation time with the cost of the solution and is able to compete against more adaptive solutions without making any assumptions on the structure of solutions.

For more details on this line of research, we refer the reader to Chapter 29 of Roughgarden (2021), on data-driven algorithm design.

2.2 Overview of Results

2.2.1 Competing with the Partially-Adaptive

Our initial approach for competing with the Partially-Adaptive benchmark (Chapter 3) is essentially reducing Pandora's Box to a special case called Min-Sum Set Cover. In this problem, all prices inside the boxes are either 0 or ∞, which effectively forces the algorithm to always find a 0.

This approach works by separating the two decisions of the policy: (1) deciding the exploration order and (2) deciding the stopping time, where each step incurs a constant factor loss in approximation (Figure 1.2). The final result is stated in the following informal version of Theorem 3.5.

Main Theorem (Informal Theorem 3.5)**.** *We can efficiently find a Partially-Adaptive strategy that is* 9.22*-competitive against the optimal Partially-Adaptive strategy.*

Our algorithm works in the following two steps.

Deciding an exploration order This part works by reducing Pandora's Box to Min-Sum Set Cover. We focus on a family of strategies called scenario-aware Partially-Adaptive, where we are still required to fix the order of exploration but then we get to see all the prices realized inside the boxes[1]. After formulating a linear program for scenario-aware partially adaptive strategies, we carefully choose a threshold T, and

[1]This is just for the analysis, since we cannot actually know the realized prices

change our prior such that for any box b and scenario s if $v_{bs} \leqslant T$, we treat this as 0, otherwise $v_{bs} = \infty$. This transforms our problem to a Min-Sum Set Cover instance, and we can use the algorithm of Feige et al. (2004) to obtain a 4-approximation. The key observation here, which will become apparent in the technical sections where the LP is described, is that the opening order **will not depend** on the scenario realized.

Deciding a stopping time After having decided an order, we are now required to pick a stopping time. We do this using a technique from an older online algorithms problem called Ski Rental. The intuition for this part is that we have two costs in the algorithm; exploration cost (always increasing) and best price seen so far (weakly decreasing). The idea is to stop whenever the two costs become equal, and this gives a 2-approximation (see also Figure 2.1 below). Using the randomized algorithm for ski rental from Karlin et al. (1990) we can improve this to 1.58.

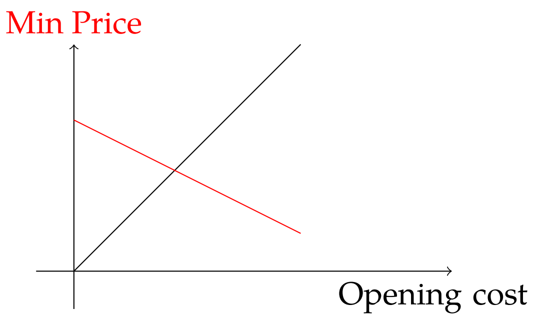

Figure 2.1: Ski rental argument for deciding the stopping time. Stopping where the lines meet gives a 2 approximation.

This approach despite having the nice property of separating the two decisions of the algorithm, suffers from a huge computational overhead, since it requires the solution of a linear program. In Chapter 4 we simplify this approach by directly generalizing the reservation values policy described by Weitzman (1979).

We first rewrite the reservation values described by Weitzman in a way that is easier to use for our algorithm, which essentially remains the same as Weitzman's with the only differnce that we have to update our prior after every step. There is two different ways to do that; (1) Updating on the fact that the best value seen V_b is more than σ_b (2) Full updates, given on the exact value we saw ($V_b = v$). Both variants give constant approximation, however Variant 1 also has the property that can be learned from samples, which we further discuss in Chapter 7.

- Updating with $V_b > \sigma_b$: For this variant we obtain the following theorem

  Main Theorem (Informal Theorem 4.2)**.** *Our Algorithm is a* 4.428*-approximation for* Pandora's Box *against the Partially-Adaptive optimal, when conditioning on* $V_b > \sigma_b$*.*

  It is worth pointing out that this algorithm is almost tight; there is a lower bound of 4 implied by Feige et al. (2004), which we discuss in Section 3.2.2.

- Updating with $V_b = v$: For this variant we obtain the following theorem.

  Main Theorem (Informal Theorem 4.3)**.** *Our Algorithm is a* 5.828*-approximation for* Pandora's Box *against the Partially-Adaptive optimal, when conditioning on* $V_b = v$*.*

  Someone might wonder why the approximation factor is worse for this case, despite using more information. We would like to point out that more information does not mean better approximation ratio, since our algorithm is greedy and not the optimal.

We summarized the results against the Partially-Adaptive benchmark in Figure 2.2 (same as Figure 1.3 from Section 1.1, repeated here for convenience).

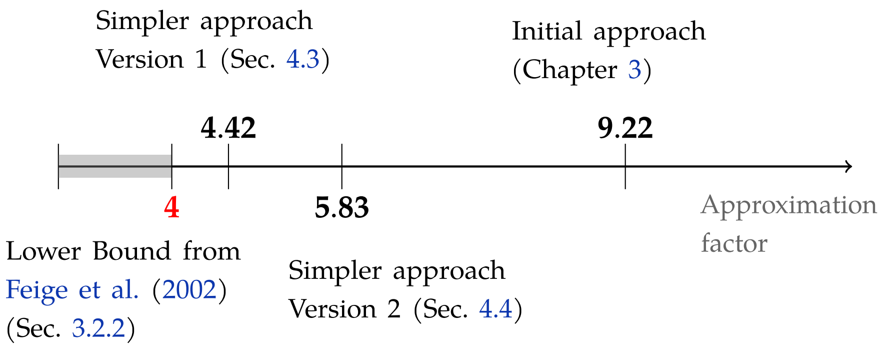

Figure 2.2: Landscape of results for Pandora's Box vs Partially-Adaptive(same as Figure 1.3)

2.2.1.1 More complex constraints

Following our study on the traditional Pandora's box objective of selecting only one option, we also extend our results to more complex constraints, summarized in table 1.2. Both the algorithms for these results are based on rounding of a generalized version of the linear program used in the single box case. We study the following constraints;

1. k items: for this constraint we are required to select k boxes instead of one at the end of the algorithm. The main result is as follows

   Main Theorem (Informal Theorem 3.12)**.** *We can efficiently find a Partially-Adaptive strategy for optimal search with* k *options that is* $O(1)$*-competitive against the optimal Partially-Adaptive strategy.*

2. matroid of rank k: in this case the boxes form a matroid and we are required to pick a matroid basis of size k. In this case we have both a lower and a matching (up to constants) upper bound. The upper bound is described in the following theorem

   Main Theorem (Informal Theorem 3.15)**.** *We can efficiently find a Partially-Adaptive strategy for optimal search over a matroid of rank* k *that is* $O(\log k)$*-competitive against the optimal partially-adaptive strategy.*

   The lower bound, is described below. For this result we give an approximation preserving reduction from Set Cover, which we know is hard to approximate within a $\log k$ factor (where k is the number of sets). This theorem is even stronger in the sense that it holds for any Fully-Adaptive algorithm (the strongest one) against any Non-Adaptive optimal (the weakest benchmark).

   Main Theorem (Informal Theorem 3.17)**.** *Assuming NP$\not\subseteq$RP, no computationally efficient fully-adaptive algorithm can approximate the optimal non-adaptive cost within a factor of* $o(\log k)$.

2.2.2 Competing with the Fully-Adaptive

We continue by studying the more general Fully-Adaptive benchmark in Chapter 5. The problem in this case becomes harder to deal with, due to the adaptive nature of the optimal. A first idea someone might come up with, is trying to fully identify which is the scenario that happened, and then choose the best box for this scenario. It turns out that this is exactly the Optimal Decision Tree problem. However, it turned out that our problem is equivalent to a uniform version of this problem called Uniform Decision Tree where the only difference is that that potential scenarios have uniform probability.

To show this equivalence, we go through a uniform version of Min-Sum Set Cover with extra feedback at every step. Specifically, we show the series of reductions shown in Figure 2.3 (same as Figure 5.1 of Chapter 5, and repeated here for convenience).

$$\underbrace{\mathrm{PB} \xleftrightarrow{\text{Section 5.3}} \mathrm{UMS}}_{\text{Log-log factors}}\underbrace{\mathrm{SC}_f \xleftrightarrow{\text{Section 5.4}} \mathrm{UDT}}_{\text{Constant factors}}$$

Figure 2.3: A summary of our approximation preserving reductions (same as Figure 5.1 from Chapter 5)

The reductions given in Sections 5.3 and 5.4 imply a series of approximation ratios and lower bounds summarized in Figure 2.4 (same as Figure 1.4 from section 1.1 and repeated here for convenience).

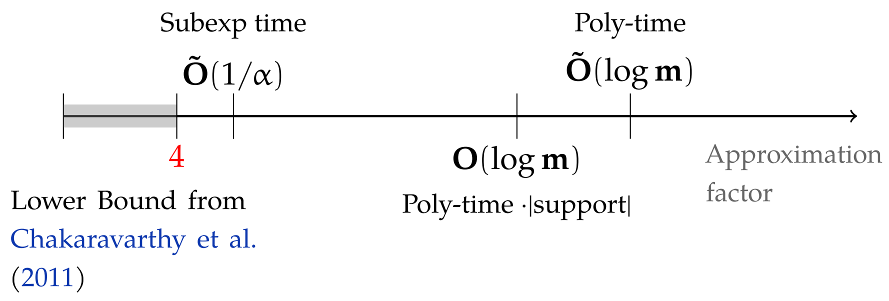
Subexp time
$\tilde{\mathbf{O}}(1/\alpha)$
Poly-time
$\tilde{\mathbf{O}}(\log \mathbf{m})$
4
$\mathbf{O}(\log \mathbf{m})$
Approximation factor
Lower Bound from Chakaravarthy et al. (2011)
Poly-time ·|support|

Figure 2.4: Landscape of results for Pandora's Box vs Fully-Adaptive.

3 PARTIALLY-ADAPTIVE: INITIAL APPROACH

Recall from our initial discussion on PANDORA'S BOX (Weitzman, 1979) in Section 1 the formulation of the problem; the online algorithm is presented with n boxes, each containing an unknown stochastic reward. The algorithm can open boxes in any order at a fixed overhead each; observes the rewards contained in the open boxes; and terminates upon selecting any one of the rewards observed. The goal is to maximize the reward selected minus the total overhead of opening boxes. Weitzman showed that a particularly simple policy is optimal for the PANDORA'S BOX problem: the algorithm computes an index for each box based on its reward distribution and opens boxes in decreasing order of these indices until it finds a reward that exceeds all of the remaining indices.

A crucial assumption underlying Weitzman's optimality result is that the rewards in different boxes are independent. This does not always bear out in practice. Suppose, for example, that you want to buy an item online and look for a website that offers a cheap price. Your goal is to minimize the price you pay for the item plus the time it takes to search for a good deal. Since the websites are competing sellers, it is likely that prices on different sites are correlated. For another example, consider a route planning service that wants to determine the fastest route between two destinations from among a set of potential routes. The driving time for each route is stochastic and depends on traffic, but the route planning service can obtain its exact value at some cost. The service wants to minimize the driving time of the route selected plus the cost spent on querying routes. Once again, because of network effects, driving times along different routes may be correlated. How do we design an online search algorithm for these settings?

In this chapter, we provide the first competitive algorithms for Pandora's Box-type problems with correlations. Formalizing the examples described above: there are n alternatives with unknown costs that are drawn from some joint distribution. A search algorithm examines these alternatives one at a time, learning their costs, and after a few steps stops and selects one of the alternatives. Given access to the full prior distribution of costs, our goal is to develop a search algorithm that minimizes the sum of the expected cost of the chosen alternative and the number of steps to find it.

Related literature The Min-Sum Set Cover problem was initially introduced in Feige et al. (2002). After its initial appearance, the *multiple intents re-ranking* problem was introduced in (Azar et al., 2009) and later named generalized Min Sum Set Cover in Bansal et al. (2010) where the objective is to select $k(S)$ items for any set S. The approximation ratio of 485 given by Bansal et al. (2010) was then improved to 28 in Skutella and Williamson (2011) and recently improved to 4.642 in Bansal et al. (2023). Im et al. (2014) study the preemptive generalized Min Sum Set Cover, where the elements ca be fractionally chosen, and give a 2-approximation. A further generalization, *Submodular Ranking* is studied in (Azar and Gamzu, 2011) where the coverage property is replaced by satisfying a submodular function. Another generalization is presented in Im et al. (2016) for the Min Latency Submodular Cover problem. Finally, there has reecntly been some work on the online version of Min-Sum Set Coverby Fotakis et al. (2020); Bienkowski and Mucha (2023); Basiak et al. (2023).

3.1 Model & Definitions

In the optimal search problem, we are given a set $\mathcal{B}$ of n boxes with unknown costs and a distribution $\mathcal{D}$ over a set of possible scenarios that determine these costs. Nature chooses a scenario s from the distribution, which then instantiates the cost of each box. We use c_{is} to denote the cost of box i when scenario s is instantiated.

The goal of the online algorithm is to choose a box of small cost while spending as little time as possible gathering information. The algorithm cannot directly observe the scenario that is instantiated, however, is allowed to "probe" boxes one at a time. Upon probing a box, the algorithm gets to observe the cost of the box. Formally let $\mathcal{P}_s$ be the random variable denoting the set of probed boxes when scenario s is instantiated and let $i_s \in \mathcal{P}_s$ be the (random) index of the box chosen by the algorithm. We require $i_s \in \mathcal{P}_s$, that is, the algorithm must probe a box to choose it. Note that the randomness in the choice of $\mathcal{P}_s$ and i_s arises both from the random instantiation of scenarios as well as from any coins the algorithm itself may flip. Our goal then is to minimize the total probing time plus the cost of the chosen box:

$$\mathbf{E}_s\left[\min_{i\in\mathcal{P}_s} c_{is} + |\mathcal{P}_s|\right].$$

Any online algorithm can be described by the pair (σ, τ), where σ is a permutation of the boxes representing the order in which they get probed, and τ is a stopping rule – the time at which the algorithm stops probing and returns the minimum cost it has seen so far. Observe that in its full generality, an algorithm may choose the i'th box to probe, $\sigma(i)$, as a function of the identities and costs of the first $i-1$ boxes, $\{\sigma(1), \cdots, \sigma(i-1)\}$ and $\{c_{\sigma(1)s}, \cdots, c_{\sigma(i-1)s}\}$[1]. Likewise, the decision of setting $\tau = i$ for $i \in [n]$ may depend on $\{\sigma(1), \cdots, \sigma(i)\}$ and $\{c_{\sigma(1)s}, \cdots, c_{\sigma(i)s}\}$. Optimizing over the class of all such algorithms is intractable. So we will consider simpler classes of strategies, as formalized in the following definition.

Definition 3.1 (Adaptivity of Strategies). *In a Fully-Adaptive strategy, both σ and τ can depend on any costs seen in a previous time step, as described above.*

In a Partially-Adaptive strategy, the sequence σ is independent of the costs observed in probed boxes. The sequence is determined before any boxes are probed. However, the stopping rule τ can depend on the identities and costs of boxes probed previously.

In a Non-Adaptive strategy, both σ and τ are fixed before any costs are revealed to the algorithm. In particular, the algorithm probes a fixed subset of the boxes, $I \subseteq [n]$, and returns the minimum cost $\min_{i \in I} c_{is}$. The algorithm's expected total cost is then $\boldsymbol{E}_s[\min_{i\in I} c_{is} + |I|]$.

General feasibility constraints. In Section 3.3 we study extensions of the search problem where our goal is to pick multiple boxes satisfying a given feasibility constraint. Let $\mathcal{F} \subseteq 2^{\mathcal{B}}$ denote the feasibility constraint. Our goal is to probe boxes in some order and select a subset of the probed boxes that is feasible. Once again we can describe an algorithm using the pair (σ, τ) where σ denotes the probing order, and τ denotes the stopping time at which the algorithm stops and returns the cheapest feasible set found so far. The total cost of the algorithm then is the cost of the feasible set returned plus the stopping time. We emphasize that the algorithm faces the same feasibility constraint in every scenario. We consider two different kinds of feasibility constraints. In the first, the algorithm is required to select exactly k boxes for some $k \geqslant 1$. In the second, the algorithm is required to select a basis of a given matroid.

3.1.1 A reduction to scenario-aware strategies

Recall that designing a PA strategy involves determining a non-adaptive probing order, and a good stopping rule for that probing order. We do not place any bounds

[1]For some realized scenario $s \in \mathcal{S}$.

on the number of different scenarios, m, or the support size and range of the boxes' costs. These numbers can be exponential or even unbounded. As a result, the optimal stopping rule can be very complicated and it appears to be challenging to characterize the set of all possible PA strategies. We simplify the optimization problem by providing *extra power* to the algorithm and then removing this power at a small loss in approximation factor.

In particular, we define a *Scenario-Aware Partially-Adaptive* (*SPA*) strategy as one where the probing order σ is independent of the costs observed in probed boxes, however, the stopping time τ is a function of the instantiated scenario s. In other words, the algorithm fixes a probing order, then learns of the scenario instantiated, and then determines a stopping rule for the chosen probing order based on the revealed scenario.

Observe that once a probing order and instantiated scenario are fixed, it is trivial to determine an optimal stopping time in a scenario aware manner. The problem therefore boils down to determining a good probing order. The space of all possible SPA strategies is also likewise much smaller and simpler than the space of all possible PA strategies. We can therefore argue that in order to learn a good SPA strategy, it suffices to optimize over a small sample of scenarios drawn randomly from the underlying distribution. We denote the cost of an SPA strategy with probing order σ by $\text{cost}(\sigma)$.

On the other hand, we argue that scenario-awareness does not buy much power for the algorithm. In particular, given any fixed probing order, we can construct a stopping time that depends only on the observed costs, but that achieves a constant factor approximation to the optimal scenario-aware stopping time for that probing order.

3.1.2 Ski Rental with varying buy costs

We now define a generalized version of the ski rental problem which is closely related to SPA strategies. The input to the generalized version is a sequence of non-increasing buy costs, $a_1 \geqslant a_2 \geqslant a_3 \geqslant \dots$. These costs are presented one at a time to the algorithm. At each step t, the algorithm decides to either rent skis at a cost of 1, or buy skis at a cost of a_t. If the algorithm decides to buy, then it incurs no further costs for the remainder of the process. Observe that an offline algorithm that knows the entire cost sequences $a_1, a_2, \dots$ can pay $\min_{t \geqslant 1}(t - 1 + a_t)$. We call this problem *ski rental* *with time-varying buy costs*. The original ski rental problem is the special case where $a_t = B$ or 0 from the time we stop skiing and on.

We first provide a simple randomized algorithm for ski rental with time-varying costs that achieves a competitive ratio of $e/(e-1)$. Then we extend this to general p_t in Corollary 3.22 in the Appendix for this Chapter. Our algorithm uses the randomized algorithm of Karlin et al. (1990) for ski rental as a building block, essentially by starting a new instance of ski rental every time the cost of the offline optimum changes. The full proof of this result is included in Section 3.A.2.1.

Lemma 3.2. [*Ski Rental with time-varying buy costs*] *Consider any sequence $a_1 \geqslant a_2 \geqslant \ldots$. There exists an online algorithm that chooses a stopping time t so that*

$$t - 1 + a_t \leqslant \frac{e}{e-1} \min_j \{j - 1 + a_j\}.$$

The next corollary connects scenario-aware partially-adaptive strategies with partially-adaptive strategy through our competitive algorithm for ski-rental with time-varying costs. Specifically, given an SPA strategy, we construct an instance of the ski-rental problem, where the buy cost a_t at any step is equal to the cost of the best feasible solution seen so far by the SPA strategy. The rent cost of the ski rental instance reflects the probing time of the search algorithm, whereas the buy cost reflects the cost of the boxes chosen by the algorithm. Our algorithm for ski rental chooses a stopping time as a function of the costs observed in the past and without knowing the (scenario-dependent) costs to be revealed in the future, and therefore gives us a PA strategy for the search problem.

This result is formalized below; the proof can be found in Section 3.A.4 in this chapter's Appendix.

Corollary 3.3. *Given any* scenario-aware partially-adaptive *strategy σ, we can efficiently construct a stopping time τ, such that the cost of the* partially-adaptive *strategy (σ, τ) is no more than a factor of $e/(e-1)$ times the cost of σ.*

3.1.3 LP formulations

We will now construct an LP relaxation for the optimal scenario-aware partially adaptive strategy.

The linear program (Relaxation-SPA) is given below and is similar to the one used for the generalized min-sum set cover problem in Bansal et al. (2010) and Skutella and Williamson (2011). Denote by $\mathcal{T}$ to set of time steps. Let x_{it} be an indicator variable for whether box i is opened at time t. Constraints (3.1) and (3.2) model matching constraints between boxes and time slots. The variable z_{ist} indicates whether box i is selected for scenario s at time t. Constraints (3.3) ensure that we only select opened boxes. Constraints (3.4) ensure that for every scenario we have selected exactly one box. The cost of the box assigned to scenario s is given by $\sum_{i,t} z_{ist}c_{is}$. Furthermore, for any scenario s and time t, the sum $\sum_i z_{ist}$ indicates whether the scenario is covered at time t, and therefore, the probing time for the scenario is given by $\sum_t \sum_i tz_{ist}$.

$$
\begin{aligned}
\text{minimize} \quad & \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S},t\in\mathcal{T}} tz_{ist} + \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S},t\in\mathcal{T}} c_{is}z_{ist} && && \text{(LP-SPA)}\\
\text{subject to} \quad & \sum_{i\in\mathcal{B}} x_{it} = 1, && \forall t\in\mathcal{T} && (3.1)\\
& \sum_{t\in\mathcal{T}} x_{it} \leqslant 1, && \forall i\in\mathcal{B} && (3.2)\\
& z_{ist} \leqslant x_{it}, && \forall s\in\mathcal{S}, i\in\mathcal{B}, t\in\mathcal{T} && (3.3)\\
& \sum_{t'\in\mathcal{T},i\in\mathcal{B}} z_{ist'} = 1, && \forall s\in\mathcal{S} && (3.4)\\
& x_{it}, z_{ist} \in [0,1] && \forall s\in\mathcal{S}, i\in\mathcal{B}, t\in\mathcal{T} &&
\end{aligned}
$$

As a warmup we include an algorithm that competes with the weakest benchmark; the Non-Adaptive, in Section 3.A.1 of the Appendix.

3.2 Pandora's Box vs Partially-Adaptive

Moving on to our main result, in this section we compete against the optimal partially-adaptive strategy. Recall that the program (Relaxation-SPA) is a relaxation for the optimal SPA strategy, and therefore, also bounds from below the cost of the optimal PA strategy. We round the optimal solution to this LP to obtain a constant-competitive SPA strategy.

Given a solution to (Relaxation-SPA), we identify for each scenario a subset of low cost boxes. Our goal is then to find a probing order, so that for each scenario we quickly find one of the low cost boxes. This problem of "covering" every scenario with a low cost box is identical to the min-sum set cover (MSSC) problem introduced by Feige et al. (2002). Employing this connection allows us to convert an approximation for MSSC into an SPA strategy at a slight loss in approximation factor.

Our main result is as follows.

Lemma 3.4. *There exists a scenario-aware partially-adaptive strategy with competitive ratio* $3+2\sqrt{2}$ *against the optimal partially-adaptive strategy.*

Combining this with Corollary 3.3 we get the following theorem.

Theorem 3.5. *We can efficiently find a partially-adaptive strategy that is* $(3+2\sqrt{2})\frac{e}{e-1} = 9.22$*-competitive against the optimal partially-adaptive strategy.*

Proof of Lemma 3.4. We use the LP formulation Relaxation-SPA from Section 3.1.3. Recall that x_{it} denotes the extent to which box i is opened at time t, z_{ist} denotes the extent to which box i is chosen for scenario s at time t.

As mentioned previously, we will employ a 4-approximation to the MSSC by Feige et al. (2002) in our algorithm. The input to MSSC is an instance of set cover. In our context, the sets of the set cover are boxes and each scenario s has an element L_s corresponding to it. The goal is to find an ordering σ over the sets/boxes so as to minimize the sum of the cover times of the elements/scenarios, where the cover time of an element is the index of the first set in σ that contains it. The following is an LP relaxation for MSSC; observe its similarity to (Relaxation-SPA). Feige et al. (2002) provide a greedy algorithm that 4-approximates the optimal solution to this LP.

$$\begin{aligned}
&\text{minimize} && \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S},t\in\mathcal{T}} t z_{ist} && \text{(LP-MSSC)}\\
&\text{subject to} && (3.1)-(3.3) &&\\
& && \sum_{t'\in\mathcal{T},i\in L_s} z_{ist'} \geqslant 1, && \forall s\in\mathcal{S}\\
& && x_{it}, z_{ist} \in [0,1] && \forall s\in\mathcal{S}, i\in\mathcal{B}, t\in\mathcal{T}
\end{aligned}$$

Define $\alpha = 3 + 2\sqrt{2}$. Given an optimal solution $\mathcal{I} = (\boldsymbol{x}, \boldsymbol{z})$ to (Relaxation-SPA), we will now construct an instance $\mathcal{I}'$ of MSSC (by specifying the elements L_s[2]) with the following properties:

1. There exists an integral solution σ for $\mathcal{I}'$ with cover time at most α times the query time for $(\boldsymbol{x}, \boldsymbol{z})$.

2. Any integral solution σ for $\mathcal{I}'$ can be paired with an appropriate stopping time τ_s, so that the query time of (σ, τ_s) is at most the MSSC cover time of σ, and the cost of (σ, τ_s) is at most α times the fractional cost for $(\boldsymbol{x}, \boldsymbol{z})$.

Constructing a "good" $\mathcal{I}'$: For each scenario s, we define a set of "low" cost boxes as

$$L_s = \{i : c_{is} \leqslant \alpha \text{OPT}^{\mathcal{I}}_{c,s}\}.$$

The second property above is immediate from this definition. In particular, we define the stopping time τ_s as the first time we encounter a box $i \in L_s$.

For property (i), we first show that instance $\mathcal{I}'$ admits a good fractional solution.

While (Relaxation-SPA) allows assigning any arbitrary boxes to a scenario, (LP-MSSC) requires assigning only the boxes in L_s to scenario s. In order to convert this into a feasible solution to (LP-MSSC), we first scale up all of the variables by a factor of $\frac{\alpha}{\alpha-1}$. Specifically, set $\boldsymbol{x}' = \frac{\alpha}{\alpha-1}\boldsymbol{x}$; $z'_{ist} = \frac{\alpha}{\alpha-1} z_{ist}$ for all $s, t, i \in s$; and $z'_{ist} = 0$ for all $s, t, i \notin s$. Now we need to ensure that all the constraints of (LP-MSSC) are satisfied.

Observe initially that by Markov's inequality, for all s, $\sum_{t, i \in L_s} z_{ist} \geqslant 1 - 1/\alpha$. Therefore, by scaling the z'_{ist} as above we have that $\sum_{it} z'_{ist} \geqslant 1$ for all s. To fix (3.2), if for some $i \in \mathcal{B}$ we have $\sum_t x'_{it} > 1$, let t' be the smallest time at which $\sum_{t \leqslant t'} x'_{it} > 1$. We set $x'_{it} = 0$ for all $t > t'$ and $x'_{it'} = 1 - \sum_{t<t'} x'_{it}$. Likewise, modify $\boldsymbol{z}'$ so as to achieve (3.3) as well as ensure that every variable lies in $[0, 1]$.

It remains to argue that constraints (3.1) can be fixed at a small extra cost. Observe that for any t, $\sum_{i \in \mathcal{B}} x'_{it} \leqslant \frac{\alpha}{\alpha-1}$. We therefore "dilate" time by a factor of $\frac{\alpha}{\alpha-1}$ in order to accommodate the higher load. Formally, interpret x, x', z and z' as continuous step functions of t. Then the objective function of (LP-MSSC) can be

[2]Each element/scenario can be thought of as a set of the boxes/sets that cover it.

written as $\sum_{s,i}\int_{t=0}^{t=n}\lceil t\rceil z'_{ist}dt$[3]. Dilating time by a factor of $\frac{\alpha}{\alpha-1}$ gives us the objective $\frac{1}{|\mathcal{S}|}\sum_{s,i}\int_{t=0}^{t=n}\left\lceil t\frac{\alpha}{\alpha-1}\right\rceil z'_{ist}dt$.

Since $z'_{ist}\leqslant\frac{\alpha}{\alpha-1}z_{ist}$ for any i,s,t, the expected query time is upper bounded by

$$\begin{aligned}&\frac{1}{|\mathcal{S}|}\sum_{i,s}\int_{t=0}^{t=n}\left\lceil t\frac{\alpha}{\alpha-1}\right\rceil\cdot\frac{\alpha}{\alpha-1}z_{ist}\,dt\\ &\leqslant\frac{1}{|\mathcal{S}|}\sum_{i,s,t}\left(\frac{\alpha}{\alpha-1}\right)^2 tz_{ist}\\ &=\left(\frac{\alpha}{\alpha-1}\right)^2\cdot\text{ Query time of }(\boldsymbol{x},\boldsymbol{z})\end{aligned}$$

where for the second inequality we used the following Lemma 3.6 with $\beta=\alpha/(\alpha-1)$. The proof of the lemma is deferred to Section 3.A.4 of the appendix.

Lemma 3.6. *For any* $\beta>1$,

$$\int_{t-1}^{t}\lceil\beta t'\rceil dt'\leqslant\beta t.$$

Applying greedy algorithm for min-sum set cover. We have so far constructed a new instance $\mathcal{I}'$ of MSSC along with a feasible fractional solution $(\boldsymbol{x'},\boldsymbol{z'})$ with cover time at most $\alpha^2/(\alpha-1)^2$ times the query time for $(\boldsymbol{x},\boldsymbol{z})$. The greedy algorithm of Feige et al. (2002) finds a probing order over the boxes with query time at most 4 times the cover time of $(\boldsymbol{x'},\boldsymbol{z'})$, that is, at most $4\alpha^2/(\alpha-1)^2=\alpha$ times the query time for $(\boldsymbol{x},\boldsymbol{z})$, where the equality follows from the definition of α. Property (1) therefore holds and the lemma follows.

□

3.2.1 An Alternative, Rounding Algorithm

The algorithm of the previous section worked using a reduction to Min-Sum Set Cover, a special case of our Pandora's Box problem. In this section we show that we can obtain the same approximation guarantees using a rounding algorithm, similarly to the k items and matroid case of Section 3.3.1 and 3.3.2, that we see further. This algorithm is going to be useful in Chapter 8, where we specifically need a *rounding* algorithm to obtain our results.

[3]Note that we are not adding extra time steps, and therefore the cost of the objective does not change.

Recall that the convex relaxation for this case (Relaxation-SPA) is given in section 3.1.3. Our main lemma, shows how to obtain a constant competitive partially adaptive strategy, when given a scenario-aware solution.

Lemma 3.7. *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 $\overline{\boldsymbol{x}}$ of cost $\overline{f}^s = \overline{f}^s_o + \overline{f}^s_c$, where $\overline{f}^s_o$ and $\overline{f}^s_c$ 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 $\overline{\boldsymbol{x}}'$ of cost $\overline{f'}^s = \overline{f'}^s_o + \overline{f'}^s_c$. For this transformed solution, Chawla et al. (2020) in Lemma 5.1 showed that

$$\overline{f'}^s_o \leqslant \left(\frac{\alpha}{\alpha-1}\right)^2 \overline{f}^s_o \tag{3.5}$$

for the opening cost and

$$\overline{f'}^s_c \leqslant \alpha \overline{f}^s_c \tag{3.6}$$

for the cost incurred 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 $\overline{\boldsymbol{x}}$ 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 3.5 and 3.6). The rounding process is shown in Algorithm 1.

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

```
  Data: Fractional solution $\boldsymbol{x}$ 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. $q_{it} = \frac{\sum_{t' \leqslant t} \overline{x}_{it'}}{t}$.
2
  /* Part 2:  Scenario-dependent stopping time                 */
3 Given scenario $s$, calculate $\boldsymbol{z}^s$ and $\overline{f}^s_c$
4 $\tau_s$ := If box $i$ is opened and has value $c^s_i \leqslant \alpha \overline{f}^s_c$ then select it.
```

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

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

$$\begin{aligned}\frac{f_o^s}{\overline{f'}_o^s} &\leqslant \frac{2\sum_{t=1}^{\infty}\prod_{k=1}^{t-1}\left(1-\frac{\sum_{i\in A,t'\leqslant k} z_{it'}^s}{k}\right)^2}{\sum_i t\cdot z_{it}^s} && \text{Since } z_{it}^s \leqslant x_{it}\\ &\leqslant \frac{2\sum_{t=1}^{\infty}\exp\left(-2\sum_{k=1}^{t-1}\frac{\sum_{t'\leqslant k} z_{it'}^s}{k}\right)}{\sum_i t\cdot z_{it}^s} && \text{Using that } 1+x\leqslant e^x\end{aligned}$$

Observe that $h(z) = \log\frac{f_o^s}{\overline{f'}_o^s} = \log f_o^s - \log \overline{f}_o^s$ is a convex function since the first part is LogSumExp, and $\log \overline{f}_o^s$ 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 $z_{it}^s = 1$ iff $t = \ell$ for some $\ell \in [n]$, otherwise $z_{it}^s = 0$. Using this fact we obtain

$$\begin{aligned}\frac{f_o^s}{\overline{f'}_o^s} &\leqslant \frac{2\ell + 2\sum_{t=\ell+1}^{\infty}\exp\left(-2\sum_{k=\ell}^{t-1}\frac{1}{k}\right)}{\ell} && \text{Using that } z_{it}^s = 1 \text{ iff } t=\ell\\ &= \frac{2\ell + 2\sum_{t=\ell+1}^{\infty}\exp\left(H_{t-1}-H_{\ell-1}\right)}{\ell} && H_t \text{ is the t'th harmonic number}\\ &\leqslant \frac{2\ell + 2\sum_{t=\ell+1}^{\infty}\left(\frac{\ell}{t}\right)^2}{\ell} && \text{Since } H_{t-1}-H_{\ell-1} \geqslant \int_\ell^t \frac{1}{x}dx = \log t - \log \ell\\ &\leqslant \frac{2\ell + 2\ell^2\int_\ell^\infty \frac{1}{t^2}dt}{\ell} && \text{Since } t^{-2}\leqslant x^{-2} \text{ for } x\in[t-1,t]\\ &= 4.\end{aligned}$$

Combining with equation 3.5, we get that $f_o^s \leqslant 4\left(\frac{\alpha}{\alpha-1}\right)^2 \overline{f}_o^s$. Recall that for the values cost, inequality (3.6) holds, therefore requiring that $4\left(\frac{\alpha}{\alpha-1}\right)^2 = \alpha$, we have the lemma for $\alpha = 3+2\sqrt{2}$. □

Corollary 3.8. *For the case of MSSC, when the costs inside the boxes are either* 0 *or* ∞, *the rounding of Lemma* 3.7 *obtains a* 4*-approximation, improving the* 11.473 *of Fotakis et al. (2020).*

3.2.2 A lower bound against the PA

To give the readers a sense of how good the guarantees that we gave are, we observe that the lower bound of Min-Sum Set Cover presented in Feige et al. (2004) also applies to Pandora's Box. In Min-Sum Set Cover we are given n elements e_i, and m sets s_j where each $s_j \subseteq [n]$. We say a set s_j *covers* an element e_i if $e_i \in s_j$. The goal is to select elements in order to minimize the sum of the *covering times* of all the sets, where *covering time* of a set is the first time an element $e_i \in s_j$ is chosen. This lower bound is also mentioned in Chawla et al. (2020), but we include it here with more details for the sake of completeness.

In Feige et al. (2004) the authors show that Min-Sum Set Cover cannot be approximated better than $4 - \varepsilon$ even in the special case where every set contains the same number of elements[5]. We restate the theorem below.

Theorem 3.9 (Theorem 13 of Feige et al. (2004))**.** *For every $\varepsilon > 0$, it is NP-hard to approximate min sum set cover within a ratio of $4 - \varepsilon$ on uniform hypergraphs.*

Our main observation is that Min-Sum Set Cover is a special case of Pandora's Box. When the boxes all have the same opening cost $c_b = 1$ and the values inside are $v_s^b \in \{0, \infty\}$, we are required to find a 0 for each scenario; equivalent to *covering* a scenario. The optimal solution of Min-Sum Set Cover is an algorithm that selects elements one by one, and stops whenever all the sets are covered. This is exactly the partially adaptive optimal we defined for Pandora's Box. The theorem restated above results in the following Corollary.

Corollary 3.10. *For every $\varepsilon > 0$ it is NP-Hard to approximate Pandora's Box against the partially-adaptive within a ratio better than $4 - \varepsilon$.*

3.3 More complex constraints

In this section we extend the problem in cases where there is a feasibility constraint $\mathcal{F}$, that limits what or how many boxes we can choose. We consider the cases where we are required to select k distinct boxes, and k independent boxes from a matroid. In both cases we design SPA strategies that can be converted to PA. These two variants are described in more detail in subsections 3.3.1 and 3.3.2 that follow.

[5]Equivalently forms a uniform hypergraph, where sets are hyperedges, and elements are vertices.

3.3.1 Choosing k items

In this section $\mathcal{F}$ requires that we pick k boxes to minimize the total cost and query time. We again aim to compete against the optimal Partially-Adaptive strategy. We design a PA strategy which achieves an $O(1)$-competitive ratio. If $c_{is} \in \{0, \infty\}$, the problem is the generalized min-sum set cover problem first introduced in Azar et al. (2009). Azar et al. (2009) gave a $\log n$-approximation, which then was improved to a constant in Bansal et al. (2010) via an LP-rounding based algorithm. Our proof will follow the latter proof in spirit, and generalize to the case where boxes have arbitrary values. Our main result is the following.

Lemma 3.11. *There exists a scenario-aware partially-adaptive* $O(1)$*-competitive algorithm to the optimal partially-adaptive algorithm for picking* k *boxes.*

Combining this with Corollary 3.3 we get the following theorem.

Theorem 3.12. *We can efficiently find a partially-adaptive strategy for optimal search with* k *options that is* $O(1)$*-competitive against the optimal partially-adaptive strategy.*

Proof of Lemma 3.11. The LP formulation we use for this problem is a variant of (Relaxation-SPA) from Section 3.1.3, with the following changes; we introduce variable y_{st} which denotes the extent to which scenario s is covered until time t and constraint (3.4) is replaced by constraint (3.8). For the LP to reflect this additional cost we modify constraint (3.1) to (3.7) so that a box now is probed for p_i steps. The program (LP-k-cover) is presented below. Denote by $\text{OPT}_{t,s}$ and $\text{OPT}_{c,s}$ the contribution of the query time and cost of scenario s in optimal fractional solution. $\text{ALG}_{t,s}$ and $\text{ALG}_{c,s}$ denote the corresponding quantities for the algorithm.

$$\text{minimize} \quad \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S},t\in\mathcal{T}}(1-y_{st}) \quad + \quad \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S},t\in\mathcal{T}} c_{is}z_{ist} \qquad \text{(LP-k-cover)}$$

$$\text{subject to} \quad \sum_{i\in\mathcal{B}} x_{it} = 1, \qquad \forall t\in\mathcal{T}$$

$$\sum_{t\in\mathcal{T}} x_{it} \leqslant 1, \qquad \forall i\in\mathcal{B} \qquad (3.7)$$

$$z_{ist} \leqslant x_{it}, \qquad \forall s\in\mathcal{S}, i\in\mathcal{B}, t\in\mathcal{T}$$

$$\sum_{t'\leqslant t, i\notin A} z_{ist'} \geqslant (k-|A|)y_{st}, \qquad \forall A\subseteq \mathcal{B}, s\in\mathcal{S}, t\in\mathcal{T} \tag{3.8}$$

$$x_{it}, \quad z_{ist}, y_{st} \in [0,1] \qquad \forall s\in\mathcal{S}, i\in\mathcal{B}, t\in\mathcal{T}$$

The LP formulation we use is exponential in size but we can efficiently find a separation oracle, as observed in Section 3.1 in Bansal et al. (2010).

We claim that Algorithm 2 satisfies the lemma. The algorithm first finds an opening sequence by probing boxes with some probability at every step, and then select every opened box with some probability until k boxes are probed. Note that the number of boxes probed at each "step" may be more than one. In the algorithm, we set constant $\alpha = 8$.

Algorithm 2: SPA vs PA, k-coverage

```
  Data: Solution $x, y, z$ to above LP, scenario $s$
1 $\sigma :=$ For each phase $\ell = 1, 2, \ldots$, open each box $i$ independently with
   probability $q_{i\ell} = \min\left(\alpha \sum_{t\leqslant 2^\ell} x_{it}, 1\right)$.
2 $\tau_s :=$
3   Define $t^*_s = \max\{t : y_{st} \leqslant 1/2\}$.
4   if $2^\ell \geqslant t^*_s$ then
5     For each opened box $i$, select it with probability $\min\left(\frac{\alpha \sum_{t\leqslant 2^\ell} z_{ist}}{q_{i\ell}}, 1\right)$.
6     Stop when we have selected $k$ boxes in total.
7 end
```

Let t^*_s be the latest time at which $y_{st} \leqslant 1/2$ as in the description of the algorithm. As observed in Bansal et al. (2010) for scenario s, we pay at least $1 - y_{st^*_s} \geqslant \frac{1}{2}$ for each time $t \in [1, t^*_s]$, thus

$$\mathrm{OPT}_{t,s} \geqslant \frac{t^*_s}{2}. \tag{3.9}$$

Fix a scenario s. We first analyze the expected probing time of the algorithm for this scenario. Denote by $\ell_0 = \lceil \log t^*_s \rceil$ the first phase during which we have a non-zero probability of selecting a box for scenario s. Notice that for each box i, the probability that it is selected in phase $\ell \geqslant \ell_0$ is $\min(1, 8\sum_{t'\leqslant 2^\ell} z_{ist'})$. The following lemma from Bansal et al. (2010) bounds the probability that in each phase ℓ such that $2^\ell \geqslant t^*_s$, at least k boxes are selected.

Lemma 3.13 (Lemma 5.1 in Bansal et al. (2010))**.** *If each box i is selected w.p. at least $\min(1, 8\sum_{t'\leqslant t} z_{ist'})$ for $t \geqslant t^*_s$, then with probability at least $1-e^{-9/8}$, at least k different boxes are selected.*

Let $\gamma = e^{-9/8}$. Observe that the number of boxes probed in a phase is independent of the event that the algorithm reaches that phase prior to covering scenario s, therefore we get

$$\begin{aligned}
\mathbf{E}\left[\text{query time after phase } \ell_0\right] &= \sum_{\ell=\ell_0}^{\infty} \mathbf{E}\left[\text{query time phase } \ell\right] \cdot \mathbf{Pr}\left[\text{ALG reaches phase } \ell\right] \\
&\leqslant \sum_{\ell=\ell_0}^{\infty} \sum_{i\in\mathcal{B}} \alpha \sum_{t'\leqslant 2^\ell} x_{it'} \cdot \prod_{j=\ell_0}^{\ell-1} \mathbf{Pr}\left[\leqslant k \text{ boxes selected phase } j\right] \qquad (3.10)\\
&\leqslant \sum_{\ell=\ell_0}^{\infty} 2^\ell \alpha \cdot \gamma^{\ell-\ell_0} \\
&= \frac{2^{\ell_0}\alpha}{1-2\gamma} < \frac{2t^*_s\alpha}{1-2\gamma} \leqslant \frac{4\alpha \mathrm{OPT}_{t,s}}{1-2\gamma}.
\end{aligned}$$

The second line follows by noting that the algorithm can reach phase ℓ only if in each previous phase there are less than k boxes selected. The third line is by Lemma 3.13 and constraint (3.7). The last line is by $\ell_0 = \lceil \log t^*_s \rceil$ and inequality (3.9). Since the expected query time at each phase ℓ is at most $\alpha 2^\ell$, thus the expected query time before phase ℓ_0 is at most $\sum_{\ell<\ell_0} \alpha 2^\ell < 2^{\ell_0}\alpha < 2t^*_s\alpha \leqslant 4\alpha\mathrm{OPT}_{t,s}$. Therefore the total query time of the algorithm for scenario s is

$$\mathrm{ALG}_{t,s} \leqslant 4\alpha\mathrm{OPT}_{t,s} + \frac{4\alpha\mathrm{OPT}_{t,s}}{1-2\gamma} < 123.25\mathrm{OPT}_{t,s}.$$

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.

$$\begin{aligned}
&\mathbf{E}[\text{cost in phase } \ell | \text{at least } k \text{ boxes are selected in phase } \ell] \\
&\qquad \leqslant \frac{\mathbf{E}\left[\text{cost in phase } \ell\right]}{\mathbf{Pr}\left[\text{at least } k \text{ boxes are selected in phase } \ell\right]}
\end{aligned}$$

$$
\begin{aligned}
&\leqslant \frac{1}{1-\gamma}\mathbf{E}\left[\text{cost in phase } \ell\right] \\
&\leqslant \frac{1}{1-\gamma}\sum_{i\in\mathcal{B}}\alpha\sum_{t\leqslant 2^{\ell}} z_{ist}c_{is} = \frac{1}{1-\gamma}\alpha \mathrm{OPT}_{c,s} < 11.85\mathrm{OPT}_{c,s}.
\end{aligned}
$$

Here the third line is by Lemma 3.13 and the last line is by definition of $\mathrm{OPT}_{c,s}$. Notice that the upper bound does not depend on the phase ℓ, so the same upper bound holds for $\mathrm{ALG}_{c,s}$. Thus the total cost contributed from scenario s in our algorithm is

$$
\mathrm{ALG}_s = \mathrm{ALG}_{t,s} + \mathrm{ALG}_{c,s} < 123.25\mathrm{OPT}_{t,s} + 11.85\mathrm{OPT}_{c,s} \leqslant 123.25\mathrm{OPT}_s.
$$

Taking the expectation over all scenarios s, we conclude that the scenario-aware strategy gives constant competitive ratio to the optimal partially-adaptive strategy. □

3.3.2 Choosing matroid basis of rank k

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 base of size k with the minimum cost and the minimum query time. We first design a scenario-aware partially-adaptive strategy in Lemma 3.14 that is $O(\log k)$-competitive against optimal partially-adaptive strategy. Then, in Theorem 3.17 we argue that such competitive ratio is asymptotically tight.

Lemma 3.14. *There exists a scenario-aware partially-adaptive $O(\log k)$-approximate algorithm to the optimal partially-adaptive algorithm for picking a matroid basis of rank k.*

Combining this with Corollary 3.3 we get the following theorem.

Theorem 3.15. *We can efficiently find a partially-adaptive strategy for optimal search over a matroid of rank k that is $O(\log k)$-competitive against the optimal partially-adaptive strategy.*

The LP formulation is similar to the one for the k-coverage constraint, presented in the previous section. Let $r(A)$ for any set $A \subseteq \mathcal{B}$ denote the rank of this set. The constraints are the same except for constraints (3.11) and (3.12) that ensure we select no more than the rank of a set and that the elements that remain unselected are adequate for us to cover the remaining rank respectively.

$$
\text{minimize} \quad \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S},t\in\mathcal{T}}(1-y_{st}) \quad + \quad \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S},t\in\mathcal{T}} c_{si}z_{ist} \qquad \text{(LP-matroid)}
$$

$$
\begin{array}{llll}
\text{subject to} & \sum_{i \in \mathcal{B}} x_{it} & = 1, & \forall t \in \mathcal{T} \\
& \sum_{t \in \mathcal{T}} x_{it} & \leqslant 1, & \forall i \in \mathcal{B} \\
& \sum_{t \in \mathcal{T}, i \in A} z_{ist} & \leqslant r(A), & \forall s \in \mathcal{S}, A \subseteq \mathcal{B}
\end{array}
\tag{3.11}
$$

$$
\begin{array}{lll}
z_{ist} & \leqslant x_{it}, & \forall s \in \mathcal{S}, i \in \mathcal{B}, t \in \mathcal{T} \\
\sum_{i \notin A} \sum_{t' \leqslant t} z_{ist'} & \geqslant (r([n]) - r(A)) y_{st}, & \forall A \subseteq \mathcal{B}, s \in \mathcal{S}, t \in \mathcal{T}
\end{array}
\tag{3.12}
$$

$$
x_{it}, \quad z_{ist}, y_{st} \in [0,1] \qquad \forall s \in \mathcal{S}, i \in \mathcal{B}, t \in \mathcal{T}
$$

Solving the LP efficiently The LP formulation we use is exponential in size but we can efficiently find a separation oracle. Every set of constraints can be verified in polynomial time except for constraints (3.12). Rewriting these last constraints we get

$$
\sum_{i} \sum_{t' \leqslant t} z_{ist'} - \sum_{i \in A} \sum_{t' \leqslant t} z_{ist'} \geqslant r([n]) - r(A), \quad \forall A \subseteq \mathcal{B}, t \in \mathcal{T}.
$$

Then the problem is equivalent to minimizing the function

$$
g(A) = r(A) - \sum_{i \in A} \sum_{t' \leqslant t} z_{ist'}
$$

over all subsets of items $A \subseteq \mathcal{B}$. The function $g(A)$ is submodular since the rank function $r(A)$ is submodular, therefore we can minimize it in polynomial time Grötschel et al. (1981). The formal statement of the main theorem is the following.

Proof of Lemma 3.14. We claim that Algorithm 3 satisfies the lemma. The algorithm first finds an opening sequence by probing boxes with some probability at every step, and then knowing the scenario selects every opened box with some probability until a basis of rank k is found. In the algorithm we set constant $\alpha = 64$.

Algorithm 3: SPA vs PA, matroid

```
  Data: Solution $x, y, z$ to above LP, scenario $s$
1 $\sigma :=$ for every $t = 1, \dots, n$, open each box $i$ independently with probability
    $q_{it} = \min\left\{\alpha \ln k \frac{\sum_{t' \leqslant t} x_{it'}}{t}, 1\right\}$.
2 $\tau_s :=$
3   Let $t^*_s = \min\{t : y_{st} \leqslant 1/2\}$.
4   if $t > t^*_s$ then
5     For each opened box $i$, select it with probability $\min\left\{\frac{\alpha \ln k \sum_{t' \leqslant t} z_{ist'}}{t q_{it}}, 1\right\}$.
6     Stop when we find a base of the matroid.
7 end
```

In scenario s, let phase ℓ be when $t \in (2^{\ell-1}t^*_s, 2^\ell t^*_s]$. The proof has almost identical flow as the proof for k-coverage case. We still 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 query time needed in each phase to get a full rank base of the matroid. The proof is deferred to Section 3.A.4 of the appendix.

Lemma 3.16. *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 $\ell \geqslant 6$ is

$$\mathbf{Pr}\left[\mathcal{X} \leqslant 2^{\ell-1}t^*_s\right] \geqslant 1 - \frac{\mathbf{E}\left[\mathcal{X}\right]}{2^{\ell-1}t^*_s} \geqslant 1 - \frac{1}{2^{\ell-1}t^*_s}(4 + 2^{\ell+2}/\alpha)t^*_s = 1 - 2^{3-\ell} - \frac{8}{\alpha} \geqslant \frac{3}{4}, \tag{3.13}$$

where we used Markov's inequality for the first inequality and Lemma 3.16 for the second inequality. To calculate the total query time, we sum up the contribution of all phases.

$$\mathbf{E}\left[\text{query time aftr phase 6}\right] = \sum_{\ell=6}^{\infty} \mathbf{E}\left[\text{query time at phase } \ell\right] \cdot \mathbf{Pr}\left[\text{ALG reaches phase } \ell\right]$$

$$\begin{aligned}
&\leqslant \sum_{\ell=6}^{\infty} \sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \sum_{i\in\mathcal{B}} \alpha \ln k \cdot \frac{\sum_{t'\leqslant t} x_{it'}}{t} \left(\frac{1}{4}\right)^{\ell-6} \qquad (3.14)\\
&\leqslant \sum_{\ell=6}^{\infty} 2^{\ell-1} t_s^* \alpha \ln k \cdot \left(\frac{1}{4}\right)^{\ell-6}\\
&= \frac{128\alpha \ln k t_s^*}{3} \leqslant \frac{256c \ln k \mathrm{OPT}_{t,s}}{3}.
\end{aligned}$$

Here the second line uses that each box i is probed at each time step t with probability $\alpha \ln k \cdot \frac{\sum_{t'\leqslant t} x_{it'}}{t}$. The third line follows from constraint (3.7). The last line uses $t_s^* \leqslant 2\mathrm{OPT}_{t,s}$ by (3.9). Since the expected query time at each step is $\alpha \ln k$ and there are $2^5 t_s^* \leqslant 64\mathrm{OPT}_{t,s}$ steps before phase 6, we have

$$\mathrm{ALG}_{t,s} \leqslant \alpha \ln k \cdot 64\mathrm{OPT}_{t,s} + \frac{256\alpha \ln k \mathrm{OPT}_{t,s}}{3} = O(\log k)\mathrm{OPT}_{t,s}.$$

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

$$\begin{aligned}
&\mathbf{E}[\text{cost in phase } \ell | \text{full rank base selected in phase } \ell]\\
&\qquad \leqslant \frac{\mathbf{E}\left[\text{cost in phase } \ell\right]}{\mathbf{Pr}\left[\text{full rank base selected in phase } \ell\right]}\\
&\qquad \leqslant \frac{1}{3/4}\mathbf{E}\left[\text{cost in phase } \ell\right]\\
&\qquad \leqslant \frac{1}{3/4} \sum_{i\in\mathcal{B}} \sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \alpha \ln k \frac{\sum_{t'\leqslant t} z_{ist'} c_{is}}{t}\\
&\qquad \leqslant \frac{1}{3/4} \sum_{t=2^{\ell-1}t_s^*+1}^{2^\ell t_s^*} \alpha \ln k \sum_{i\in\mathcal{B}} \frac{\sum_{t'\in\mathcal{T}} z_{ist'} c_{is}}{2^{\ell-1} t_s^*}\\
&\qquad = \frac{1}{3/4} \alpha \ln k \mathrm{OPT}_{c,s} = O(\log k)\mathrm{OPT}_{c,s}.
\end{aligned}$$

Such upper bound of conditional expectation does not depend on ℓ, thus also gives the same upper bound for $\mathrm{ALG}_{c,s}$. Therefore $\mathrm{ALG}_s = \mathrm{ALG}_{t,s} + \mathrm{ALG}_{c,s} \leqslant O(\log k)(\mathrm{OPT}_{t,s} + \mathrm{OPT}_{c,s}) = O(\log k)\mathrm{OPT}_s$. Take expectation over s, we have the scenario-aware adaptive strategy Algorithm 3 is $O(\log k)$-competitive against the optimal partially-adaptive strategy. □

Now we argue that the $O(\log k)$-approximation we got is essentially tight. The following theorem implies that under common complexity assumption, no efficient fully-adaptive algorithm can get asymptotically better competitive ratio, even compared to optimal non-adaptive cost.

Theorem 3.17. *Assuming NP$\not\subseteq$RP, no computationally efficient fully-adaptive algorithm can approximate the optimal non-adaptive cost within a factor of* $o(\log k)$.

Proof. We provide an approximation-preserving reduction from Set Cover problem to finding good fully-adaptive strategy. Let $\mathcal{SC} = ([n], \{S_1, \dots, S_k\})$ be a Set Cover instance on a ground set of n elements, and k sets $S_1, \dots, S_k$. Denote by OPT_{SC} the optimal solution to this Set Cover instance. We construct an instance of partition matroid coverage, where the rank is k. Each segment of the partition consists of multiple copies of the sets $S_1, \dots, S_k$. Every scenario consists of one set from each segment, as seen in Table 3.1.

	Segment 1	**Segment** 2	...	**Segment** k
Scenario 1	S_1	S_1	...	S_1
Scenario 2	S_1	S_1		S_2
...	...	...	...	...
Scenario k^k	S_k	S_k		S_k

Table 3.1: Instance of partition matroid k-coverage

A scenario is covered when k elements are selected, one for each of the k sets of every segment. This is an instance of the probing problem we study with cost for each box being 0 or ∞. Similarly, we say a segment is *covered* when we have chosen at least one element in every set it contains. Denote by ALG_{FA} and OPT_{NA} the solution of any fully-adaptive algorithm and the optimal non-adaptive solution respectively for this transformed instance. Any fully-adaptive algorithm will select elements, trying to cover all scenarios. Initially, observe that $\text{OPT}_{NA} \leqslant k\text{OPT}_{SC}$, since the non-adaptive will at most solve the Set Cover problem in the k different segments. We assume that we can approximate the non-adaptive strategy with competitive ratio α i.e. $\text{ALG}_{FA} = O(\alpha)\text{OPT}_{NA} = O(k\alpha)\text{OPT}_{SC}$.

Let ℓ be the number of elements ALG_{FA} has selected when exactly $k/2$ segments are covered and let s be a randomly chosen scenario. For each one of the $k/2$ uncovered segments, there are at least 1 uncovered set. Therefore

$$\mathbf{Pr}\left[s \text{ is uncovered}\right] \geqslant 1 - \left(1 - \frac{1}{k}\right)^{k/2} \approx 1 - \frac{1}{\sqrt{e}}.$$

This implies $\mathrm{ALG}_{\mathrm{FA}} \geqslant \ell\left(1 - \frac{1}{\sqrt{e}}\right)$ thus $\ell = O(k\alpha)\mathrm{OPT}_{\mathrm{SC}}$. Notice that there exists some segment that is covered using $\ell/(k/2) = O(\alpha)\mathrm{OPT}_{\mathrm{SC}}$ elements. Thus any efficient algorithm that provides $O(\alpha)$-approximation of non-adaptive strategy using fully-adaptive strategy can be transformed efficiently to an $O(\alpha)$-approximation algorithm for Set Cover.

Although above reduction from set cover has k^k scenarios that cannot be constructed in polynomial time, by Lemma 7.1 $\mathrm{poly}(n, \frac{1}{\epsilon}, \log\frac{1}{\delta})$ samples of all scenarios is sufficient to get accuracy within ϵ with probability $1-\delta$ for any probing strategy. Let $\varepsilon = 1$ and $\delta = \frac{1}{3}$. The above reduction implies that if there is a poly-time algorithm that computes a probing strategy with cost $o(\log k)\mathrm{OPT}_{\mathrm{NA}}$, there exists a poly-time algorithm to solve Set Cover with competitive ratio $o(\log k)$ with probability $\frac{2}{3}$. By Dinur and Steurer (2014) such algorithm cannot exist assuming NP$\not\subseteq$RP. □

3.A Appendix for Chapter 3

3.A.1 Warm-up: Competing with the Non-Adaptive

As a warm-up for our main result, we approximate the optimal NA strategy by a PA strategy. The relaxation (LP-NA) for the optimal NA strategy is simpler. Here x_i is an indicator variable for whether box i is opened and z_{is} indicates whether box i is assigned to scenario s.

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in\mathcal{B}} x_i + \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B}, s\in\mathcal{S}} c_{is} z_{is} & & \text{(LP-NA)}\\
\text{subject to} \quad & \sum_{i\in\mathcal{B}} z_{is} = 1, & \forall s \in \mathcal{S} \quad & (3.15)\\
& z_{is} \leqslant x_i, & \forall i \in \mathcal{B}, s\in\mathcal{S} \quad &\\
& x_i, z_{is} \in [0,1] & \forall i \in \mathcal{B}, s\in\mathcal{S} \quad &
\end{aligned}$$

3.A.1.1 An upper bound via PA strategies

Our main result of this section is as follows.

Lemma 3.18. *We can efficiently compute a scenario-aware partially-adaptive strategy with competitive ratio* 1 *against the optimal non-adaptive strategy.*

Putting this together with Corollary 3.3 we get the following theorem.

Theorem 3.19. *We can efficiently find a partially-adaptive strategy with total expected cost at most* $e/(e-1)$ *times the total cost of the optimal non-adaptive strategy.*

Proof of Lemma 3.18. We use the LP relaxation (LP-NA) described above. Given an optimal fractional solution $(\boldsymbol{x}, \boldsymbol{z})$, we denote by $\text{OPT}_{c,s} = \sum_i c_{is} z_{is}$ the cost for scenario s in this solution, and by $\text{OPT}_c = \frac{1}{|\mathcal{S}|}\sum_s \text{OPT}_{c,s}$ the cost for all scenarios. Let $\text{OPT}_t = \sum_{i \in \mathcal{B}} x_i$ denote the probing time for the fractional solution. Similarly, we define ALG_t, ALG_c and $\text{ALG}_{c,s}$ to be the algorithm's query time, cost for all scenarios and cost for scenario s respectively.

Algorithm 4 rounds $(\boldsymbol{x}, \boldsymbol{z})$ to an SPA strategy. Note that the probing order σ in the rounded solution is independent of the instantiated scenario, but the stopping time τ_s depends on the scenario specific variables z_{is}. τ_s is not necessarily the optimal stopping time for the constructed probing order, but its definition allows us to relate the cost of our solution to the fractional cost OPT_c.

Algorithm 4: SPA vs NA

```
   Data: Solution $x, z$ to program (LP-NA); scenario $s$
1  $\sigma$ := For $t \geqslant 1$, 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 $\frac{z_{is}}{x_i}$.
```

Notice that for each step t, the probability of stopping is

$$\textbf{Pr}\left[\text{stop at step } t\right] = \sum_{i \in \mathcal{B}} \frac{x_i}{\sum_{i \in \mathcal{B}} x_i} \frac{z_{is}}{x_i} = \frac{\sum_{i \in \mathcal{B}} z_{is}}{\sum_{i \in \mathcal{B}} x_i} = \frac{1}{\text{OPT}_t},$$

where we used the first set of LP constraints (3.15) and the definition of OPT_t. Observe that the probability is independent of the step t and therefore $\mathbf{E}\left[\text{ALG}_t\right] = \text{OPT}_t$. The expected cost of the algorithm is

$$\begin{aligned}\mathbf{E}\left[\text{ALG}_{c,s}\right] &= \sum_{i\in\mathcal{B},t} \mathbf{Pr}\left[\text{select } i \text{ at step } t \mid \text{stop at step } t\right] \mathbf{Pr}\left[\text{stop at step } t\right] c_{is} \\ &\leqslant \sum_{i\in\mathcal{B},t} \frac{z_{is}}{\sum_{i\in\mathcal{B}} z_{is}} \mathbf{Pr}\left[\text{stop at step } t\right] c_{is} = \sum_{i\in\mathcal{B}} z_{is} c_{is} = \text{OPT}_{c,s}\end{aligned}$$

Taking expectation over all scenarios we get $\mathbf{E}\left[\text{ALG}_c\right] \leqslant \text{OPT}_c$, and the lemma follows. □

3.A.1.2 A Lower Bound Against the Non-Adaptive

We now show that we cannot achieve a competitive ratio of 1 against the optimal NA strategy even if we use the full power of fully adaptive strategies.

Theorem 3.20. *Assuming P≠NP, no computationally efficient fully-adaptive algorithm can approximate the optimal non-adaptive strategy within a factor smaller than* 1.278*.*

Our lower bound is based on the hardness of approximating Set Cover. We use the following lemma which rules out bicriteria results for Set Cover; a proof can be found in Appendix 3.A.4.

Lemma 3.21. *Unless P=NP, for any constant* $\epsilon > 0$*, there is no algorithm that for every instance of Set Cover finds* k *sets that cover at least* $1 - \left(1 - \frac{1+\varepsilon}{OPT}\right)^k$ *of the elements for some integer* $k \in \left[1, \frac{\log n}{1+\varepsilon} OPT\right]$*.*

Proof of Theorem 3.20. Let $H > 0$ and $p \in [0, 1]$ be appropriate constants, to be determined later. We will define a family of instances of the optimal search problem based on set cover. Let $\mathcal{SC} = ([m], \{S_1, \dots, S_n\})$ be a set cover instance with m elements and n sets. Denote its optimal value by OPT_{SC}. To transform this into an instance of the search problem, every element $e_j \in [m]$ corresponds to a scenario j, and every set S_i to a box i. We set $c_{ij} = 0$ iff $e_j \in S_i$, otherwise $c_{ij} = H$. We also add a new scenario X with $v_{Xi} = H$, $\forall i \in [n]$. Scenario X occurs with probability p and all the other m scenarios happen with probability $(1-p)/m$ each.

In this instance, the total cost of optimal non-adaptive strategy is $\text{OPT}_{NA} \leqslant pH + \text{OPT}_{SC}$, since we may pay the set-cover cost to cover all scenarios other than X, and pay an additional cost H to cover X.

Consider any computationally efficient algorithm $\mathcal{A}$ that returns a fully adaptive strategy for such an instance. Since the costs of the boxes are 0 or H, we may assume without loss of generality that any FA strategy stops probing as soon as it observes a box with cost 0 and chooses that box. We say that the strategy covers a scenario when it finds a box of cost 0 in that scenario. Furthermore, prior to finding a box with cost 0, the FA strategy learns no information about which scenario it is in other than that the scenario is as yet uncovered. Consequently, the strategy follows a fixed probing order that is independent of the scenario that is instantiated. We can now convert such a strategy into a bicriteria approximation for the underlying set cover instance. In particular, for $k \in [n]$, let r_k denote the number of scenarios that are covered by the first k probed boxes. Then, we obtain a solution to $\mathcal{SC}$ with k sets covering r_k elements. By Lemma 3.21 then, for every $\varepsilon > 0$, there must exist an instance of set cover, $\mathcal{SC}$, and by extension an instance of optimal search, on which $\mathcal{A}$ satisfies $r_k \leqslant 1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^{k-1}$ for all $k \leqslant \frac{\log n}{(1+\varepsilon)}\mathrm{OPT}_{SC}$.

For the rest of the argument, we focus on that hard instance for $\mathcal{A}$. Let N denote the maximum number of boxes $\mathcal{A}$ probes before stopping to return a box of cost H.[6]

Then the expected query time of the strategy is at least

$$\begin{aligned}
\mathbf{Pr}[s = X] \cdot N + \mathbf{Pr}[s \neq X] \sum_{k=1}^{N} \mathbf{Pr}[\text{FA reaches step } k | s \neq X] \\
\geqslant pN + (1-p)\sum_{k=1}^{N}\left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^{k-1} \\
= pN + (1-p)\left(1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^{N}\right)\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}.
\end{aligned} \tag{3.16}$$

On the other hand, the expected cost of the FA strategy is at least

$$\begin{aligned}
H(\mathbf{Pr}[s = X] + \mathbf{Pr}[s \neq X \wedge \text{FA didn't find cost 0 in first N steps }]) \\
\geqslant pH + (1-p)H\left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^{N}.
\end{aligned}$$

[6]We may safely assume that $N \leqslant \log n \mathrm{OPT}_{SC}$.

Thus the total cost of such fully-adaptive strategy is lower bounded by

$$\begin{aligned}\mathrm{ALG}_{FA} \geqslant\ & pH + (1-p)H\left(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^N \\ & + pN + (1-p)\left(1-\left(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^N\right)\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}.\end{aligned}$$

Let x be defined so that $(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}})^N = e^{-x}$. Then, $N = -x/\ln(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}}) \geqslant x(\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}-1)$. Substituting these expressions in the above equation we get

$$\mathrm{ALG}_{FA} \geqslant pH + (1-p)He^{-x} + p\cdot x\left(\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}-1\right) + (1-p)(1-e^{-x})\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}.$$

The RHS is minimized at $x = \ln\left(\frac{(1-p)(H(1+\varepsilon)-\mathrm{OPT}_{SC})}{p(\mathrm{OPT}_{SC}-(1+\varepsilon))}\right)$. By setting $\epsilon \to 0$, $p = 0.22$ and $H = 4.59\mathrm{OPT}_{SC}$, the competitive ratio becomes

$$\frac{\mathrm{ALG}_{FA}}{\mathrm{OPT}_{NA}} \geqslant 1.278$$

when $\mathrm{OPT}_{SC} \to \infty$. □

3.A.2 General Probing Times: Revisiting the Main Results

In this section, we consider settings where different boxes require different amounts of time to probe. Let p_i denote the probing time required to probe box i. We assume $p_i \in [1, P]$ for some P that is polynomially large in n. The running time and sample complexity of our algorithms will depend linearly on P. Henceforth we will assume that the p_i's are integers: rounding up each probing time to the next integer only increases the total objective function value by a factor of at most 2.

3.A.2.1 Ski rental with general rent cost

In order to show Corollary 3.3 in our case, we need to solve a further generalization of the ski rental problem where we have arbitrary rent costs. Specifically, in the ski rental problem with general rent cost, the input is a sequence of non-increasing buy costs, $a_1 \geqslant a_2 \geqslant a_3 \geqslant \dots$ as well as an integral rent costs p_t for each time t. At each step t, the algorithm decides to either rent skis at a cost of p_t, or buy skis at a cost of a_t. We show that Lemma 3.2 still holds beyond the unit-rental-cost case.

Lemma 3.22. *Consider any sequence of integral buy cost* $a_1 \geqslant a_2 \geqslant \ldots$ *and integral rent cost* $p_1, p_2, \cdots$. *There exists an online algorithm that chooses a stopping time* t *so that*

$$\sum_{i=1}^{t-1} p_i + a_t \leqslant \frac{e}{e-1} \min_j \left\{ \sum_{i=1}^{j-1} p_i + a_j \right\}.$$

Proof of Lemma 3.22. The case with the general rental cost is equivalent to the following unit-rent-cost problem: at time $t = \sum_{i=1}^{j-1} p_i$, the buyer can decide to either pay a_j for buying skis, or continue to rent for p_j consecutive time slots, each with rent cost 1, and then get to see the next possible skis buying cost a_{j+1}. The two problems have the same offline optimal cost.

To solve the case with general rental cost, we use the algorithm in Lemma 3.2 as a subroutine. Assume that in the general-rental-cost case, we have already rented skis for $j-1$ days, the total rental cost we have paid is $t \sum_{i=1}^{j-1} p_i$. Now we see the next buy and rent values a_j and p_j. To decide what to do at the current step, we run the unit-rental-cost algorithm as in the proof of Lemma 3.2 for additional p_j time steps without doing real probing, because we know the buying cost will not change in the next p_j unit time steps. If the algorithm with unit rental cost does not stop in the following p_j unit time steps in the simulation, we decide to do the same, i.e. paying the rental cost p_j at the current time step. If the algorithm with unit rental cost stops and buys skis in the following p_j unit time steps in the simulation, we decide to buy skis immediately, which results in a better total cost than in the corresponding unit-rental-cost case. Since the algorithm in the previous lemma pays an $\frac{e}{e-1}$-approx to the optimal offline cost of the corresponding unit-rental-cost case, our algorithm for the general-rental-case is no worse than it, thus an $\frac{e}{e-1}$-approx to the optimal offline cost of the general-rental-cost case.

□

3.A.2.2 Linear program formulations

To get the linear program relaxation of the optimal Non-Adaptive strategy for selecting one box, we only need to change the objective function of the linear program.

$$\text{minimize} \quad \sum_{i \in \mathcal{B}} x_i p_i \quad + \quad \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{B}, s \in \mathcal{S}} c_{is} z_{is} \qquad \text{(LP-NA-General)}$$

$$
\begin{aligned}
\text{subject to} \quad \sum_{i \in \mathcal{B}} z_{is} &= 1, && \forall s \in \mathcal{S} \\
z_{is} &\leqslant x_i, && \forall i \in \mathcal{B}, s \in \mathcal{S} \\
x_i, z_{is} &\in [0,1] && \forall i \in \mathcal{B}, s \in \mathcal{S}
\end{aligned}
$$

For the LP of optimal SPA strategy for selecting one box, we need to account for the probing time of every box in the constraint. In order to do that, we will require that every box is being probed for p_i consecutive steps: $x_{it} = 1$ means that box i has been probed since time $t - p_i + 1$, and the probing of the box finishes at time t. Thus at each time step t, there are $\sum_{i \in \mathcal{B}} \sum_{t \leqslant t' \leqslant t+p_i-1} x_{it'}$ boxes under probing, and this should be upper bounded by 1. The rest of the program will be the same. Since the probing time of each box is polynomially bounded, such LP still has a polynomial size.

$$
\begin{aligned}
\text{minimize} \quad & \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{B}, s \in \mathcal{S}, t \in \mathcal{T}} t z_{ist} + \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{B}, s \in \mathcal{S}, t \in \mathcal{T}} c_{is} z_{ist} && \text{(LP-SPA-General)} \\
\text{subject to} \quad & \sum_{i \in \mathcal{B}} \sum_{t \leqslant t' \leqslant t+p_i-1} x_{it'} \leqslant 1, && \forall t \in \mathcal{T} \quad (3.17) \\
& \sum_{t \in \mathcal{T}} x_{it} \leqslant 1, && \forall i \in \mathcal{B} \\
& z_{ist} \leqslant x_{it}, && \forall s \in \mathcal{S}, i \in \mathcal{B}, t \in \mathcal{T} \\
& \sum_{t' \in \mathcal{T}, i \in \mathcal{B}} z_{ist'} = 1, && \forall s \in \mathcal{S} \\
& x_{it}, z_{ist} \in [0,1] && \forall s \in \mathcal{S}, i \in \mathcal{B}, t \in \mathcal{T}
\end{aligned}
$$

For the case of selecting k boxes or picking a matroid basis of rank k, the change to the LP of SPA strategy would be the same: replacing the first constraint "$\sum_{i \in \mathcal{B}} x_{it} = 1$" by (3.17).

3.A.2.3 Warmup: SPA vs NA, selecting a single item

We show that Algorithm 4 works for the general-probing-time case with approximation ratio only losing a factor of 2.

Lemma 3.23. *In general-probing-times case, we can efficiently compute a scenario-aware partially-adaptive strategy with competitive ratio 2 against the optimal non-adaptive strategy.*

Proof. The analysis of ALG_c remains the same, i.e. $\mathbf{E}[\text{ALG}_c] \leqslant \text{OPT}_c$. Now we consider ALG_t. Notice that each step of the algorithm for constructing the probing order is completely independent, with stopping probability $\frac{1}{\sum_{i\in\mathcal{B}} x_i}$ at each point. However, the "length" of each step depends on the probing time for the box picked for that step. Let τ denote the step at which we stop. We have $\mathbf{E}[\tau] = \sum_{i\in\mathcal{B}} x_i$. For any step $t < \tau$, the expected probing time for this step is

$$\begin{aligned}\mathbf{E}[\text{probing time at step } t | t \text{ not stopping time}] &< \frac{\mathbf{E}[\text{probing time at step } t]}{\mathbf{Pr}[t \text{ is not stopping time}]} \\ &= \frac{\sum_{i\in\mathcal{B}} x_i p_i / \sum_{i\in\mathcal{B}} x_i}{1 - 1/\sum_{i\in\mathcal{B}} x_i}.\end{aligned}$$

Thus the expected total probing time is

$$\begin{aligned}\mathbf{E}[\mathrm{ALG}_t] &= \mathbf{E}[\text{probing time at steps } < \tau] + \mathbf{E}[\text{probing time at step } \tau] \\ &\leqslant \frac{\sum_{i\in\mathcal{B}} x_i p_i / \sum_{i\in\mathcal{B}} x_i}{1 - 1/\sum_{i\in\mathcal{B}} x_i}(\mathbf{E}[\tau] - 1) + \sum_{i\in\mathcal{B}} x_i p_i \\ &= 2\sum_{i\in\mathcal{B}} x_i p_i = 2\mathrm{OPT}_t.\end{aligned}$$

Thus $\mathbf{E}[\mathrm{ALG}] \leqslant 2\mathrm{OPT}$. □

3.A.2.4 SPA vs PA: k-coverage and matroid base

Now we show the algorithms for the case of selecting k boxes and selecting a matroid base still works when we have general probing times. The only difference is that the algorithms will now base on the modified LP in Section 3.A.2.2.

In the entire analysis of the two cases, the only place where we employ probing times of boxes is when we try to bound the expected total probing time of each phase ℓ in (3.10) and (3.14) respectively. These expected probing time terms, $\sum_{t'\leqslant 2^\ell} x_{it'}$ in (3.10) and $\sum_{t'\leqslant t} x_{it'}$ in (3.14), will get changed to $\sum_{t'\leqslant 2^\ell} p_i x_{it'}$ and $\sum_{t'\leqslant t} p_i x_{it'}$ respectively.

Now we argue that the proof will still go through step by step, and it suffices to show that $\sum_{i\in\mathcal{B}} \sum_{t'\leqslant t} p_i x_{it'} \leqslant t$. Sum up LP constraint (3.17) from 1 to t, we have

$$\sum_{i\in\mathcal{B}} \sum_{t'\leqslant t} \sum_{t'\leqslant t''\leqslant t'+p_i-1} x_{it''} \leqslant t.$$

Notice that for any $t' \leqslant t$, $x_{it'}$ appears exactly p_i times in the sum. The counting argument implies

$$\sum_{i\in\mathcal{B}}\sum_{t'\leqslant t} p_i x_{it'} \leqslant \sum_{i\in\mathcal{B}}\sum_{t'\leqslant t}\sum_{t'\leqslant t''\leqslant t'+p_i-1} x_{it''} \leqslant t.$$

Observe that for the case of $k = 1$ discussed in Section 3.2 this extension implies the 124-approximation of Theorem 3.12. We believe that the argument can be tightened to obtain a much better factor for $k = 1$ but do not attempt to optimize the constant. We also note that our reduction to MSSC in Section 3.2 continues to work with general probing times, however this general setting has not been studied previously for MSSC.

3.A.3 Inapproximability of the profit maximization variant

In this section we consider the profit maximization variant of the problem discussed above. The boxes now contain some prize value v_{is} for each box i in scenario s, and we want to maximize expected *profit*. Formally, let $\mathcal{P}_s$ be the set of probed boxes in scenario s, our objective is to maximize

$$\mathbf{E}_s\left[\max_{i\in\mathcal{P}_s} v_{is} - |\mathcal{P}_s|\right].$$

It turns out that, contrary to the minimization case, obtaining a constant approximation in this setting is impossible, as the following theorem shows.

Theorem 3.24. *Assuming P≠NP, no computationally efficient fully-adaptive algorithm can approximate the optimal non-adaptive profit within a constant factor.*

The proof follows similarly to the minimization case, where we use again Lemma 3.21 to construct a bad instance that can give arbitrarily bad approximation.

Proof of Theorem 3.24. Let $H > 0$ and $p \in [0, 1]$ be appropriate constants, to be determined later. Let $\mathcal{SC} = ([m], \{S_1, \dots, S_n\})$ be a set cover instance with m elements and n sets. Denote its optimal value by OPT_{SC}. To transform this into an instance of the search problem, every element $e_j \in [m]$ corresponds to a scenario j, and every set S_i to a box i. We set $v_{ij} = H$ iff $e_j \in S_i$, otherwise $v_{ij} = 0$. We also add a new scenario X with $v_{Xi} = 0$, $\forall i \in [n]$. Scenario X occurs with probability p and all the other m scenarios happen with probability $(1-p)/m$ each. Observe that contrary to the minimization lower bound of Section 3.A.1.2, the additional scenario X has a low value (0 instead of H), and the other scenarios give a high value (H) when covered.

In this instance, the profit of optimal non-adaptive strategy is $\mathrm{OPT}_{NA} \geqslant (1-p)H - \mathrm{OPT}_{SC}$, since we may pay the set-cover cost to find a box with value H in every scenario other than X.

Now let us consider any computationally efficient algorithm $\mathcal{A}$ that returns a fully adaptive strategy for such an instance. Since the values of the boxes are 0 or H, we may assume without loss of generality that any fully-adaptive strategy stops probing as soon as it observes a box with value H and chooses that box. We say that the adaptive strategy covers a scenario when it finds a box of value H in that scenario. Observe that similarly to the lower bound of subsection 3.A.1.2, any FA strategy will follow a probing order independent of the scenario, which we can then convert to an approximate solution for the underlying set cover instance. Then, for any constant $\varepsilon > 0$, by Lemma 3.21, there must exist a set cover instance and correspondingly an instance of the search problem, such that for the adaptive strategy returned by the algorithm for that instance, for every k, the fraction of scenarios other than X covered before step k is at most $1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^{k-1}$. Consider such an instance and let N denote the maximum number of boxes the strategy probes before stopping to return a box of value H.

The same as (3.16), the expected query time of the strategy is at least

$$pN + (1-p)\left(1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^N\right)\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}.$$

On the other hand, the expected value obtained by the fully-adaptive strategy is at most

$$\begin{aligned} &H\cdot\mathbf{Pr}\left[s \neq X \wedge \text{FA finds value } H \text{ in first } N \text{ steps}\right] \leqslant \\ &(1-p)H\left(1 - (1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}})^N\right). \end{aligned}$$

Thus the profit of such fully-adaptive strategy is upper bounded by

$$\begin{aligned} \mathrm{ALG}_{FA} \leqslant (1-p)H\cdot&\left(1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^N\right) \\ &- (1-p)\left(1 - \left(1 - \frac{1+\varepsilon}{\mathrm{OPT}_{SC}}\right)^N\right)\frac{\mathrm{OPT}_{SC}}{1+\varepsilon} - pN. \end{aligned}$$

Let x be defined so that $(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}})^N = e^{-x}$. Then, $N = -x/\ln(1-\frac{1+\varepsilon}{\mathrm{OPT}_{SC}}) \geqslant x(\frac{\mathrm{OPT}_{SC}}{1+\varepsilon}-1)$. Substituting these expressions in the above equation we get

$$\mathrm{ALG}_{FA}(x) \leqslant (1-e^{-x})(1-p)\left(H - \frac{\mathrm{OPT}_{SC}}{1+\varepsilon}\right) - px\left(\frac{\mathrm{OPT}_{SC}}{1+\varepsilon} - 1\right).$$

Observe that the right hand side is maximized at $x = \ln\left(\frac{(1-p)((1+\varepsilon)H-\mathrm{OPT}_{SC})}{p(\mathrm{OPT}_{SC}-(1+\varepsilon))}\right)$. By setting $(1-p)H = \frac{2\varepsilon+1}{\varepsilon+1}\mathrm{OPT}_{SC}$, $\varepsilon \to 0$ and $p \to 1$, we get

$$\frac{\mathrm{ALG}_{FA}}{\mathrm{OPT}_{NA}} \leqslant 2 - \frac{p}{\varepsilon}\log\left(\frac{2\varepsilon}{p}+1\right) \to 0$$

when $\mathrm{OPT}_{SC} \to \infty$. Thus no efficient fully-adaptive algorithm can approximate the optimal non-adaptive profit within any constant factor. □

3.A.4 Missing Proofs of Chapter 3

Lemma 3.2. *[Ski Rental with time-varying buy costs] Consider any sequence $a_1 \geqslant a_2 \geqslant \ldots$. There exists an online algorithm that chooses a stopping time t so that*

$$t - 1 + a_t \leqslant \frac{e}{e-1}\min_j\{j - 1 + a_j\}.$$

Proof of Lemma 3.2. We prove that Algorithm 5 satisfies the lemma. The algorithm sees an instance $\mathcal{I} = \{a_1, a_2, \ldots\}$ and essentially starts a new Ski Rental problem every time it finds a lower $a_t + t - 1$ value, using the $\frac{e}{e-1}$-competitive randomized Ski Rental algorithm Karlin et al. (1990) as a black box, to choose the new stopping time τ.

Every time the algorithm changes the current cost value C (line 4 of Algorithm 5) we say the sequence $\{a_t\}$ takes a *step*. Suppose that the sequence takes a step at time $1 = t_N < t_{N-1} < \cdots < t_1$. Notice that $N \leqslant a_1$, since the sum of renting cost and buying cost after time $t = a_1$ will be at least a_1, which is the total cost at time 1. Denote by ALG^k the algorithm's cost on instance $a_{t_k}, a_{t_{k+1}}, \cdots$, which is the truncated instance that takes k steps until the end. Define $\mathrm{OPT}^k = \min_t\{a_t + t - 1\}$ be the optimal cost for the same instance that the algorithm takes k steps until the end. We claim that $\mathbf{E}\left[\mathrm{ALG}^k\right] \leqslant e/(e-1)\mathrm{OPT}^k$ for any k, and prove the claim using induction on the number of steps.

Observe that for $k = 1$, since the algorithm only takes a step at the beginning, we have $a_t + t - 1 \geqslant a_1$ for any $t \geqslant 1$. In this case, $\mathrm{OPT}^1 = a_1$ and ALG only considers

Algorithm 5: Ski Rental for time-varying buying prices

```
Data: ski(C): random stopping time according to Ski Rental with buying cost C
Input: Sequence $a_1, a_2, \dots$ of buying costs
1  $C = \infty, \tau = \infty$
2  foreach time $t \geqslant 1$ do
3      if $a_t + t - 1 < C$ then
4          $C = a_t + t - 1$
5          $\tau = t - 1 +$ ski$(C - t + 1)$
6      end
7      if $t = \tau$ then
8          Buy at price $\min_{t' \leqslant t}\{a_{t'}\}$
9      end
10 end
```

a_1 as buying cost. This is exactly a special case of the traditional Ski Rental problem with one buying cost. Therefore we get $\mathrm{ALG}^1 \leqslant e/(e-1)\mathrm{OPT}^1$.
For any $k > 1$, denote by T be the first time the algorithm takes a step, τ_0 the first stopping time set by the algorithm. The expected cost of the algorithm is

$$\begin{aligned}
\mathbf{E}\left[\mathrm{ALG}^k\right] &= \mathbf{E}\left[\mathrm{ALG}^k \mathbb{1}_{\{\tau_0 \leqslant T\}}\right] + \mathbf{E}\left[(T + \mathrm{ALG}^{k-1})\mathbb{1}_{\{S > T\}}\right] \\
&\leqslant \mathbf{E}\left[(a_1 + \tau_0)\mathbb{1}_{\{\tau_0 \leqslant T\}}\right] + T\mathbf{Pr}\left[\tau_0 > T\right] + \mathbf{E}\left[\mathrm{ALG}^{k-1}\right] \\
&\leqslant \mathbf{E}\left[(a_1 + \tau_0)\mathbb{1}_{\{\tau_0 \leqslant T\}}\right] + T\mathbf{Pr}\left[\tau_0 > T\right] + \frac{e}{e-1}\mathrm{OPT}^{k-1} \\
&\leqslant \frac{e}{e-1}T + \frac{e}{e-1}(\mathrm{OPT}^k - T) \\
&= \frac{e}{e-1}\mathrm{OPT}^k.
\end{aligned}$$

Here the second line comes from the algorithm's cost when $\tau_0 \leqslant T$ is exactly $a_1 + \tau_0$, which is paying the buying cost a_1 at time 1 and the renting cost for τ_0 rounds. The third line is by inductive hypothesis. The fourth line is true since for a ski-rental instance with T days and buying cost $a_1 > T$, renting for T days is optimal, while the strategy of using τ_0 as stopping time has cost $\mathbf{E}\left[(a_1 + \tau_0)\mathbb{1}_{\{\tau_0 \leqslant T\}}\right] + T\mathbf{Pr}\left[\tau_0 > T\right]$ should give $\frac{e}{e-1}$-approximation to optimal. The last line comes from the fact that $\mathrm{OPT}^k = \mathrm{OPT}^{k-1} + T$, as the optimal solution always choose to rent in the first T steps.

By induction, $\mathrm{ALG}^N \leqslant \frac{e}{e-1}\mathrm{OPT}^N$. □

Corollary 3.3. *Given any* scenario-aware partially-adaptive *strategy* σ*, we can efficiently construct a stopping time* τ*, such that the cost of the* partially-adaptive *strategy* (σ, τ) *is no more than a factor of* $e/(e-1)$ *times the cost of* σ*.*

Proof of Corollary 3.3. Recall that a scenario-aware strategy consists of a sequence and a scenario dependent stopping rule. Let (σ, τ) be the scenario-aware partially-adaptive strategy. For the case with unit query time, by running the algorithm described in the proof of Lemma 3.2 using sequence σ as input[7], we obtain a stopping rule that does not depend on the scenario and only worsens the approximation by a factor of $e/(e-1)$. Similarly for the case of general probing time, we will use the algorithm for Corollary 3.22. □

Lemma 3.21. *Unless P=NP, for any constant* $\epsilon > 0$*, there is no algorithm that for every instance of Set Cover finds* k *sets that cover at least* $1 - \left(1 - \frac{1+\varepsilon}{OPT}\right)^k$ *of the elements for some integer* $k \in \left[1, \frac{\log n}{1+\varepsilon} OPT\right]$*.*

Proof of Lemma 3.21. Assume that there exists an algorithm $\mathcal{A}$ such that for every instance of Set Cover finds k sets that cover at least $1 - (1 - \frac{1+\varepsilon}{OPT})^k$ of the elements. Given an instance of Set Cover, with a ground set of n elements, we repeatedly run $\mathcal{A}$ on the set of uncovered elements left at each round. Every time, we create a new instance with ground set composing of only the elements left uncovered in the previous round. Using the guarantee for $\mathcal{A}$ in each round i we cover at least $(1 - (1 - \frac{1+\varepsilon}{OPT})^{k_i})$ for some $k_i \in \left[1, \log n \frac{OPT}{1+\varepsilon}\right]$.

Denote by z the number of rounds we need to cover all elements and by k_i the number of the elements we cover at round i. In the end of round z there are

$$n \prod_{i=1}^{z} \left(1 - \frac{1+\varepsilon}{OPT}\right)^{k_i} \tag{3.18}$$

elements left uncovered. When the above quantity equals 1, we are left with 1 element and the following holds

$$\sum_{i=1}^{z} k_i = \frac{\log n}{\log\left(\frac{OPT}{OPT-1-\varepsilon}\right)} < \log n \frac{OPT}{1+\varepsilon}$$

[7]For this reduction, we set the costs $a_t = 1 + \min_{i \leqslant t}(c_{\sigma(i)s})$ so that we have a decreasing sequence as Lemma 3.2 requires.

where the first sum is exactly the cost of covering all the elements but one[8], and for the inequality we used Lemma 3.25 with $x = \mathrm{OPT}$ and $c = 1 + \varepsilon$. This result directly implies a better than $(1 - \varepsilon') \ln n$ approximation for Set Cover, which is impossible unless P = NP Dinur and Steurer (2014).

□

Lemma 3.6. *For any $\beta > 1$,*

$$\int_{t-1}^{t} \lceil \beta t' \rceil dt' \leqslant \beta t.$$

Proof of Lemma 3.6.

$$\begin{aligned}
\int_{t-1}^{t} \lceil \beta t' \rceil dt' &\leqslant (\lceil \beta t \rceil - 1) \left(\frac{\lceil \beta t \rceil - 1}{\beta} - t + 1 \right) + \lceil \beta t \rceil \left(t - \frac{\lceil \beta t \rceil - 1}{\beta} \right) \\
&= t - \frac{\lceil \beta t \rceil - 1}{\beta} - 1 + \lceil \beta t \rceil \\
&< \beta \left(t - \frac{\lceil \beta t \rceil - 1}{\beta} \right) - 1 + \lceil \beta t \rceil \\
&= \beta t.
\end{aligned}$$

where the first line is true since for any $t' \leqslant \frac{\lceil \beta t \rceil - 1}{\beta}$, $\lceil \beta t' \rceil \leqslant \lceil \beta t \rceil - 1$; while for any t' such that $\frac{\lceil \beta t \rceil - 1}{\beta} < t' \leqslant t$, $\lceil \beta t' \rceil = \lceil \beta t \rceil$. On the third line we used that $\beta > 1$ and $t > \frac{\lceil \beta t \rceil - 1}{\beta}$. □

Lemma 3.25.

$$\left(\log \left(\frac{x}{x - c} \right) \right)^{-1} < \frac{x}{c}$$

for any $x > c > 0$.

Proof. First we prove the following inequality

$$\log(x + 1) > \frac{x}{x + 1} \text{ for any } x > 0. \tag{3.19}$$

Let $g(x) = \log(x+1) - \frac{x}{x+1}$. The derivative of $g(x)$ is $g'(x) = \frac{1}{x+1} - \frac{1}{(x+1)^2}$ and $g'(x) > 0$ for $x > 0$. Then g is increasing and $\lim_{x \to 0} g(x) = 0$, therefore $g(x) > 0$ for $x > 0$, and the inequality follows. By setting $x + 1$ to be $x/(x - c)$ in inequality 3.19, the lemma follows.

□

[8] By adding 1 to $\sum_{i=1}^{z} k_i$, the inequalities still hold.

Lemma 3.16. *In phase ℓ, the expected number of steps needed to select a set of full rank is at most* $(4 + 2^{\ell+2}/\alpha)t_s^*$.

Proof of Lemma 3.16. Denote by A_j the span of the first j elements selected. Notice that for all $i \in \mathcal{B} \setminus A_j$, the probability of selecting box i is $\alpha \ln k \frac{\sum_{t' \leqslant t} z_{ist'}}{t}$. Thus, the probability of selecting a box that increases the rank by 1 at step $t \geqslant t_s^*$ is

$$\begin{aligned}
\mathbf{Pr}\left[\text{rank } j \text{ to } j+1\right] &= 1 - \prod_{i \in \mathcal{B} \setminus A_j} \left(1 - \alpha \ln k \frac{\sum_{t' \leqslant t} z_{ist'}}{t}\right) \\
&\geqslant 1 - \prod_{i \in \mathcal{B} \setminus A_j} \left(1 - \alpha \ln k \frac{\sum_{t' \leqslant t} z_{ist'}}{2^\ell t_s^*}\right) \\
&\geqslant 1 - \exp\left(- \sum_{i \in \mathcal{B} \setminus A_j} \alpha \ln k \frac{\sum_{t' \leqslant t} z_{ist'}}{2^\ell t_s^*}\right) \\
&\geqslant 1 - \exp\left(- \frac{\alpha \ln k (k-j)}{2^{\ell+1} t_s^*}\right) \\
&\geqslant \min\left(\frac{1}{2}, \frac{\alpha \ln k (k-j)}{2^{\ell+2} t_s^*}\right),
\end{aligned}$$

here the second line follows from $t \leqslant t_s^*$ in phase ℓ; the third line follows from $\prod_i (1 - a_i) \leqslant e^{-\sum_i a_i}$ for any $a_1, a_2, \dots \in [0, 1]$; the fourth line follows from constraint (3.12) and $y_{st} \geqslant \frac{1}{2}$ by (3.9); the last line follows from $1 - e^{-a} \geqslant \frac{1}{2} \min(1, a)$. Thus the expected total steps until a full rank basis is found is

$$\begin{aligned}
\mathbf{E}\left[\mathcal{X}\right] &= \sum_{j=0}^{k-1} \mathbf{E}\left[\text{steps from rank } j \text{ to } j+1\right] \leqslant \sum_{j=0}^{k-1} \left(\frac{2^{\ell+2} t_s^*}{\alpha (k-j) \ln k} + 2\right) \\
&\leqslant \frac{2^{\ell+2}}{\alpha} t_s^* + 2k \leqslant \frac{2^{\ell+2}}{\alpha} t_s^* + 4 t_s^*.
\end{aligned}$$

Here the last equality is by $t_s^* \geqslant \frac{k}{2}$. □

4 PARTIALLY ADAPTIVE: A SIMPLER APPROACH

In this chapter we revisit Pandora's Box with correlations, and provide **simpler**, **learnable** algorithms with **better approximation guarantees**, that directly **generalize** Weitzman's reservation values. More specifically, our results are the following.

- **Generalizing**: we first show how the original reservation values given by Weitzman (1979) can be generalized to work in correlated distributions, thus allowing us to use a version of their initial greedy algorithm.
- **Better approximation**: we give two different variants of our main algorithm, that each uses different updates on the distribution $\mathcal{D}$ after every step.

  1. *Variant 1: partial updates*. We condition on the algorithm not having stopped yet.
  2. *Variant 2: full updates*. We condition on the exact value v revealed in the box opened.

  Both variants improve the approximation described in Chapter 3 from 9.22 to 4.428 for Variant 1 and to 5.828 for Variant 2. It is worth noting that our result for Variant 1 is *almost tight*, since the best possible approximation factor we can obtain is 4, implied by Feige et al. (2004). We included more details on the lower bound in Section 3.2.2.
- **Simplicity**: our algorithms are greedy and only rely on the generalized version of the reservation value, while the algorithms in previous work rely on solving a linear program, and reducing first to Min-Sum Set Cover then to Ski-Rental, making them not straightforward to implement. A 9.22 approximation was also given in Gergatsouli and Tzamos (2022), which followed the same approach but bypassed the need to reduce to Min-Sum Set Cover by directly rounding the linear program via randomized rounding.

Our analysis is enabled by drawing similarities from Pandora's Box to Min-Sum Set Cover, which corresponds to the special case of when the values inside the boxes are 0 or ∞. For Min-Sum Set Cover a simple greedy algorithm was shown to achieve the optimal 4-approximation (Feige et al., 2002). Surprisingly, Weitzman's algorithm can be seen as a direct generalization of that algorithm. Our analysis follows the histogram method introduced in Feige et al. (2002), for bounding the approximation ratio. However, we significantly generalize it to handle values in the boxes and work with tree-histograms required to handle the case with full-updates.

4.1 Preliminaries

In Pandora's Box ($\mathcal{PB}$) we are given a set of n boxes $\mathcal{B}$, each with a known opening cost $c_b \in \mathbb{R}^+$, and a distribution $\mathcal{D}$ over a vector of unknown values $\boldsymbol{v} = (v_1, \dots, v_n) \in \mathbb{R}^n_+$ inside the boxes. Each box $b \in \mathcal{B}$, once it is opened, reveals the value v_b. The algorithm can open boxes sequentially, by paying the opening cost each time, and observe the value instantiated inside the box. The goal of the algorithm is to choose a box of small value, while spending as little cost as possible "opening" boxes. Formally, denoting by $\mathcal{O} \subseteq \mathcal{B}$ the set of opened boxes, we want to minimize

$$\mathbf{E}_{v\sim\mathcal{D}}\left[\min_{b\in\mathcal{O}} v_b + \sum_{b\in\mathcal{O}} c_b\right].$$

A *strategy* for Pandora's Box is an algorithm that in every step decides which is the next box to open and when to stop. We measure the performance of our algorithm usign the competitive (or approximation) ratio; a strategy $\mathcal{A}$ is α-approximation if $\mathbf{E}[\mathcal{A}] \leqslant \alpha\text{OPT}$, where OPT is the optimal online algorithm[1]

A strategy can pick any open box to select at any time. To model this, we assume without loss of generality that after a box is opened the opening cost becomes 0, allowing us to select the value without opening it again. In its full generality, a strategy can make decisions based on every box opened and value seen so far. We call this the *Fully-Adaptive* (FA) strategy.

4.1.1 Weitzman's Algorithm

When the distributions of values in the boxes are independent, Weitzman (1979) described a greedy algorithm that is also the optimal strategy. In this algorithm, we first calculate an index for every box b, called *reservation value* σ_b, defined as the value

[1]The optimal online has the exact same information as our algorithm $\mathcal{A}$ but has infinite computation time to solve the problem.

that satisfies the following equation

$$\mathbf{E}_{v \sim \mathcal{D}}\left[(\sigma_b - v_b)^+\right] = c_b, \tag{4.1}$$

where $(a-b)^+ = \max(0, a-b)$. Then, the boxes are ordered by increasing σ_b and opened until the minimum value revealed is less than the next box in the order. Observe that this is a Partially-Adaptive strategy.

4.2 Generalizing Weitzman's Algorithm

We begin by showing how Weitzman's algorithm can be extended to correlated distributions. Our algorithm calculates a reservation value σ for every box at each step, and opens the box $b \in \mathcal{B}$ with the minimum σ_b. We stop if the value is less than the reservation value calculated, and proceed in making this box *free*; we can re-open this for no cost, to obtain the value just realized at any later point. The formal statement is shown in Algorithm 6.

We give two different variants based on the type of update we do after every step on the distribution $\mathcal{D}$. In the case of partial updates, we only condition on $V_b > \sigma_b$, which is equivalent to the algorithm not having stopped. On the other hand, for full updates we condition on the exact value that was instantiated in the box opened. Theorem 4.1 gives the approximation guarantees for both versions of this algorithm.

Theorem 4.1. *Algorithm 6 is a* 4.428*-approximation for Variant 1 and* 5.828*-approximation for Variant 2 of* Pandora's Box *against the partially-adaptive optimal.*

Proof. We seperately show the two components of this theorem in Theorems 4.2 and 4.3. □

Observe that for independent distributions this algorithm is exactly the same as Weitzman's (Weitzman, 1979), since the product prior $\mathcal{D}$ remains the same, regardless of the values realized. Therefore, the calculation of the reservation values does not change in every round, and suffices to calculate them only once at the beginning.

Scenarios To proceed with the analysis of Theorem 4.1, we assume that $\mathcal{D}$ is supported on a collection of m vectors, $(\boldsymbol{v}^s)_{s \in \mathcal{S}}$, which we call scenarios, and sometimes abuse notation to say that a scenario is sampled from the distribution $\mathcal{D}$. We assume

Algorithm 6: Weitzman's algorithm, for correlated $\mathcal{D}$.

```
  Input: Boxes with costs $c_i \in \mathbb{R}$, distribution over scenarios $\mathcal{D}$.
1 An unknown vector of values $v \sim \mathcal{D}$ is drawn
2 repeat
3     Calculate $\sigma_b$ for each box $b \in \mathcal{B}$ by solving:
          $$\mathbf{E}_{v\sim\mathcal{D}}\left[(\sigma_b - v_b)^+\right] = c_b.$$
4     Open box $b = \text{argmin}_{b\in\mathcal{B}} \sigma_b$
5     Stop if the value the observed $V_b = v_b \leqslant \sigma_b$
6     $c_b \leftarrow 0$ // Box is always open now or can be reopened
7     Update the prior distribution
          - Variant 1: $\mathcal{D} \leftarrow \mathcal{D}|_{V_b > \sigma_b}$ (partial updates)
          - Variant 2: $\mathcal{D} \leftarrow \mathcal{D}|_{V_b = v_b}$ (full updates)
8 until termination;
```

that all scenarios have equal probability. The general case with unequal probabilities follows by creating more copies of the higher probability scenarios until the distribution is uniform.

A scenario is *covered* when the algorithm decides to stop and choose a value from the opened boxes. For a specific scenario $s \in \mathcal{S}$ we denote by $c(s)$ the total opening cost paid by an algorithm before this scenario is covered and by $v(s)$ the value chosen for this scenario.

Reservation Values To analyze Theorem 4.1, we introduce a new way of defining the reservation values of the boxes that is equivalent to (4.1). For a box b, we have that

$$\sigma_b = \min_{A\subseteq\mathcal{S}} \frac{c_b + \sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]\, v_b^s}{\sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]}$$

The equivalence to (4.1), follows since σ_b is defined as the root of the expression

$$\begin{aligned}\mathbf{E}_{s\sim\mathcal{D}}\left[(\sigma_b - v_b^s)^+\right] - c_b &= \sum_{s\in\mathcal{S}} \mathbf{Pr}_{\mathcal{D}}[s]\,(\sigma_b - v_b^s)^+ - c_b \\ &= \max_{A\subseteq\mathcal{S}} \sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]\,(\sigma_b - v_b^s) - c_b.\end{aligned}$$

If we divide the above expression by any positive number, the result will not be affected since we require the root of the equation; σ_b being the root is equivalent to σ_b being the root of the numerator. Thus, dividing by $\sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]$ we get that σ_b is also the root of

$$\max_{A\subseteq \mathcal{S}} \frac{\sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]\,(\sigma_b - v_b^s) - c_b}{\sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]} = \sigma_b - \min_{A\subseteq \mathcal{S}} \frac{c_b + \sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]\, v_b^s}{\sum_{s\in A} \mathbf{Pr}_{\mathcal{D}}[s]}. \tag{4.2}$$

This, gives our formula for computing σ_b, which we can further simplify using our assumption that all scenarios have equal probability. In this case, $\mathbf{Pr}_{\mathcal{D}}[s] = 1/|\mathcal{S}|$ which implies that

$$\sigma_b = \min_{A\subseteq \mathcal{S}} \frac{c_b|\mathcal{S}| + \sum_{s\in A} v_b^s}{|A|}. \tag{4.3}$$

4.3 Conditioning on $V_b > \sigma_b$

We start by describing the simpler variant of our algorithm where after opening each box we update the distribution by conditioning on the event $V_b > \sigma_b$. This algorithm is *partially adaptive*, since the order for each scenario does not depend on the actual value that is realized every time. At every step the algorithm will either stop or continue opening boxes conditioned on the event “We have not stopped yet" which does not differentiate among the surviving scenarios.

Theorem 4.2. *Algorithm 6 is a* 4.428*-approximation for* Pandora*'s Box against the partially-adaptive optimal, when conditioning on* $V_b > \sigma_b$.

In this section we show a simpler proof for Theorem 4.2 that gives a $3+2\sqrt{2} \approx 5.828$-approximation. The full proof for the 4.428-approximation is given in section 4.A.1 of the Appendix. Using the equivalent definition of the reservation value (Equation (4.3)) we can rewrite Algorithm 6 as follows.

Algorithm 7: Weitzman's rule for Partial Updates

```
   Input: Boxes with costs $c_i \in \mathbb{R}$, set of scenarios $\mathcal{S}$.
1  $t \leftarrow 0$
2  $R_0 \leftarrow \mathcal{S}$ the set of scenarios still uncovered
3  while $R_t \neq \emptyset$ do
4      Let $\sigma_t \leftarrow \min_{b \in \mathcal{B}, A \subseteq R_t} \frac{c_b |R_t| + \sum_{s \in A} v_b^s}{|A|}$
5      Let $b_t$ and $A_t$ be the box and the set of scenarios that achieve the minimum
6      Open box $b_t$ and pay $c_{b_t}$
7      Stop and choose the value $v_{b_t}$ at box $b_t$ if it is less than $\sigma_t$ (see also
        Fact 4.2.1)
8      Set $c_{b_t} \leftarrow 0$
9      $R_t \leftarrow R_t \setminus A_t$
10     $t \leftarrow t + 1$
11 end
```

Structure of the solution. An important property to note is that by the equivalent definition of the reservation value (4.3) the set of scenarios that stop at each step are the ones that give a value at most σ for the box opened, as we formally state in the following fact.

Fact 4.2.1. *The value at box* b_t *is less than* σ_t *if and only if* $s \in A_t$.

In equation (8) the set A_t that maximizes the expression contains all the scenarios with value at most σ_b for the box b. Therefore, the set A_t are exactly the scenarios covered at each step t of the algorithm, and can be removed from consideration.

Before showing our result, observe that this algorithm is partially adaptive; the order of the boxes does not depend on the scenario realized. This holds since we only condition on "not having stopped" (i.e. $\mathcal{D}_{V_b > \sigma_b}$) and therefore each scenario either stops or uses the same updated prior as all other surviving scenarios to calculate the next reservation values. If we were to draw our solution, it would look like a line, (see also Figure 4.2 in Appendix 4.A.1), which as we observe in Section 4.4 differs from Variant 2.

Moving on to show the proof, we first start by giving a bound on the cost of the algorithm. The cost can be broken down into opening cost plus the value obtained. Since at any time t, all remaining scenarios R_t pay the opening cost c_{b_t}, we have that the total opening cost is $\sum_t c_{b_t}|R_t|$. Moreover, the chosen value is given as $\sum_t \sum_{s\in A_t} v^s_{b_t}$. Overall, we have that

$$\text{ALG} = \sum_t \left(c_{b_t}|R_t| + \sum_{s\in A_t} v^s_{b_t} \right) = \sum_t |A_t| \frac{c_{b_t}|R_t| + \sum_{s\in A_t} v^s_{b_t}}{|A_t|} = \sum_t |A_t|\sigma_t.$$

Defining σ_s to be the reservation value of scenario s at the time it is covered, i.e. when $s \in A_t$, we get $\text{ALG} = \sum_{s\in\mathcal{S}} \sigma_s$[2]. We follow a *histogram analysis* similar to the proof of Theorem 4 in Feige et al. (2004) for Min-Sum Set Cover and construct the following histograms.

- The OPT_o histogram: put the scenarios on the x-axis on increasing opening cost order c_s^{OPT} according to OPT, the height of each scenario is the opening cost it paid.
- The OPT_v histogram: put the scenarios on the x-axis on increasing covering value order v_s^{OPT} according to OPT, the height of each scenario is the value with which it was covered.
- The ALG histogram: put scenarios on the x-axis in the order the algorithm covers them. The height of each scenario is σ_s. Observe that the area of the ALG histogram is exactly the cost of the algorithm.

Proof of Theorem 4.2. Initially, observe that the algorithm will eventually stop; every time we open a box we cover at least one scenario (since line 3 is cannot be ∞ while scenarios are left uncovered).

To show the approximation factor, we scale the histograms as follows; OPT_o scale horizontally by $1/\alpha_o$ and vertically by $1/(\beta \cdot \gamma)$, and OPT_v scale by $1/\alpha_v$ horizontally, for some constants $\alpha_o, \alpha_v, \gamma, \beta \in (0, 1)$ to be determined later[3]. We align the ALG histogram with OPT_v and OPT_o so that all of them have the same right-hand side. Observe that the optimal opening cost is the area below the histogram OPT_o and has increased by $\beta \cdot \gamma \cdot \alpha_o$, and similarly the area below OPT_v has increased by α_v as a result of the scaling.

[2]Throughout this proof we omit the normalization term $1/|\mathcal{S}|$ both on the algorithms cost and on the optimal cost, without loss of generality, since our guarantee is multiplicative.

[3]Scaling horizontally means that we duplicate every scenario and scaling vertically we just multiply the height at every point by the scale factor.

To conclude the proof it suffices to show that any point in the ALG histogram is inside the sum of the rescaled OPT_v and OPT_o histograms. Consider any point p in the ALG histogram, and let s be its corresponding scenario and t be the time this scenario is covered. We have that the height of the ALG histogram is

$$\sigma_s = \frac{c_{b_t}|R_t| + \sum_{s\in A_t} v^s_{b_t}}{|A_t|} \leqslant \frac{c_b|R_t| + \sum_{s\in A} v^s_b}{|A|} \tag{4.4}$$

where the last inequality holds for all $A \subseteq R_t$ and any $b \in \mathcal{B}$.

Denote by c^* the opening cost such that $\gamma|R_t|$ of the scenarios in R_t have opening cost less than c^*, and by $R_{\text{low}} = \{s \in R_t : c_s^{\text{OPT}} \leqslant c^*\}$ the set of these scenarios. Similarly denote by v^* the value of scenarios in R_{low} such that $\beta|R_{\text{low}}|$ of the scenarios have value less than v^* and by $L = \{s \in R_{\text{low}} : v_s^{\text{OPT}} \leqslant v^*\}$ these scenarios. This split is shown in Figure 4.1, and the constants $\beta, \gamma \in (0, 1)$ will be determined at the end of the proof.

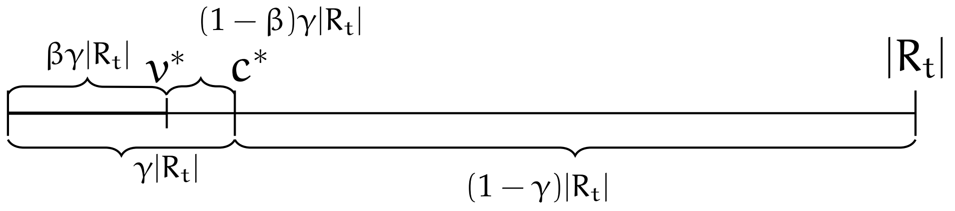

Figure 4.1: Split of scenarios in R_t.

Let B_L be the set of boxes that the optimal solution uses to cover the scenarios in L. Let $L_b \subseteq L \subseteq R_t$ be the subset of scenarios in L that choose the value at box b in OPT. Using inequality (4.4) with $b \in B_L$ and $A = L_b$, we obtain $\sigma_s|L_b| \leqslant c_b|R_t| + \sum_{s\in L_b} v_s^{\text{OPT}}$, and by summing up the inequalities for all $b \in B_L$ we get

$$\sigma_s \leqslant \frac{|R_t|\sum_{b\in B_L} c_b + \sum_{s\in L} v_s^{\text{OPT}}}{|L|} \leqslant \frac{|R_t|c^* + \sum_{s\in L} v_s^{\text{OPT}}}{|L|} \leqslant \frac{c^*}{\beta\cdot\gamma} + \frac{\sum_{s\in L} v_s^{\text{OPT}}}{|L|} \tag{4.5}$$

where for the second inequality we used that the cost for covering the scenarios in L is at most c^* by construction, and in the last inequality that $|L| = |R_t|/(\beta\cdot\gamma)$. We consider each term above separately, to show that the point p is within the histograms.

Bounding the opening cost. By the construction of c^*, the point in the OPT_o histogram that has cost at least c^* is at distance at least $(1-\gamma)|R_t|$ from the right hand side.

This means that in the rescaled histogram, the point that has cost at least $c^*/(\beta \cdot \gamma)$ is at distance at least $(1-\gamma)|R_t|/\alpha_o$ from the right hand side.

On the other hand, in the ALG histogram the distance of p from the right edge of the histogram is at most $|R_t|$, therefore for the point p to be inside the OPT_o histogram we require

$$\alpha_o \leqslant 1-\gamma. \tag{4.6}$$

Observe that throughout the proof we did not use the fact that we change the opening cost to 0, therefore the bound on our cost works even if we re-pay the boxes that are reopened.

The fact that the opening cost becomes 0 is not directly used in the analysis (i.e. inequalities (4.4) and (4.5)). Our analysis gives an upper bound on the cost of the algorithm, even if the algorithm never changes the cost of an opened box to 0. That is the reason in (4.4) and (4.5) the cost appears unchanged but the analysis still works for the algorithm since we just want an upper bound (and if we changed the cost to 0 this would only lower the cost of the algorithm).

Bounding the values cost. By the construction of v^*, the point in the OPT_v histogram that has value v^* is at distance at least $|R_t|(1-\beta)\gamma$ from the right hand side. This means that in the rescaled histogram, the point that has value at least v^* is at distance at least $(1-\beta)\gamma|R_t|/\alpha_v$ from the right hand side.

On the other hand, in the ALG histogram the distance of p from the right edge of the histogram is at most $|R_t|$, therefore for the point p to be inside the OPT_o histogram we require

$$\alpha_v \leqslant (1-\beta)\gamma. \tag{4.7}$$

We optimize the constants $\alpha_o, \alpha_v, \beta, \gamma$ by ensuring that inequalities (4.6) and (4.7) hold. We set $\alpha_o = 1-\gamma$ and $\alpha_v = (1-\beta)\gamma$, and obtain that $\text{ALG} \leqslant \text{OPT}_o/(\beta \cdot \gamma \cdot (1-\gamma)) + \text{OPT}_v/((1-\beta)\gamma)$. Requiring these to be equal we get $\beta = 1/(2-\gamma)$, which is minimized for $\beta = 1/\sqrt{2}$ and $\gamma = 2-\sqrt{2}$ for a value of $3+2\sqrt{2}$.

□

4.4 Conditioning on $V_b = v$

In this section we switch gears to our second variant of Algorithm 6, where in each step we update the prior $\mathcal{D}$ conditioning on the event $V_b = v$. We state our result in Theorem 4.3. In this case, the conditioning on $\mathcal{D}$ implies that the algorithm at every step removes the scenarios that are *inconsistent* with the value realized. In order to understand better the differences of the two variants and their conditioning we included an example and a discussion in section 4.A of the Appendix.

Theorem 4.3. *Algorithm 6 is a* $3 + 2\sqrt{2} \approx 5.828$*-approximation for* PANDORA*'s* BOX *against the partially-adaptive optimal, when conditioning on* $V_b = v$*.*

The main challenge was that the algorithm's solution is now a tree with respect to scenarios instead of a line as in the case of $\mathcal{D}|_{V_b > \sigma_b}$. Specifically, in the $D|_{V_b > \sigma_b}$ variant at every step all scenarios that had $V_b \leqslant \sigma_b$ were covered and removed from consideration. However in the $D|_{V_b = v}$ variant the remaining scenarios are split into different cases, based on the realization of V, as shown in the example of Figure 4.4, which is deferred to Section 4.A.2 of the Appendix due to space constraints.

This results into the ALG histogram not being well defined, since there is no unique order of covering the scenarios. We overcome this by generalizing the histogram approach to trees.

Proof of Theorem 4.3. The proof follows similar steps to that of Theorem 4.2, thus we only highlight the differences. The algorithm is presented below, the only change is line 5 where we remove the inconsistent with the value revealed scenarios, which also leads to our solution branching out for different scenarios and forming a tree.

Bounding the opening cost Consider the tree $\mathcal{T}$ of ALG where at every node u a set A_u of scenarios is covered. We associate this tree with node weights, where at every node u, we assign $|A_u|$ weights $(\sigma_u, ..., \sigma_u)$. Denote, the weighted tree by $\mathcal{T}_{\text{ALG}}$. As before, the total cost of ALG is equal to the sum of the weights of the tree.

We now consider two alternative ways of assigning weights to the the nodes, forming trees $\mathcal{T}_{\text{OPT}_o}$, $\mathcal{T}_{\text{OPT}_v}$ using the following process.

- $\mathcal{T}_{\text{OPT}_o}$. At every node u we create a vector of weights $\boldsymbol{w}_u^{\text{OPT}_o} = (c_s^{\text{OPT}})_{s \in A_u}$ where each c_s^{OPT} is the opening cost that scenario $s \in A_u$ has in the optimal solution.

Algorithm 8: Weitzman's rule for Full Updates

```
Input: Boxes with costs $c_i \in \mathbb{R}$, set of scenarios $\mathcal{S}$.
1  Define a root node $u$ corresponding to the set $\mathcal{S}$
2  $R_u \leftarrow \mathcal{S}$ the set of scenarios still uncovered
3  while $R_u \neq \emptyset$ do
4      Let $\sigma_u \leftarrow \min_{b \in \mathcal{B}, A \subseteq R_u} \frac{c_b |R_u| + \sum_{s \in A} v_b^s}{|A|}$
5      Let $b_u$ and $A_u$ be the box and the set of scenarios that achieve the
        minimum
6      Open box $b_u$ paying $c_{b_u}$ and observe value $v$
7      Stop and choose the value at box $b_u$ if it is less than $\sigma_u$: this holds iff
        $s \in A_u$
8      Set $c_{b_u} \leftarrow 0$
9      Let $u'$ be a vertex corresponding to the set of consistent scenarios with
           $R_{u'} \triangleq R_u \setminus \left(A_u \cup \{s \in R_u : v_{b_u}^s \neq v\}\right)$ // Remove inconsistent
           scenarios
10     Set $u \leftarrow u'$
11 end
```

- $\mathcal{T}_{\text{OPT}_v}$. At every node u we create a vector of weights $\boldsymbol{w}_u^{\text{OPT}_v} = (v_s^{\text{OPT}})_{s \in A_u}$ where each v_s^{OPT} is the value the optimal uses to cover scenario $s \in A_u$.

We denote by $\text{cost}(\mathcal{T}_{\text{ALG}})$ the sum of all weights in every node of the tree $\mathcal{T}$. We have that $\text{cost}(\mathcal{T})$ is equal to the total cost of ALG, while $\text{cost}(\mathcal{T}_{\text{OPT}_o})$ and $\text{cost}(\mathcal{T}_{\text{OPT}_v})$ is equal to the optimal opening cost OPT_o and optimal value OPT_v respectively. Intuitively, the weighted trees correspond to the histograms in the previous analysis of Theorem 4.2.

We want to relate the cost of ALG, to that of $\mathcal{T}_{\text{OPT}_o}$ and $\mathcal{T}_{\text{OPT}_v}$. To do this, we define an operation similar to histogram scaling, which replaces the weights of every node u in a tree with the top ρ-percentile of the weights in the subtree rooted at u. As the following lemma shows, this changes the cost of a tree by a bounded multiplicative factor.

Lemma 4.4. *Let $\mathcal{T}$ be a tree with a vector of weights $\boldsymbol{w}_u$ at each node $u \in \mathcal{T}$, and let $\mathcal{T}^{(\rho)}$ be the tree we get when we substitute the weights of every node with the top ρ-percentile of all the weights in the subtree of $\mathcal{T}$ rooted at u. Then*

$$\rho \cdot \textit{cost}(\mathcal{T}^{(\rho)}) \leqslant \texttt{cost}(\mathcal{T}).$$

We defer the proof of Lemma 4.4 to Section 4.A.2 of the Appendix. To complete the proof of Theorem 4.3, and bound $\text{cost}(\mathcal{T}_{\text{ALG}})$, we show as before that the weights at every node u, are bounded by the weights of $\mathcal{T}^{(1-\gamma)}_{\text{OPT}_o}$ scaled by $\frac{1}{\beta\gamma}$ plus the weights of $\mathcal{T}^{((1-\beta)\gamma)}_{\text{OPT}_v}$, for the constants $\beta, \gamma \in (0,1)$ chosen in the proof of Theorem 4.2. This implies that

$$\begin{aligned}\text{cost}(\mathcal{T}_{\text{OPT}_o}) &\leqslant \frac{1}{\beta\gamma}\text{cost}(\mathcal{T}^{(1-\gamma)}_{\text{OPT}_o}) + \text{cost}(\mathcal{T}^{((1-\beta)\gamma)}_{\text{OPT}_v}) \\ &\leqslant \frac{1}{\beta\gamma(1\text{-}\gamma)}\text{cost}(\mathcal{T}_{\text{OPT}_o}) + \frac{1}{(1\text{-}\beta)\gamma}\text{cost}(\mathcal{T}_{\text{OPT}_v})\end{aligned}$$

which gives ALG $\leqslant 5.828\,$OPT for the choice of β and γ. The details of the proof are similar to the one of Theorem 4.1, and are deferred to section 4.A.2 of the Appendix. □

Note on the approximation factors. Observe that Variant 2, where we condition on $V_b = v$ has a worse approximation factor than Variant 1 where we only condition on $V_b > \sigma_b$. Intuitively someone might expect that with more information the approximation factor will improve. However, it is challenging to argue about this formally. It is also plausible that such monotonicity may not hold as more information might lead the greedy algorithm to make wrong decisions. Instead of making any such claims, we analyze this case directly by showing that our proof approach extends to the full update variant with a generalization of the histogram method to work on trees. Our technique for improving the approximation for the partial updates variant could not be generalized however and thus we only obtain the worse approximation guarantee.

4.A Appendix for Chapter 4

Updating the prior. We include an example showing the process of updating the prior for our two different updating rules. The (correlated) distribution is a set of vectors of size n, where each is drawn with some probability. When we open a box and see a value, some scenarios are not “possible” anymore, i.e. we know they cannot be the ones realized. We illustrate in the following example. Assume there are 3 of these vectors (scenarios).

	b_1	b_2	b_3
S_1	3	4	7
S_2	6	4	2
S_3	7	7	2

Table 4.1: Example with 3 scenarios and 3 boxes.

The rows in the matrix above are the scenarios, and the columns are the boxes. For example, if scenario S_2 is the one realized (i.e. drawn from the distribution) then the values inside boxes b_1, b_2 and b_3 are 6, 4 and 2 respectively. The distribution $\mathcal{D}$ is essentially drawing one of the scenarios with some probability.

To see what the conditioning means: assume we open box b_1 and we see the value 6 (and assume for the sake of the example that the reservation value of box 1 is $\sigma_1 = 5$).

- **Variant 1**: we condition on $6 = V_b > \sigma_1 = 5$ meaning that scenario S_1 is not possible anymore (because if S_1 was the one drawn from $\mathcal{D}$, then we would have seen a value less than $\sigma_1 = 5$ when opening the box), and is removed from the set S the algorithm considers (line 9, Alg 7)
- **Variant 2**: we condition on $V_b = 6$, which means that scenarios S_1 and S_3 are both removed (similarly, because if any of these were drawn, we would not have seen 6 upon opening the box)

Differences in the variants. As a result of the different conditioning, the solution for the $V_b > \sigma$ variant is *partially adaptive* meaning that the next box the algorithm opens, only depends on the scenarios that remain. However, for the $V_b = v$ variant the solution is *fully adaptive* (meaning that the next box opened, depends on the exact value seen). This is illustrated in Figures 4.2 and 4.4 in the Appendix, where Variant 1's solution can be represented by a line graph (Figure 4.2), while Variant 2's solution is a tree (Figure 4.4).

4.A.1 Proofs from Section 4.3

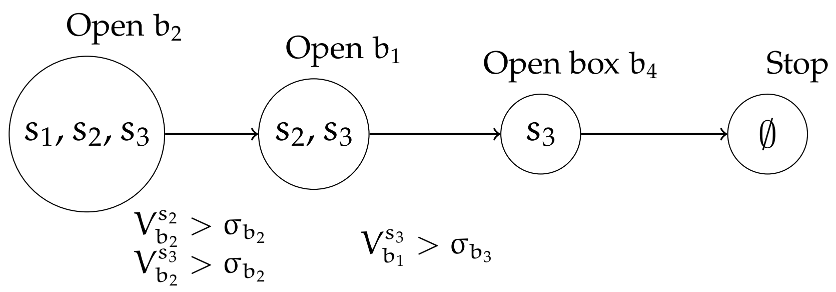
Open b_2
Open b_1
Open box b_4
Stop
s_1, s_2, s_3
s_2, s_3
s_3
$\emptyset$
$V_{b_2}^{s_2} > \sigma_{b_2}$
$V_{b_2}^{s_3} > \sigma_{b_2}$
$V_{b_1}^{s_3} > \sigma_{b_3}$

Figure 4.2: Algorithm's solution when $\mathcal{D} \leftarrow \mathcal{D}_{V>\sigma}$, for an instance with scenarios $\mathcal{S} = \{s_1, s_2, s_3\}$, and boxes $\mathcal{B} = \{b_1, b_2, b_3, b_4\}$. The circles contain the scenarios that have not stopped at each step. Scenario s_1 stopped at box b_2, scenario s_2 stopped at box b_1 and s_3 at box b_4.

Theorem 4.2. *Algorithm 6 is a* 4.428*-approximation for* Pandora*'s Box against the partially-adaptive optimal, when conditioning on* $V_b > \sigma_b$.

The tighter guarantee proof follows the steps of the proof in section 4.3 for the opening cost, but provides a tighter analysis for the values cost.

Tight proof of Theorem 4.2. Denote by σ_s the reservation value for scenario s when it was covered by ALG and by $\mathcal{T}$ the set of boxes opened i.e. the steps taken by the algorithm. Then we can write the cost paid by the algorithm as follows

$$\text{ALG} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \sigma_s = \frac{1}{|\mathcal{S}|} \sum_{p \in \mathcal{T}} |A_t| \sigma_p. \tag{4.8}$$

We use the same notation as section 4.3 which we repeat here for convenience. Consider any point p in the ALG histogram, and let s be its corresponding scenario and t be the time this scenario is covered.

- R_t : set of uncovered scenarios at step t
- A_t : set of scenarios that ALG chooses to cover at step t
- c^*: the opening cost such that $\gamma|R_t|$ of the scenarios in R_t have opening cost less than c^*
- $R_{\text{low}} = \{s \in R_t : c_s^{\text{OPT}} \leqslant c^*\}$ the set of these scenarios

- v^*: the value of scenarios in R_{low} such that $b|R_{\text{low}}|$ of the scenarios have value less than v^*
- $L = \{s \in R_{\text{low}} : v_s^{\text{OPT}} \leqslant v^*\}$ the set of scenarios with value at most v^*
- B_L: set of boxes the optimal uses to cover the scenarios in L of step t

The split described in the definitions above is again shown in Figure 4.3, and the constants $1 > \beta, \gamma > 0$ will be determined in the end of the proof.

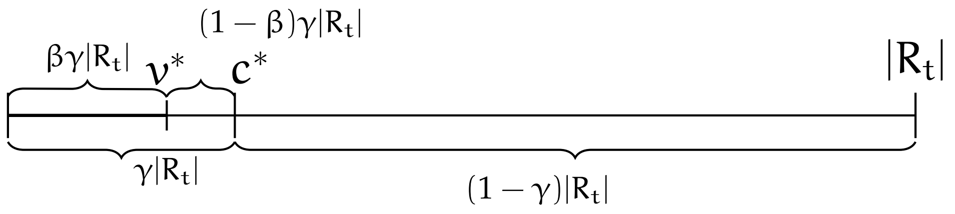

Figure 4.3: Split of scenarios in R_t.

Continuing from equation (4.8) we obtain the following.

$$\begin{aligned}
\text{ALG} &\leqslant \frac{1}{|\mathcal{S}|}\sum_{t\in\mathcal{T}}|A_t|\frac{|R_t|\sum_{b\in B_L}c_b + \sum_{s\in L}v_s^{\text{OPT}}}{|L|} && \text{Inequality 4.5}\\
&\leqslant \frac{1}{|\mathcal{S}|}\sum_{t\in\mathcal{T}}\left(|A_t|\frac{c^*}{\beta\gamma} + \frac{\sum_{s\in L}v_s^{\text{OPT}}}{|L|}\right) && \text{Ineq. 4.5 and } |L| = \gamma\beta|R_t|\\
&\leqslant \frac{\text{OPT}_o}{\beta\gamma(1-\gamma)}\sum_{t\in\mathcal{T}}\frac{|A_t|}{|\mathcal{S}|} + \sum_{t\in\mathcal{T}}\frac{|A_t|}{|\mathcal{S}|}\frac{\sum_{s\in L}v_s^{\text{OPT}}}{|L|} && \text{Since } c^* \leqslant \text{OPT}_o/(1-\gamma)\\
&= \frac{\text{OPT}_o}{\beta\gamma(1-\gamma)} + \sum_{p\in\mathcal{T}}\frac{|A_t|}{|\mathcal{S}|}\frac{\sum_{s\in L}v_s^{\text{OPT}}}{|L|} && \text{Since } \sum_t |A_t| = |\mathcal{S}|
\end{aligned}$$

Where in the second to last inequality we used the same histogram argument from section 4.3, to bound c^* by $\text{OPT}_o/(1-\gamma)$.

To bound the values term, observe that if we sorted the optimal values v_s^{OPT} that cover each scenario by decreasing order, and denote j_s the index of v_s^{OPT} in this ordering, we add v_s^{OPT} multiplied by the length of the interval every time $j_s \in \left[(1-\beta)\gamma|R_t|, \gamma|R_t|\right]$. This implies that the length of the intervals we sum up for v_s^{OPT}

ranges from j_s/γ to $j_s/((1-\beta)\gamma)$, therefore the factor for each v_s^{OPT} is

$$\frac{1}{\gamma}\sum_{i=j_s/\gamma}^{j_s/(1-\beta)\gamma}\frac{1}{i} \leqslant \frac{1}{\gamma}\log\left(\frac{1}{1-\beta}\right)$$

We want to balance the terms $1/(\beta\gamma(1-\gamma))$ and $1/\gamma\log(1/(1-\beta))$ which gives that

$$\gamma = 1 - \frac{1}{\beta\log\left(\frac{1}{1-\beta}\right)}.$$

Since we balanced the opening cost and value terms, by substituting the expression for γ we get that the approximation factor is

$$\frac{1}{\beta\gamma(1-\gamma)} = \frac{\beta\log^2\left(\frac{1}{1-\beta}\right)}{\beta\log\left(\frac{1}{1-\beta}\right)-1}.$$

Numerically minimizing that ratio for β and ensuring that $0<\beta,\gamma<1$ we get that the minimum is 4.428 obtained at $\beta\approx 0.91$ and $\gamma\approx 0.55$. □

4.A.2 Proofs from Section 4.4

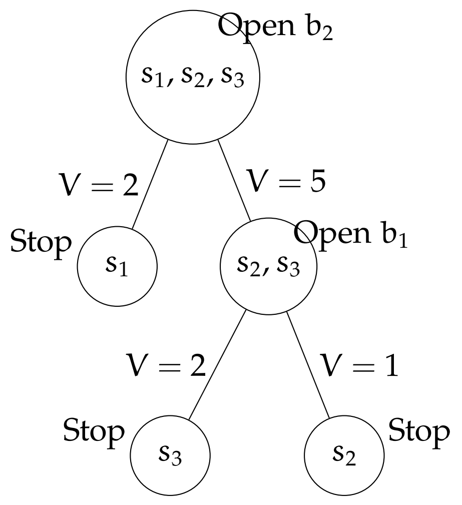

Figure 4.4: Algorithm's solution when conditioning on $V = v$, for an instance with scenarios $\mathcal{S} = \{s_1, s_2, s_3\}$, and boxes $\mathcal{B} = \{b_1, b_2\}$. The nodes contain the consistent scenarios at each step, and the values V are revealed once we open the corresponding box.

Theorem 4.3. *Algorithm 6 is a $3+2\sqrt{2} \approx 5.828$-approximation for* P*ANDORA's* B*ox against the partially-adaptive optimal, when conditioning on $V_b = v$.*

Continued proof of Theorem 4.3. We now proceed to give the bound on the weights of the nodes of $\mathcal{T}_{ALG}$. Consider any node u. We have that the weights at this node are equal to

$$\sigma_u = \frac{c_{b_u}|R_u| + \sum_{s\in A_t} v^s_{b_u}}{|A_t|} \leqslant \frac{c_b|R_u| + \sum_{s\in A} v^s_b}{|A|}$$

where the last inequality holds for all $A \subseteq R_u$ and any $b \in \mathcal{B}$.

Let c^*_u be the opening cost such that $\gamma|R_u|$ of the scenarios in R_u have opening cost less than c^*_u, and by $R_{low} = \{s \in R_u : c^{OPT}_s \leqslant c^*_u\}$ the set of these scenarios. Similarly denote by v^*_u the value of scenarios in R_{low} such that $\beta|R_{low}|$ of the scenarios have value less than v^*_u and by $L = \{s \in R^p_{low} : v^{OPT}_s \leqslant v^*_u\}$ these scenarios. This split is shown in Figure 4.1.

Note that, c^*_u corresponds to the weights of node u in $\mathcal{T}^{(1-\gamma)}_{OPT_o}$, while the weights of node u at $\mathcal{T}^{(1-\gamma)}_{OPT_v}$ are at least v^*_u.

Let B_L be the set of boxes that the optimal solution uses to cover the scenarios in L. Let $L_b \subseteq L \subseteq R_u$ be the subset of scenarios in L that choose the value at box b in OPT. Using inequality (4.4) with $b \in B_L$ and $A = L_b$, we obtain $\sigma_u|L_b| \leqslant c_b|R_u| + \sum_{s\in L_b} v^{OPT}_s$, and by summing up the inequalities for all $b \in B_L$ we get

$$\sigma_u \leqslant \frac{|R_u|\sum_{b\in B_L} c_b + \sum_{s\in L} v^{OPT}_s}{|L|} \leqslant \frac{|R_u|c^* + \sum_{s\in L} v^{OPT}_s}{|L|} \leqslant \frac{c^*_u}{\beta\cdot\gamma} + v^*_u \tag{4.9}$$

where for the second inequality we used that the cost for covering the scenarios in L is at most c^*_u by construction, and in the last inequality that $|L| = |R_t|/(\beta\cdot\gamma)$. We consider each term above separately, to show that the point p is within the histograms. □

Lemma 4.4. *Let $\mathcal{T}$ be a tree with a vector of weights $\boldsymbol{w}_u$ at each node $u \in \mathcal{T}$, and let $\mathcal{T}^{(\rho)}$ be the tree we get when we substitute the weights of every node with the top ρ-percentile of all the weights in the subtree of $\mathcal{T}$ rooted at u. Then*

$$\rho \cdot \mathit{cost}(\mathcal{T}^{(\rho)}) \leqslant \mathrm{cost}(\mathcal{T}).$$

Proof of Lemma 4.4. We denote by $\mathcal{T}_u$ the subtree rooted at u, by $W(\mathcal{T}) = \{w : w \in \boldsymbol{w}_v \text{ for } v \in \mathcal{T}\}$ the (multi)set of weights in the tree $\mathcal{T}$. Denote, by $q^\rho(\mathcal{T})$ be the top ρ percentile of all the weights in $\mathcal{T}$. Finally, we define $Q(\rho|\mathcal{T})$ for any tree $\mathcal{T}$ as follows:

- We create a histogram $\mathrm{H}(x)$ of the weights in $W(\mathcal{T})$ in increasing order.
- We calculate the area enclosed within $(1-\rho)|W(\mathcal{T})|$ until $|W(\mathcal{T})|$:

  $$Q\left(\rho|\mathcal{T}\right) = \int_{(1-\rho)|W(\mathcal{T})|}^{|W(\mathcal{T})|} \mathrm{H}(x)dx$$

  This is approximately equal to the sum of all the values greater than $q^{\rho}(\mathcal{T})$ with values exactly $q^{\rho}(\mathcal{T})$ taken fractionally so that exactly ρ fraction of values are selected.

We show by induction that for every node u, it holds that $\rho \cdot \mathrm{cost}(\mathcal{T}_u^{(\rho)}) \leqslant Q\left(\rho|\mathcal{T}\right)$

- For the base case, for all leaves u, the subtree $\mathcal{T}_u$ only has one node and the lemma holds as $\rho q^{\rho}(\mathcal{T}_u) \leqslant Q\left(\rho|\mathcal{T}_u\right)$.
- Now, let r be any node of the tree, and denote by $\mathrm{child}(r)$ the set of the children nodes of r.

$$\begin{aligned}
\rho \cdot \mathrm{cost}(\mathcal{T}_r^{(\rho)}) &= \rho \cdot q^{\rho}(\mathcal{T}_r)|\boldsymbol{w}_r| + \rho \cdot \sum_{v \in \mathrm{child}(r)} \mathrm{cost}(\mathcal{T}_v^{(\rho)}) && \text{Def of } \mathrm{cost}(\mathcal{T}_r^{(\rho)}) \\
&\leqslant \rho \cdot q^{\rho}(\mathcal{T}_r)|\boldsymbol{w}_r| + \rho \cdot \sum_{v \in \mathrm{child}(r)} Q(\rho|\mathrm{T}_v) && \text{Induction hyp.} \\
&\leqslant \rho \cdot q^{\rho}(\mathcal{T}_r)|\boldsymbol{w}_r| + Q\left(\rho \frac{|W(\mathcal{T}_r)| - |\boldsymbol{w}_r|}{|W(\mathcal{T}_r)|} \,\middle|\, \mathrm{T}_r\right) && \text{Since } \mathcal{T}_v \subseteq \mathrm{T}_r \\
&\leqslant Q\left(\rho|\mathrm{T}_r\right)
\end{aligned}$$

The second-to-last inequality follows since Q is defined as the area of the largest weights of the histogram. Including more weights only increases and keeping the length of the integration range the same (equal to $\rho(|W(\mathcal{T}_r)| - |\boldsymbol{w}_r|)$) can only increase the value Q.

The last inequality follows by noting that if $\mathrm{H}(x)$ is the histogram corresponding to the values of $\mathcal{T}_r$, then

$$\begin{aligned}
Q\left(\rho|\mathrm{T}_r\right) - Q\left(\rho \frac{|W(\mathcal{T}_r)| - |\boldsymbol{w}_r|}{|W(\mathcal{T}_r)|} \,\middle|\, \mathrm{T}_r\right) &= \int_{(1-\rho)|W(\mathcal{T}_r)|}^{|W(\mathcal{T}_r)|} \mathrm{H}(x)dx \\
&\quad - \int_{(1-\rho)|W(\mathcal{T}_r)|+\rho|\boldsymbol{w}_r|}^{|W(\mathcal{T}_r)|} \mathrm{H}(x)dx
\end{aligned}$$

$$
\begin{aligned}
&= \int_{(1-\rho)|W(\mathcal{T}_r)|}^{(1-\rho)|W(\mathcal{T}_r)|+\rho|\boldsymbol{w}_r|} H(x)dx \\
&\geqslant \int_{(1-\rho)|W(\mathcal{T}_r)|}^{(1-\rho)|W(\mathcal{T}_r)|+\rho|\boldsymbol{w}_r|} q^{\rho}(\mathcal{T}_r)dx \\
&= \rho q^{\rho}(\mathcal{T}_r)|\boldsymbol{w}_r|
\end{aligned}
$$

where the inequality follows since $H(x) \geqslant q^{\rho}(\mathcal{T}_r)$ for $x \geqslant (1-\rho)|W(\mathcal{T}_r)|$ by the definition of $q^{\rho}(\mathcal{T}_r)$ as the top-r quantile of the weights in $\mathcal{T}_r$.

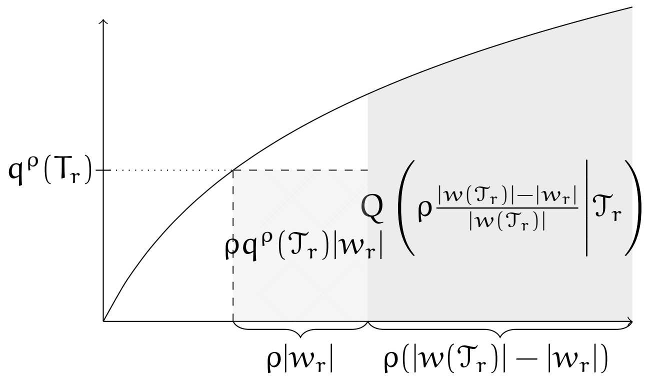
$q^{\rho}(T_r)$
$Q\left(\rho\frac{|w(\mathcal{T}_r)|-|w_r|}{|w(\mathcal{T}_r)|}\middle|\mathcal{T}_r\right)$
$\rho q^{\rho}(\mathcal{T}_r)|w_r|$
$\rho|w_r|$
$\rho(|w(\mathcal{T}_r)|-|w_r|)$

Figure 4.5: Picture depicting the proof above.

$\square$

5 FULLY ADAPTIVE

In this chapter we are moving on to the most powerful of all benchmarks ; the Fully-Adaptive. A primary challenge in approximating PANDORA's BOX with correlations is that the optimal solution can be an adaptive policy that determines which box to open depending on the instantiations of values in all of the boxes opened previously. It is not clear that such a policy can even be described succinctly. Furthermore, the choice of which box to open is complicated by the need to balance two desiderata – finding a low value box quickly versus learning information about the values in unopened boxes (a.k.a. the state of the world or realized scenario) quickly. Indeed, the value contained in a box can provide the algorithm with crucial information about other boxes, and inform the choice of which box to open next; an aspect that is completely missing in the independent values setting studied by Weitzman.

In this chapter we develop the first approximately-optimal policies for the PANDORA's BOX problem with correlated values, via a reduction to Optimal Decision Tree.

Some aspects of the PANDORA's BOX problem have been studied separately in other contexts. For example, in the OPTIMAL DECISION TREE problem (DT) (Guillory and Bilmes, 2009; Li et al., 2020), the goal is to identify an unknown hypothesis, out of m possible ones, by performing a sequence of costly tests, whose outcomes depend on the realized hypothesis. This problem has an informational structure similar to that in PANDORA's BOX. In particular, we can think of every possible joint instantiation of values in boxes as a possible hypothesis, and every opening of a box as a test. The difference between the two problems is that while in OPTIMAL DECISION TREE we want to identify the realized hypothesis exactly, in PANDORA's BOX it suffices to terminate the process as soon as we have found a low value box.

Another closely related problem is the MIN SUM SET COVER (Feige et al., 2004), where boxes only have two kinds of values – *acceptable* or *unacceptable* – and the goal is to find an *acceptable* value as quickly as possible. A primary difference relative to PANDORA's BOX is that unacceptable boxes provide no further information about the values in unopened boxes.

One of the main contributions of our work is to unearth connections between PANDORA's BOX and the two problems described above. We show that PANDORA's BOX is essentially equivalent to a special case of OPTIMAL DECISION TREE (called UNIFORM

Decision Tree or UDT) where the underlying distribution over hypotheses is uniform – the approximation ratios of these two problems are related within log-log factors. Surprisingly, in contrast, the non-uniform DT appears to be harder than non-uniform Pandora's Box. We relate these two problems by showing that both are in turn related to a new version of Min Sum Set Cover, that we call Min-Sum Set Cover (MSSC_f). These connections are summarized in Figure 5.1. We can thus build on the rich history and large collection of results on these problems to offer efficient algorithms for Pandora's Box. We obtain a polynomial time $\tilde{O}(\log m)$ approximation for Pandora's Box, where m is the number of distinct value vectors (a.k.a. scenarios) that may arise; as well as constant factor approximations in subexponential time.

$$\underbrace{\text{PB} \xleftrightarrow{\text{Section 5.3}} \text{UM}}_{\text{Log-log factors}}\underbrace{\text{SSC}_f \xleftrightarrow{\text{Section 5.4}} \text{UDT}}_{\text{Constant factors}}$$

Figure 5.1: A summary of our approximation preserving reductions

It is an important open question whether constant factor approximations exist for Uniform Decision Tree: the best known lower-bound on the approximation ratio is 4 while it is known that it is not NP-hard to obtain super-constant approximations under the Exponential Time Hypothesis. The same properties transfer also to Pandora's Box and Min-Sum Set Cover. Pinning down the tight approximation ratio for any of these problems will directly answer these questions for any other problem in the equivalence class we establish.

The key technical component in our reductions is to find an appropriate stopping rule for Pandora's Box: after opening a few boxes, how should the algorithm determine whether a small enough value has been found or whether further exploration is necessary? We develop an iterative algorithm that in each phase finds an appropriate threshold, with the exploration terminating as soon as a value smaller than the threshold is found, such that there is a constant probability of stopping in each phase. Within each phase then the exploration problem can be solved via a reduction to UDT. The challenge is in defining the stopping thresholds in a manner that allows us to relate the algorithm's total cost to that of the optimal policy.

Related work Optimal decision tree is an old problem studied in a variety of settings (Pattipati and Dontamsetty, 1992; Guillory and Bilmes, 2009; Golovin et al., 2010)), while its most notable application is in active learning settings Guillory and Bilmes (2009), classification tasks (Safavian and Landgrebe, 1991; Bertsimas and Dunn, 2017), data mining (Rokach and Maimon, 2014) and medical diagnosis (Podgorelec et al., 2002). It was proven to be NP-Hard to approximate better than $\log m$ by Hyafil and Rivest Hyafil and Rivest (1976). Since then the problem of finding the best approximation algorithm was an active one (Garey and Graham, 1974; Loveland, 1985; Kosaraju et al., 1999; Dasgupta, 2004; Chakaravarthy et al., 2011, 2009; Gupta et al., 2017; Cicalese et al., 2010; Adler and Heeringa, 2012). Finally, in Guillory and Bilmes (2009) the authors present a greedy algorithm for the most general case of optimal decision tree and in Chakaravarthy et al. (2011) they find the best possible algorithm. For the case of Uniform decision tree less is known, until recently the best algorithm was the same as the optimal decision tree, and the lower bound was 4 (Chakaravarthy et al., 2011). The recent work of Li et al. (2020) showed that there is an algorithm strictly better than $O(\log m)$ for the uniform decision tree.

5.1 Preliminaries

In this paper we study the connections between three different sequential decision making problems – Optimal Decision Tree, Pandora's Box, and Min Sum Set Cover. We describe these problems formally below.

Optimal Decision Tree

In the Optimal Decision Tree problem (denoted DT) we are given a set $\mathcal{S}$ of m scenarios $s \in \mathcal{S}$, each occurring with (known) probability p_s; and n tests $\mathcal{T} = \{T_i\}_{i\in[n]}$, each with cost 1. Nature picks a scenario $s \in \mathcal{S}$ from the distribution p but this scenario is unknown to the algorithm. The goal of the algorithm is to determine which scenario is realized by running a subset of the tests $\mathcal{T}$. When test T_i is run and the realized scenario is s, the test returns a result $T_i(s) \in \mathbb{R}$.

Output. The output of the algorithm is a decision tree where at each node there is a test that is performed, and the branches are the outcomes of the test. In each of the leaves there is an individual scenario that is the only one consistent with the results of the test in the unique path from the root to this leaf. Observe that there is a single leaf corresponding to each scenario s. We can represent the tree as an *adaptive policy* defined as follows:

Definition 5.1 (Adaptive Policy π)**.** *An adaptive policy $\pi : \cup_{X\subseteq\mathcal{T}}\mathbb{R}^X \to \mathcal{T}$ is a function that given a set of tests done so far and their results, returns the next test to be performed.*

Objective. For a given decision tree or policy π, let $\text{cost}_s(\pi)$ denote the total cost of all of the tests on the unique path in the tree from the root to the leaf labeled with scenario s. The objective of the algorithm is to find a policy π that minimizes the average cost $\sum_{s\in\mathcal{S}} p_s \,\text{cost}_s(\pi)$.
We use the term Uniform Decision Tree (UDT) to denote the special case of the problem where $p_s = 1/m$ for all scenarios s.

Pandora's Box

In the Pandora's Box problem we are given n boxes, each with cost $c_i \geqslant 0$ and value v_i. The values $\{v_i\}_{i\in[n]}$ are distributed according to known distribution $\mathcal{D}$. We assume that $\mathcal{D}$ is an arbitrary correlated distribution over vectors $\{v_i\}_{i\in[n]} \in \mathbb{R}^n$. We call vectors of values *scenarios* and use $s = \{v_i\}_{i\in[n]}$ to denote a possible realization of the scenario. As in DT, nature picks a scenario from the distribution D and this realization is a priori unknown to the algorithm. The goal of the algorithm is to pick a box of small value. The algorithm can observe the values realized in the boxes by opening any box i at its respective costs c_i.

Output. The output of the algorithm is an adaptive policy π for opening boxes and a stopping condition. The policy π takes as input a subset of the boxes and their associated values, and either returns the index of a box $i \in [n]$ to be opened next or stops and selects the minimum value seen so far. That is, $\pi : \cup_{X\subseteq[n]}\mathbb{R}^X \to [n] \cup \{\perp\}$ where $\perp$ denotes stopping.

Objective. For a given policy π, let $\pi(s)$ denote the set of boxes opened by the policy prior to stopping when the realized scenario is s. The objective of the algorithm is to minimize the expected cost of the boxes opened plus the minimum value discovered, where the expectation is taken over all possible realizations of the values in each box.[1] Formally the objective is given by

$$\mathbf{E}_{s\sim\mathcal{D}}\left[\min_{i\in\pi(s)} v_{is} + \sum_{i\in\pi(s)} c_i\right],$$

For simplicity of presentation, from now on we assume that $c_i = 1$ for all boxes, but we show in Section 5.A.6 how to adapt our results to handle non-unit costs, without any loss in the approximation factors.

We use UPB to denote the special case of the problem where the distribution $\mathcal{D}$ is uniform over m scenarios.

Min Sum Set Cover with Feedback

In Min Sum Set Cover, we are given n elements and a collection of m sets $\mathcal{S}$ over them, and a distribution $\mathcal{D}$ over the sets. The output of the algorithm is an ordering π over the elements. The cost of the ordering for a particular set $s \in \mathcal{S}$ is the index of the first element in the ordering that belongs to the set s, that is, $\text{cost}_s(\pi) = \min\{i : \pi(i) \in s\}$. The goal of the algorithm is to minimize the expected cost $\mathbf{E}_{s\sim\mathcal{D}}[\text{cost}_s(\pi)]$.

We define a variant of the Min Sum Set Cover problem, called Min Sum Set Cover with Feedback (MSSC_f). As in the original problem, we are given a set of n elements, a collection of m sets $\mathcal{S}$ and a distribution $\mathcal{D}$ over the sets. Nature instantiates a set $s \in \mathcal{S}$ from the distribution $\mathcal{D}$; the realization is unknown to the algorithm. Furthermore, in this variant, each element provides *feedback* to the algorithm when the algorithm "visits" this element; this feedback takes on the value $f_i(s) \in \mathbb{R}$ for element $i \in [n]$ if the realized set is $s \in \mathcal{S}$.

Output. The algorithm once again produces an ordering π over the elements. Observe that the feedback allows the algorithm to adapt its ordering to previously observed values. Accordingly, π is an adaptive policy that maps a subset of the elements and their associated feedback, to the index of another element $i \in [n]$. That is, $\pi : \cup_{X\subseteq[n]}\mathbb{R}^X \to [n]$.

[1]In the original version of the problem studied by Weitzman Weitzman (1979) the values are independent across boxes, and the goal is to maximize the value collected minus the costs paid, in contrast to the minimization version we study here.

Objective. As before, the cost of the ordering for a particular set $s \in \mathcal{S}$ is the index of the first element in the ordering that belongs to the set s, that is, $\text{cost}_s(\pi) = \min\{i : \pi(i) \in s\}$. The goal of the algorithm is to minimize the expected cost $\mathbf{E}_{s\sim\mathcal{D}}[\text{cost}_s(\pi)]$.

We use UMSSC_f to denote the special case of the problem where the distribution $\mathcal{D}$ is uniform over m scenarios.

Commonalities and notation

As the reader has observed, we capture the commonalities between the different problems through the use of similar notation. Scenarios in DT correspond to value vectors in $\mathcal{PB}$ and to sets in MSSC_f; all are denoted by s, lie in the set $\mathcal{S}$, and are drawn by nature from a known joint distribution $\mathcal{D}$. Tests in DT correspond to boxes in $\mathcal{PB}$ and elements in MSSC_f; we index each by $i \in [n]$. The algorithm for each problem produces an adaptive ordering π over these tests/boxes/elements. Test outcomes $T_i(s)$ in DT correspond to box values $v_i(s)$ in $\mathcal{PB}$ and feedback $f_i(s)$ in MSSC_f. We will use the terminology and notation across different problems interchangeably in the rest of the paper.

5.1.1 Modeling Correlation

In this work we study two general ways of modeling the correlation between the values in the boxes.

Explicit Distributions. In this case, $\mathcal{D}$ is a distribution over m *scenarios* where the j'th scenario is realized with probability p_j, for $j \in [m]$. Every scenario corresponds to a fixed and known vector of values contained in each box. Specifically, box i has value $v_{ij} \in \mathbb{R}^+ \cup \{\infty\}$ for scenario j.

5.2 Roadmap of the Reductions and Implications

In Figure 5.2, we give an overview of all the main technical reductions shown in Sections 5.3 and 5.4. An arrow $A \to B$ means that we gave an approximation preserving reduction from problem A to problem B. Therefore an algorithm for B that achieves approximation ratio α gives also an algorithm for A with approximation ratio $O(\alpha)$ (or $O(\alpha \log \alpha)$ in the case of black dashed lines). For the exact guarantees we refer to the formal statement of the respective theorem. The gray lines denote less important claims or trivial reductions (e.g. in the case of A being a subproblem of B).

Figure 5.2: Summary of all our reductions. Bold black lines denote our main theorems, gray dashed are minor claims, and dotted lines are trivial reductions.

5.2.1 Approximating Pandora's Box

Given our reductions and using the best known results for Uniform Decision Tree from Li et al. (2020) we immediately obtain efficient approximation algorithms for Pandora's Box. We repeat the results of Li et al. (2020) below.

Theorem 5.2 (Theorems 3.1 and 3.2 from Li et al. (2020))**.**

- *There is a $O(\log m/\log \mathrm{OPT})$-approximation algorithm for UDT that runs in polynomial time, where OPT is the cost of the optimal solution of the UDT instance.*
- *There is a $\frac{9+\varepsilon}{\alpha}$-approximation algorithm for UDT that runs in time $n^{\tilde{O}(m^{\alpha})}$ for any $\alpha \in (0,1)$.*

Using the results of Theorem 5.2 combined with Theorem 5.8 and Claim 5.16 we get the following corollary.

Corollary 5.3. *From the best-known results for UDT, we have that*

- *There is a $\tilde{O}(\log m)$-approximation algorithm for PB that runs in polynomial time*[2]*.*
- *There is a $\tilde{O}(1/\alpha)$-approximation algorithm for PB that runs in time $n^{\tilde{O}(m^{\alpha})}$ for any $\alpha \in (0,1)$.*

An immediate implication of the above corollary is that it is not NP-hard to obtain a superconstant approximation for PB, formally stated below.

Corollary 5.4. *It is not NP-hard to achieve any superconstant approximation for PB assuming the Exponential Time Hypothesis.*

[2]If additionally the possible number of outcomes is a constant K, this gives a $O(\log m)$ approximation without losing an extra logarithmic factor, since $\mathrm{OPT} \geqslant \log_K m$, as observed by Li et al. (2020).

Observe that the logarithmic approximation achieved in Corollary 5.3 loses a $\log\log m$ factor (hence the $\tilde{O}$) as it relies on the more complex reduction of Theorem 5.8. If we choose to use the more direct reduction of Theorem 5.18 to the Optimal Decision Tree where the tests have non-unit costs (which also admits a $O(\log m)$-approximation Gupta et al. (2017); Kambadur et al. (2017)), we get the following corollary.

Corollary 5.5. *There exists an efficient algorithm that is $O(\log m)$-approximate for Pandora's Box and with or without unit-cost boxes.*

5.2.2 Constant approximation for Partially Adaptive PB

Moving on, we show how our reduction can be used to obtain and improve the results of Chawla et al. (2020). Recall that in Chawla et al. (2020) the authors presented a constant factor approximation algorithm against a Partially Adaptive benchmark where the order of opening boxes must be fixed up front.

In such a case, the reduction of Section 5.3 can be used to reduce PB to the standard Min Sum Set Cover (i.e. without feedback), which admits a 4-approximation Feige et al. (2004).

Corollary 5.6. *There exists a polynomial time algorithm for PB that is $O(1)$-competitive against the partially adaptive benchmark.*

The same result applies even in the case of non-uniform opening costs. This is because a 4-approximate algorithm for Min Sum Set Cover is known even when elements have arbitrary costs Munagala et al. (2005). The case of non-uniform opening costs has also been considered for Pandora's Box by Chawla et al. (2020) but only provide an algorithm to handle polynomially bounded opening costs.

5.3 Connecting Pandora's Box and $MSSC_f$

In this section we establish the connection between Pandora's Box and Min-Sum Set Cover. We show that the two problems are equivalent up to logarithmic factors in approximation ratio.

One direction of this equivalence is easy to see in fact: Min-Sum Set Cover is a special case of Pandora's Box. Note that in both problems we examine boxes/elements in an adaptive order. In Pandora's Box we stop when we find a sufficiently small value; in MSSC_f we stop when we find an element that belongs to the instantiated scenario. To establish a formal connection, given an instance of MSSC_f, we can define the "value" of each element i in scenario s as being 0 if the element belongs to the set s and as being $L + f_i(s)$ for some sufficiently large value L where $f_i(s)$ is the feedback of element i for set s. This places the instance within the framework of Pandora's Box and a Pandora's Box algorithm can be used to solve it. We formally describe this reduction in Section 5.A.2 of the Appendix.

Claim 5.7. *If there exists an* $\alpha(n, m)$*-approximation algorithm for* $\mathcal{PB}$ *then there exists a* $\alpha(n, m)$*-approximation for* MSSC_f*.*

The more interesting direction is a reduction from $\mathcal{PB}$ to MSSC_f. In fact we show that a general instance of $\mathcal{PB}$ can be reduced to the simpler *uniform* version of Min-Sum Set Cover. We devote the rest of this section to proving the following theorem.

Theorem 5.8 (Pandora's Box to MSSC_f)**.** *If there exists an* $a(n, m)$ *approximation algorithm for* UMSSC_f *then there exists a* $O(\alpha(n+m, m^2) \log \alpha(n+m, m^2))$*-approximation for* $\mathcal{PB}$*.*

Guessing a stopping rule and an intermediate problem

The feedback structure in $\mathcal{PB}$ and MSSC_f is quite similar, and the main component in which the two problems differ is the stopping condition. In MSSC_f, an algorithm can stop examining elements as soon as it finds one that "covers" the realized set. In $\mathcal{PB}$, when the algorithm observes a value in a box, it is not immediately apparent whether the value is small enough to stop or whether the algorithm should probe further, especially if the scenario is not fully identified. The key to relating the two problems is to "guess" an appropriate stopping condition for $\mathcal{PB}$, namely an appropriate threshold T such that as soon as the algorithm observes a value smaller than this threshold, it stops. We say that the realized scenario is "covered".

To formalize this approach, we introduce an intermediate problem called *Pandora's Box with costly outside option* T (also called *threshold*), denoted by $\mathrm{PB}_{\leqslant T}$. In this version the objective is to minimize the cost of finding a value $\leqslant T$, while we have the extra option to quit searching by opening an *outside option* box of cost T. We say that a scenario is *covered* in a given run of the algorithm if it does not choose the outside option box T.

We show that Pandora's Box can be reduced to PB$_{\leqslant T}$ with a logarithmic loss in approximation factor, and then PB$_{\leqslant T}$ can be reduced to Min-Sum Set Cover with a constant factor loss. The following two results capture the details of these reductions.

Claim 5.9. *If there exists an $\alpha(n, m)$ approximation algorithm for UMSSC$_f$ then there exists an $3\alpha(n+m, m^2)$-approximation for UPB$_{\leqslant T}$.*

It is also worth noting that PB$_{\leqslant T}$ is a special case of the Adaptive Ranking problem which directly implies a $\log m$ approximation factor (given in Kambadur et al. (2017)).

Main Lemma 5.10. *Given a polynomial-time $\alpha(n, m)$-approximation algorithm for UPB$_{\leqslant T}$, there exists a polynomial-time $O(\alpha(n, m) \log \alpha(n, m))$-approximation for PB.*

The relationship between PB$_{\leqslant T}$ and Min-Sum Set Cover is relatively straightforward and requires explicitly relating the structure of feedback in the two problems. We describe the details in Section 5.A.2 of the Appendix.

Putting it all together. The proof of Theorem 5.8 follows by combining Claim 5.9 with Lemmas 5.11 and 5.10 presented in the following sections. Proofs of Claims 5.7, 5.9 deferred to Section 5.A.2 of the Appendix. The rest of this section is devoted to proving Lemmas 5.11 and 5.10. The landscape of reductions shown in this section is presented in Figure 5.3.

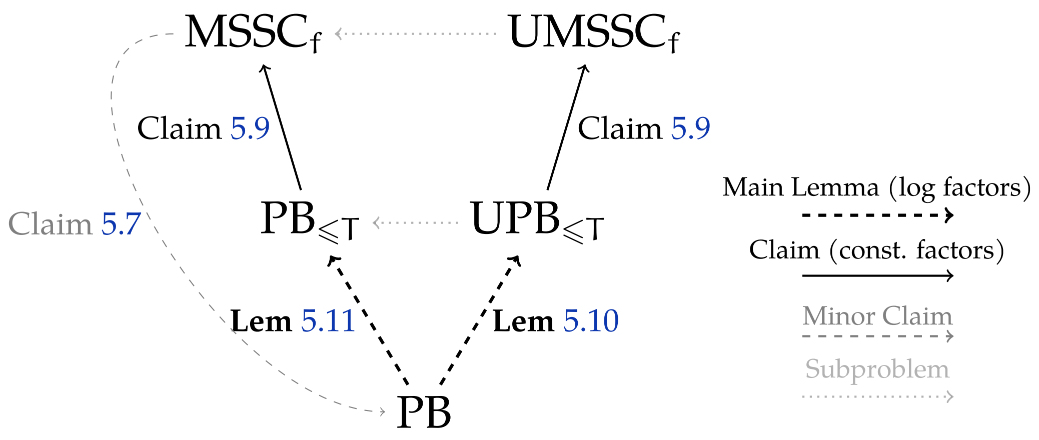

Figure 5.3: Reductions shown in this section. Claim 5.9 alongside Lemmas 5.11 and 5.10 are part of Theorem 5.8.

5.3.1 Reducing Pandora's Box to $PB_{\leqslant T}$

Recall that a solution to Pandora's Box involves two components ; (1) the order in which to open boxes and (2) a stopping rule. The goal of the reduction to $PB_{\leqslant T}$ is to simplify the stopping rule of the problem, by making values either 0 or ∞, therefore allowing us to focus on the order in which boxes are opened, rather than which value to stop at. We start by presenting our main tool, a reduction to Min-Sum Set Cover in Section 5.3.1.1 and then improve upon that to reduce from the **uniform** version of $MSSC_f$ (Section 5.3.1.2).

5.3.1.1 Main Tool

The high level idea in this reduction is that we repeatedly run the algorithm for $PB_{\leqslant T}$ with increasingly larger value of T with the goal of covering some mass of scenarios at every step. The thresholds for every run have to be cleverly chosen to guarantee that enough mass is covered at every run. The distributions on the boxes remain the same, and this reduction does not increase the number of boxes, therefore avoiding the issues faced by the naive reduction given in Section 5.A.1 of the Appendix. Formally, we show the following lemma.

Main Lemma 5.11. *Given a polynomial-time $\alpha(n, m)$-approximation algorithm for $PB_{\leqslant T}$, there exists a polynomial-time $O(\alpha(n, m) \log \alpha(n, m))$-approximation for PB.*

Algorithm 9: Reduction from PB to $PB_{\leqslant T}$.

```
Input: Oracle $\mathcal{A}(T)$ for $PB_{\leqslant T}$, set of all scenarios $\mathcal{S}$.
1 $i \leftarrow 0$ // Number of current Phase
2 while $\mathcal{S} \neq \emptyset$ do
3     Use $\mathcal{A}$ to find smallest $T_i$ via Binary Search s.t.
        $\mathbf{Pr}$ [accepting the outside option $T_i$] $\leqslant 0.2$
4     Call the oracle $\mathcal{A}(T_i)$ on set $\mathcal{S}$ to obtain policy $\pi_i$
5     $\mathcal{S} \leftarrow \mathcal{S} \setminus$ {scenarios with total cost $\leqslant T_i$}
6 end
7 for $i \leftarrow 0$ to $\infty$ do
8     Run policy $\pi_i$ until it terminates and selects a box, or accumulates probing
        cost $T_i$.
9 end
```

We will now analyze the policy produced by this algorithm.

Proof of Main Lemma 5.11. We start with some notation. Given an instance $\mathcal{I}$ of PB, we repeatedly run $PB_{\leqslant T}$ in *phases*. Phase i consists of running $PB_{\leqslant T}$ with threshold T_i on a sub instance of the original problem where we are left with a smaller set of scenarios, with their probabilities reweighted to sum to 1. Call this set of scenarios $\mathcal{S}_i$ for phase i and the corresponding instance $\mathcal{I}_i$. After every phase i, we remove the probability mass that was covered[3], and run $PB_{\leqslant T}$ on this new instance with a new threshold T_{i+1}. In each phase, the boxes, costs and values remain the same, but the stopping condition changes: thresholds T_i increase in every subsequent phase. The thresholds are chosen such that at the end of each phase, 0.8 of the remaining probability mass is covered. The reduction process is formally shown in Algorithm 9.

Accounting for the cost of the policy. We first note that the total cost of the policy in phase i conditioned on reaching that phase is at most $2T_i$: if the policy terminates in that phase, it selects a box with value at most T_i. Furthermore, the policy incurs probing cost at most T_i in the phase. We can therefore bound the total cost of the policy as $\leqslant 2\sum_{i=0}^{\infty}(0.2)^i T_i$.

We will now relate the thresholds T_i to the cost of the optimal PB policy for $\mathcal{I}$. To this end, we define corresponding thresholds for the optimal policy that we call p-*thresholds*. Let $\pi^*_{\mathcal{I}}$ denote the optimal PB policy for $\mathcal{I}$ and let c_s denote the cost incurred by $\pi^*_{\mathcal{I}}$ when scenario i is realized. A p-threshold is the minimum possible threshold T such that at most p mass of the scenarios has cost more than T in PB, formally defined below.

Definition 5.12 (p-Threshold)**.** *Let $\mathcal{I}$ be an instance of PB and c_s be the cost of scenario $s \in \mathcal{S}$ in $\pi^*_{\mathcal{I}}$, we define the p-threshold as*

$$t_p = \min\{T : \boldsymbol{Pr}\left[c_s > T\right] \leqslant p\}.$$

The following two lemmas relate the cost of the optimal policy to the p-thresholds, and the p-thresholds to the thresholds T_i our algorithm finds. The proofs of both

[3]Recall, a scenario is *covered* if it does not choose the outside option box.

lemmas are deferred to Section 5.A.3 of the Appendix. We first formally define a *sub-instance* of the given PANDORA's Box instance.

Definition 5.13 (Sub-instance)**.** *Let $\mathcal{I}$ be an instance of $\{PB_{\leqslant T}, PB\}$ with set of scenarios $\mathcal{S}_\mathcal{I}$ each with probability $p_s^\mathcal{I}$. For any $q \in [0,1]$ we call $\mathcal{I}'$ a q-sub instance of $\mathcal{I}$ if $\mathcal{S}_{\mathcal{I}'} \subseteq \mathcal{S}_\mathcal{I}$ and $\sum_{s\in\mathcal{S}_{\mathcal{I}'}} p_s^\mathcal{I} = q$.*

Lemma 5.14. (*Optimal Lower Bound*) *Let $\mathcal{I}$ be the instance of PB. For any $q < 1$, any $\alpha > 1$, and $\beta \geqslant 2$, for the optimal policy $\pi_\mathcal{I}^*$ for PB it that*

$$\text{cost}(\pi_\mathcal{I}^*) \geqslant \sum_{i=0}^{\infty} \frac{1}{\beta\alpha} \cdot (q)^i \, t_{q^i/\beta\alpha}.$$

Lemma 5.15. *Given an instance $\mathcal{I}$ of PB; an α-approximation algorithm $\mathcal{A}_T$ to $PB_{\leqslant T}$; and any $q < 1$ and $\beta \geqslant 2$, suppose that the threshold T satisfies*

$$T \geqslant t_{q/(\beta\alpha)} + \beta\alpha \sum_{\substack{c_s \in [t_q, t_{q/(\beta\alpha)}] \\ s\in\mathcal{S}}} c_s \frac{p_s}{q}.$$

Then if $\mathcal{A}_T$ is run on a q-sub instance of $\mathcal{I}$ with threshold T, at most a total mass of $(2/\beta)q$ of the scenarios pick the outside option box T.

Calculating the thresholds. For every phase i we choose a threshold T_i such that $T_i = \min\{T : \mathbf{Pr}[c_s > T] \leqslant 0.2\}$ i.e. at most 0.2 of the probability mass of the scenarios are not covered. In order to select this threshold, we do binary search starting from $T = 1$, running every time the α-approximation algorithm for $PB_{\leqslant T}$ with outside option box T and checking how many scenarios select it. We denote by $\text{Int}_i = [t_{(0.2)^i}, t_{(0.2)^i/(10\alpha)}]$ the relevant interval of costs at every run of the algorithm, then by Lemma 5.15 for $\beta = 10$, we know that for remaining total probability mass $(0.2)^i$, any threshold which satisfies

$$T_i \geqslant t_{(0.2)^{i-1}/10a} + 10\alpha \sum_{\substack{s\in\mathcal{S} \\ c_s \in \text{Int}_i}} c_s \frac{p_s}{(0.2)^i}$$

also satisfies the desired covering property, i.e. at least 0.8 mass of the current scenarios is covered. Therefore the threshold T_i found by our binary search satisfies the following

$$T_i = t_{(0.2)^{i-1}/10\alpha} + 10\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s \frac{p_s}{(0.2)^i}. \tag{5.1}$$

Bounding the final cost. To bound the final cost, we recall that at the end of every phase we cover 0.8 of the remaining scenarios. Furthermore, we observe that each threshold T_i is charged in the above Equation (5.1) to optimal costs of scenarios corresponding to intervals of the form $\mathrm{Int}_i = [t_{(0.2)^i}, t_{(0.2)^i/(10\alpha)}]$. Note that these intervals are overlapping. We therefore get

$$\begin{aligned}
\mathrm{cost}(\pi_{\mathcal{J}}) &\leqslant 2\sum_{i=0}^{\infty}(0.2)^i T_i \\
&= 2\sum_{i=0}^{\infty}\left((0.2)^i t_{(0.2)^{i-1}/10\alpha} + 10\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s p_s\right) && \text{From equation (5.1)}\\
&\leqslant 4\cdot 10\alpha\pi^*_{\mathcal{J}} + 20\alpha\sum_{i=0}^{\infty}\sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s p_s \\
&\leqslant 40\alpha\log\alpha\cdot\pi^*_{\mathcal{J}}.
\end{aligned}$$

where for the second to last inequality we used Lemma 5.14 for $\beta = 10$, $q = 0.2$, and the last inequality follows since each scenario with cost c_s can belong to at most $\log\alpha$ intervals, therefore we get the theorem. □

Notice the generality of this reduction; the distributions on the values are preserved, and we did not make any more assumptions on the scenarios or values throughout the proof. Therefore we can apply this tool regardless of the type of correlation or the way it is given to us, e.g. we could be given a parametric distribution, or an explicitly given distribution, as we see in the next section.

5.3.1.2 An Even Stronger Tool

Moving one step further, we show that if we instead of $\mathrm{PB}_{\leqslant T}$ we had an α-approximation algorithm for $\mathrm{UPB}_{\leqslant T}$ we can obtain the same guarantees as the ones described in Lemma 5.11. Observe that we cannot directly use Algorithm 9 since the oracle now requires that all scenarios have the same probability, while this might not be the case in the initial PB instance. The theorem stated formally follows.

Main Lemma 5.10. *Given a polynomial-time $\alpha(n, m)$-approximation algorithm for UPB$_{\leqslant T}$, there exists a polynomial-time $O(\alpha(n, m) \log \alpha(n, m))$-approximation for PB.*

We are going to highlight the differences with the proof of Main Lemma 5.11, and show how to change Algorithm 9 to work with the new oracle, that requires the scenarios to have uniform probability. The function **Expand** shown in Algorithm 10 is used to transform the instance of scenarios to a uniform one where every scenario has the same probability by creating multiple copies of the more likely scenarios. The function is formally described in Algorithm 11 in Section 5.A.4 of the Appendix, alongside the proof of Main Lemma 5.10.

Algorithm 10: Reduction from PB to UPB$_{\leqslant T}$.

```
Input: Oracle $\mathcal{A}(T)$ for UPB$_{\leqslant T}$, set of all scenarios $\mathcal{S}$, $c = 1/10, \delta = 0.1$.
1 $i \leftarrow 0$ // Number of current Phase
2 while $\mathcal{S} \neq \emptyset$ do
3     Let $\mathcal{L} = \left\{ s \in \mathcal{S} : p_s \leqslant c \cdot \frac{1}{|\mathcal{S}|} \right\}$ // Remove low probability scenarios
4     $\mathcal{S}' = \mathcal{S} \setminus \mathcal{L}$
5     $\mathcal{UI} = \text{Expand}(\mathcal{S}')$
6     In instance $\mathcal{UI}$ use $\mathcal{A}$ to find smallest $T_i$ via Binary Search s.t.
       $\mathbf{Pr}[\text{accepting } T_i] \leqslant \delta$
7     Call the oracle $\mathcal{A}(T_i)$
8     $\mathcal{S} \leftarrow \left(\mathcal{S}' \setminus \{s \in \mathcal{S}' : c_s \leqslant T_i\}\right) \cup \mathcal{L}$
9 end
```

5.4 Connecting MSSC$_f$ and Optimal Decision Tree

In this section we establish the connection between Min-Sum Set Cover and Optimal Decision Tree. We show that the uniform versions of these problems are equivalent up to constant factors in approximation ratio. The results of this section are summarized in Figure 5.4 and the two results below.

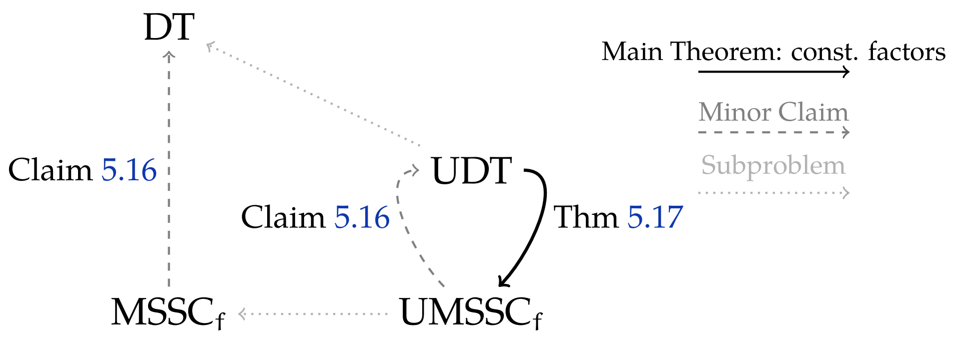

Figure 5.4: Summary of reductions in Section 5.4

Claim 5.16. *If there exists an $\alpha(n, m)$-approximation algorithm for DT (UDT) then there exists a $(1 + \alpha(n, m))$-approximation algorithm for MSSC$_f$ (resp. UMSSC$_f$).*

Theorem 5.17 (Uniform Decision Tree to UMSSC$_f$)**.** *Given an $\alpha(m, n)$-approximation algorithm for UMSSC$_f$ then there exists an $O(\alpha(n + m, m))$-approximation algorithm for UDT.*

The formal proofs of these statements can be found in Section 5.A.5 of the Appendix. Here we sketch the main ideas.

One direction of this equivalence is again easy to see. The main difference between Optimal Decision Tree and MSSC$_f$ is that the former requires scenarios to be exactly identified whereas in the latter it suffices to simply find an element that covers the scenario. In particular, in MSSC$_f$ an algorithm could cover a scenario without identifying it by, for example, covering it with an element that covers multiple scenarios. To reduce MSSC$_f$ to DT we simply introduce extra feedback into all of the elements of the MSSC$_f$ instance such that the elements isolate any scenarios they cover. (That is, if the algorithm picks an element that covers some subset of scenarios, this element provides feedback about which of the covered scenarios materialized.) This allows us to relate the cost of isolation and the cost of covering to within the cost of a single additional test, implying Claim 5.16.

Proof Sketch of Theorem 5.17. The other direction is more complicated, as we want to ensure that covering implies isolation. Given an instance of UDT, we create a special element for each scenario which is the unique element covering the scenario and also isolates the scenario from all other scenarios. The intention is that an algorithm for

MSSC_f on this new instance only chooses the special isolating element in a scenario after it has identified the scenario. If that happens, then the algorithm's policy is a feasible solution to the UDT instance and incurs no extra cost. The problem is that an algorithm for MSSC_f over the modified instance may use the special covering element before isolating a scenario. We argue that this choice can be "postponed" in the policy to a point at which isolation is nearly achieved without incurring too much extra cost. This involves careful analysis of the policy's decision tree and we present details in the appendix.

Why our reduction does not work for DT. Our analysis above heavily uses the fact that the probabilities of all scenarios in the UDT instance are equal. This is because the "postponement" of elements charges increased costs of some scenarios to costs of other scenarios. In fact, our reduction above fails in the case of non-uniform distributions over scenarios – it can generate an MSSC_f instance with optimal cost much smaller than that of the original DT instance.

To see this, consider an example with m scenarios where scenarios 1 through $m-1$ happen with probability $\varepsilon/(m-1)$ and scenario m happens with probability $1-\varepsilon$. There are $m-1$ tests of cost 1 each. Test i for $i \in [m-1]$ isolates scenario i from all others. Observe that the optimal cost of this DT instance is at least $(1-\varepsilon)(m-1)$ as all $m-1$ tests need to be run to isolate scenario m. Our construction of the MSSC_f instance adds another isolating test for scenario m. A solution to this instance can use this new test at the beginning to identify scenario m and then run other tests with the remaining ε probability. As a result, it incurs cost at most $(1-\varepsilon)+\varepsilon(m-1)$, which is a factor of $1/\varepsilon$ cheaper than that of the original DT instance.

5.A Appendix for Chapter 5

5.A.1 A Naive Reduction to $\mathrm{PB}_{\leqslant T}$

In this section we present a straightforward reduction from Pandora's Box to $\mathrm{PB}_{\leqslant T}$ as an alternative to Theorem 5.8. This reduction has a simpler construction compared to the reduction of Section 5.3, and does not lose a logarithmic factor in the approximation, it however faces the following issues.

1. It incurs an extra computational cost, since it adds a number of boxes that depends on the size of the values' support.

2. It requires opening costs, which means that the oracle for Pandora's Box with outside option should be able to handle non-unit costs on the boxes, even if the original $\mathcal{PB}$ problem had unit-cost boxes.

We denote by $\text{PB}^c_{\leqslant T}$ the version of Pandora's Box with outside option that has **non-unit** cost boxes, and formally state the guarantees of our naive reduction below.

Theorem 5.18. *For n boxes and m scenarios, given an $\alpha(n, m)$-approximation algorithm for $\text{PB}^c_{\leqslant T}$ for arbitrary T, there exits a $2\alpha(n \cdot |supp(v)|, m)$-approximation for PB that runs in polynomial time in the number of scenarios, number of boxes, and the number of values.*

Figure 5.5 summarizes all the reductions from $\mathcal{PB}$ to $\text{PB}_{\leqslant T}$ and in Table 5.1 we compare the properties of the naive reduction of this section, to the ones show in Section 5.3. The main differences are that there is a blow-up in the number of boxes that depends on the support, while losing only constant factors in the approximation.

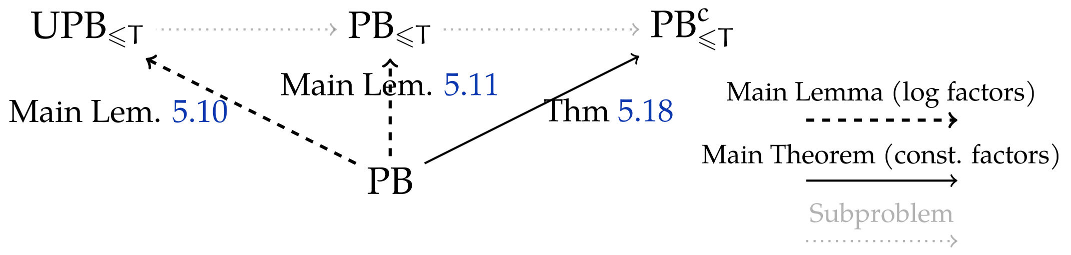
$\text{UPB}_{\leqslant T}$
$\text{PB}_{\leqslant T}$
$\text{PB}^c_{\leqslant T}$
Main Lem. 5.10
Main Lem. 5.11
Thm 5.18
PB
Main Lemma (log factors)
Main Theorem (const. factors)
Subproblem

Figure 5.5: Reductions shown in Section 5.3.1

	Reducing PB to $\text{PB}^c_{\leqslant T}$**, Theorem 5.18**	$(\mathcal{U})$**PB**$_{\leqslant T}$**, Lemma 5.11 (5.10)**
Costs of boxes	Introduces non-unit costs	Maintains costs
Probabilities	Maintains probabilities	Maintains probabilities (Makes probabilities uniform)
# of extra scenarios	0	0
# of extra boxes	$n \cdot \text{supp}(v)$	0
Approximation loss	$2\alpha(n \cdot \text{supp}(v), m)$	$O(\alpha(n, m) \log a(n, m))$

Table 5.1: Differences of reductions of Theorems 5.18, and the Main Lemmas 5.11 and 5.10 that comprise Theorem 5.8.

The main idea is that we can move the information about the values contained in the boxes into the cost of the boxes. We do achieve this effect by creating one new box for every (box, value)-pair. Note, that doing this risks losing the information about the realized scenario that the original boxes revealed. To retain this information, we keep the original boxes, but replace their values by high values. The high values guarantee the effect of the new boxes is retained. Now, we can formalize this intuition.

$\mathrm{PB}_{\leqslant T}$ Instance. Given an instance $\mathcal{I}$ of PB, we construct an instance $\mathcal{I}'$ of $\mathrm{PB}_{\leqslant T}$. We need T to be sufficiently large so that the outside option is never chosen. The net effect is that a policy for PB is easily inferred from a policy for $\mathrm{PB}_{\leqslant T}$. We define the instance $\mathcal{I}'$ to have the same scenarios s_i and same scenario probabilities p_i as $\mathcal{I}$. We choose $T = \infty$[4], and define the new values by $v'_{i,j} = v_{i,j} + T + 1$. Note that all of these values will be larger than T and so a feasible policy cannot terminate after receiving such a value. At the same time, these values ensure the same branching behaviour as before since each distinct value is mapped one to one to a new distinct value. Next, we add additional “final" boxes for each pair (j, v) where j is a box and v a potential value of box j. Each “final" box (j, v) has cost $c_j + v$. Box (j, v) has value 0 for the scenarios where box j gives exactly value v and values $T + 1$ for all other scenarios. Formally,

$$v'_{i,(j,v)} = \begin{cases} 0 & \text{if } v_{i,j} = v \\ T+1 & \text{else} \end{cases}$$

Intuitively, these “final” boxes indicate to a policy that this will be the last box opened, and so its values, which are at least that of the best values of the boxes chosen, should now be taken into account in the cost of the solution.

In order to prove Theorem 5.18, we use two key lemmas. In Lemma 5.19 we show that the optimal value for the transformed instance $\mathcal{I}'$ of $\mathrm{PB}_{\leqslant T}$ is not much higher than the optimal value for original instance $\mathcal{I}$. In Lemma 5.20 we show how to obtain a policy for the initial instance with values, given a policy for the problem with a threshold.

Lemma 5.19. *Given the instance $\mathcal{I}$ of PB and the constructed instance $\mathcal{I}'$ of $\mathrm{PB}_{\leqslant T}$ it holds that*

$$c(\pi^*_{\mathcal{I}'}) \leqslant 2c(\pi^*_{\mathcal{I}}).$$

[4]We set T to a value larger than $\sum_i c_i + \max_{i,j} v_{ij}$.

Proof. We show that given an optimal policy for PB, we can construct a feasible policy π' for $\mathcal{I}'$ such that $c(\pi_0) \leqslant 2c(\pi^*_{\mathcal{I}})$. We construct the policy π' by opening the same boxes as π and finally opening the corresponding “values" box, in order to find the 0 needed to stop.

Fix any scenario i, and suppose box j achieved the smallest value $V_{i,j}$ of all boxes opened under scenario i. Since j is opened, in the instance $\mathcal{I}'$ we open box $(j, v_{i,j})$, and from the construction of $\mathcal{I}'$ we have that $v'_{i,(j,v_{i,j})} = 0$. Since on every branch we open a box with values 0[5], we see that π' is a feasible policy for $\mathcal{I}'$. Under scenario i, we have that the cost of $\pi(i)$ is

$$c(\pi(i)) = \min_{k \in \pi(i)} v_{i,k} + \sum_{k \in \pi(i)} c_k.$$

In contrast, the minimum cost following $\pi'(i)$ is 0 and there is the additional cost of the “values" box. Formally, the cost of $\pi'(i)$ is

$$c(\pi'(i)) = 0 + \sum_{k \in \pi(i)} c_k + c_{(j,v_{i,j})} = \min_{k \in \pi(i)} v_{i,k} + \sum_{k \in \pi(i)} c_k + c_j = c(\pi(i)) + c_j$$

Since c_j appears in the cost of $\pi(i)$, we know that $c(\pi(i)) \geqslant c_j$. Thus, $c(\pi'(i)) = c(\pi(i)) + c_j \leqslant 2c(\pi(i))$, which implies that $c(\pi') \leqslant 2c(\pi^*_{\mathcal{I}})$ for our feasible policy π'. Observing that $c(\pi') \geqslant c(\pi^*_{\mathcal{I}'})$ for any policy, completes the proof. □

Lemma 5.20. *Given a policy π' for the constructed instance $\mathcal{I}'$ of $PB_{\leqslant T}$, there exists a feasible policy π for the instance $\mathcal{I}$ of PB with no larger expected cost. Furthermore, any branch of π can be constructed from π' in polynomial time.*

Proof of Lemma 5.20. We construct a policy π for $\mathcal{I}$ using the policy π'. Fix some branch of π'. If π' opens box j along this branch, we define policy π to open the same box along this branch. When π' opens a “final” box (j, v), we define the policy π to open box j if it has not been opened already.

Next, we show this policy π has no larger expected cost than π'. There are two cases to consider depending on where the “final” box (j, v) is opened:

1. “Final" box (j, v) is at a leaf of π': since π' has finite expected cost and this is the first “final" box we encountered, the result must be 0. Therefore, under π the

[5] π opens at least one box.

values will be v by definition of $\mathcal{I}'$. Observe that in this case, $c(\pi) \leqslant c(\pi')$ since the (at most) extra v paid by π for the value term, has already been paid by the box cost in π' when box (j, v) was opened.

2. "Final" box (j, v) is at an intermediate node of π': after π opens box j, we copy the subtree of π' that follows the 0 branch into the branch of π that follows the v branch. Also, we copy the subtree of π' that follows the ∞_1 branch into each branch that has a value different from v (the non-v branches). The cost of this new subtree is c_j instead of the original $c_j + v$. The v branch may accrue an additional cost of v or smaller if j was not the smallest values box on this branch, so in total, the v branch has cost at most its original cost.

   However, the non-v branches have a v term removed going down the tree. Specifically, since the feedback of (j, v) down the non-v branch was ∞_1, some other box with 0 values had to be opened at some point, and this box is still available to be used as the final values for this branch later on (since if this branch already had a 0, it would have stopped). Thus, the cost of this subtree is at most that originally, and has one fewer "final" box opened.

Putting these cases together implies that $c(\pi) \leqslant c(\pi')$.

Lastly, we argue that any branch of π can be computed efficiently. To compute a branch for π, we follow the corresponding branch of π'. As we go along this branch, we open box j whenever π' opens box (j, v) and remember the feedback. We use the feedback to know which boxes of π' to open in the future. Hence, we can compute a branch of π from π' in polynomial time. □

We are now ready to give the proof of Lemma 5.18.

Proof of Lemma 5.18. Suppose we have an α-approximation for $\text{PB}_{\leqslant T}$. Given an instance $\mathcal{I}$ to PB, we construct the instance $\mathcal{I}'$ for $\text{PB}_{\leqslant T}$ as described and then run the approximation algorithm on $\mathcal{I}'$ to get a policy $\pi_{\mathcal{I}'}$. Next, we prune the tree as described in Lemma 5.20 to get a policy, $\pi_{\mathcal{I}}$ of no worse cost. Our policy will use time at most polynomially more than the policy for $\text{PB}_{\leqslant T}$ since each branch of $\pi_{\mathcal{I}}$ can be computed in polynomial time from $\pi_{\mathcal{I}'}$. Hence, the runtime is polynomial in the size of $\mathcal{I}'$. We also note that we added at most mn total "final" boxes to construct our new instance $\mathcal{I}'$, and so this algorithm will run in polynomial time in m and n. Thus, by

Lemma 5.20 and Lemma 5.19 we know the cost of the constructed policy is

$$c(\pi) \leqslant c(\pi') \leqslant \alpha c(\pi^*_{\mathcal{I}'}) \leqslant 2\alpha c(\pi^*_{\mathcal{I}})$$

Hence, this algorithm is a 2α-approximation for PB.

□

5.A.2 Proofs from Section 5.3

Claim 5.7. *If there exists an $\alpha(n, m)$-approximation algorithm for $\mathcal{PB}$ then there exists a $\alpha(n, m)$-approximation for MSSC$_f$.*

Proof of Claim 5.7. Let $\mathcal{I}$ be an instance of MSSC$_f$. We create an instance $\mathcal{I}'$ of $\mathcal{PB}$ the following way: for every set s_j of $\mathcal{I}$ that gives feedback f_{ij} when element e_i is selected, we create a scenario s_j with the same probability and whose value for box i, is either 0 if $e_i \in s_j$ or $\infty_{f_{ij}}$ otherwise, where $\infty_{f_{ij}}$ denotes an extremely large value which is different for different values of the feedback f_{ij}. Observe that any solution to the $\mathcal{PB}$ instance gives a solution to the MSSC$_f$ at the same cost and vice versa. □

Claim 5.9. *If there exists an $\alpha(n, m)$ approximation algorithm for UMSSC$_f$ then there exists an $3\alpha(n + m, m^2)$-approximation for UPB$_{\leqslant T}$.*

Before formally proving this claim, recall the correspondence of scenarios and boxes of PB-type problems, to elements and sets of MSSC-type problems. The idea for the reduction is to create T copies of sets for each scenario in the initial PB$_{\leqslant T}$ instance and one element per box, where if the price a box gives for a scenario i is $< T$ then the corresponding element belongs to all T copies of the set i. The final step is to "simulate" the outside option T, for which we we create T elements where the k'th one belongs only to the k'th copy of each set.

Proof of Claim 5.9. Given an instance $\mathcal{I}$ of UPB$_{\leqslant T}$ with outside cost box b_T, we construct the instance $\mathcal{I}'$ of UMSSC$_f$ as follows.

Construction of the instance. For every scenario s_i in the initial instance, we create T sets denoted by s_{ik} where $k \in [T]$. Each of these sets has equal probability $p_{ik} = 1/(mT)$. We additionally create one element e^B per box B, which belongs to every set s_{ik} for all k iff $v_{Bi} < T$ in the initial instance, otherwise gives feedback v_{Bi}. In order to simulate box b_T without introducing an element with non-unit cost, we use a sequence of T *outside option* elements e^T_k where $e^T_k \in s_{ik}$ for all $i \in [m]$ i.e. element e^T_{ik} belongs to “copy k" of every set [6].

Construction of the policy. We construct policy $\pi_{\mathcal{I}}$ by ignoring any outside option elements that $\pi_{\mathcal{I}'}$ selects until $\pi_{\mathcal{I}'}$ has chosen at least $T/2$ such elements, at which point $\pi_{\mathcal{I}}$ takes the outside option box b_T. To show feasibility we need that for every scenario either b_T is chosen or some box with $v_{ij} \leqslant T$. If b_T is not chosen, then less than $T/2$ isolating elements were chosen, therefore in instance of UMSSC$_f$ some sub-sets will have to be covered by another element e^B, corresponding to a box. This corresponding box however gives a value $\leqslant T$ in the initial UPB$_{\leqslant T}$ instance.

Approximation ratio. Let s_i be any scenario in $\mathcal{I}$. We distinguish between the following cases, depending on whether there are outside option tests on s_i's branch.

1. **No outside option tests** on s_i's branch: scenario s_i contributes equally in both policies, since absence of *isolating elements* implies that all copies of scenario s_i will be on the same branch (paying the same cost) in both $\pi_{\mathcal{I}'}$ and $\pi_{\mathcal{I}}$
2. **Some outside option tests** on i's branch: for this case, from Lemma 5.21 we have that $c(\pi_{\mathcal{I}}(s_i)) \leqslant 3c(\pi_{\mathcal{I}'}(s_i))$.

Putting it all together we get

$$c(\pi_{\mathcal{I}}) \leqslant 3c(\pi_{\mathcal{I}'}) \leqslant 2\alpha(n+m, m^2)c(\pi^*_{\mathcal{I}'}) \leqslant 3\alpha(n+m, m^2)c(\pi^*_{\mathcal{I}}),$$

where the second inequality follows since we are given an α approximation and the last inequality since if we are given an optimal policy for UPB$_{\leqslant T}$, the exact same policy is also feasible for any $\mathcal{I}'$ instance of UDT, which has cost at least $c(\pi^*_{\mathcal{I}'})$. We also used that $T \leqslant m$, since otherwise the initial policy would never take the outside option. □

Lemma 5.21. *Let $\mathcal{I}$ be an instance of UPB$_{\leqslant T}$, and $\mathcal{I}'$ the instance of UMSSC$_f$ constructed by the reduction of Claim 5.9. For a scenario s_i, if there is at least one outside option test run in $\pi_{\mathcal{I}}$, then $c(\pi_{\mathcal{I}}(s_i)) \leqslant 3c(\pi_{\mathcal{I}'}(s_i))$.*

[6]Observe that there are exactly T possible options for k for any set. Choosing all these elements costs T and covers all sets thus simulating b_T.

Proof. For the branch of scenario s_i, denote by M the box elements chosen before there were $T/2$ *outside option* elements, and by N the number of *outside option* elements in $\pi_{\mathcal{J}'}$. Note that the smallest cost is achieved if all the outside option elements are chosen first[7]. The copies of scenario s_i can be split into two groups; those that were isolated **before** $T/2$ *outside option* elements were chosen, and those that were isolated **after**. We distinguish between the following cases, based on the value of N.

1. $N \geqslant T/2$: in this case each of the copies of s_i that are isolated after pays at least $M + T/2$ for the initial box elements and the initial sequence of *outside option* elements. For the copies isolated before, we lower bound the cost by choosing all *outside option* elements first.

   The cost of all the copies in $\pi_{\mathcal{J}'}$ then is at least

$$\begin{aligned}\sum_{j=1}^{K_i}\sum_{k=1}^{T/2}\frac{cp_\ell}{T}k + \sum_{j=1}^{K_i}\sum_{k=T/2+1}^{T}\frac{cp_\ell}{T}(T/2+M) &= cp_i\frac{\frac{T}{2}(\frac{T}{2}+1)}{2T} + cp_i\frac{\frac{T}{2}(T/2+M)}{T}\\ &\geqslant cp_i(3T/8 + M/2)\\ &\geqslant \frac{3}{8}p_i(T+M)\end{aligned}$$

   Since $N \geqslant T/2$, policy $\pi_{\mathcal{J}}$ will take the outside option box for s_i, immediately after choosing the M initial boxes corresponding to the box elements. So, the total contribution s_i has on the expected cost of $\pi_{\mathcal{J}}$ is at most $p_i(M+T)$ in this case. Hence, we have that s_i's contribution in $\pi_{\mathcal{J}}$ is at most $\frac{8}{3} \leqslant 3$ times s_i's contribution in $\pi_{\mathcal{J}'}$.

2. $N < T/2$: policy $\pi_{\mathcal{J}}$ will only select the M boxes (corresponding to *box* elements) and this was sufficient for finding a value less than T. The total contribution of s_i on $c(\pi_{\mathcal{J}})$ is exactly p_iM. On the other hand, since $N < T/2$ we know that at least half of the copies will pay M for all of the box elements. The cost of all the copies is at least

$$\sum_{j=1}^{K_i}\sum_{k=N}^{T}\frac{cp_\ell}{T}M = cp_i\frac{T-N}{T}M \geqslant cp_iM/2,$$

[7]Since the outside option tests cause some copies to be isolated and so can reduce their cost.

therefore, the contribution s_i has on $c(\pi_{\mathcal{I}'})$ is at least $cp_iM/2$. Hence, we have $c(\pi_{\mathcal{I}}) \leqslant 3c(\pi_{\mathcal{I}'})$

□

5.A.3 Proofs from subsection 5.3.1.1

Lemma 5.15. *Given an instance $\mathcal{I}$ of PB; an α-approximation algorithm $\mathcal{A}_T$ to $PB_{\leqslant T}$; and any $q < 1$ and $\beta \geqslant 2$, suppose that the threshold T satisfies*

$$T \geqslant t_{q/(\beta\alpha)} + \beta\alpha \sum_{\substack{c_s \in [t_q, t_{q/(\beta\alpha)}] \\ s \in \mathcal{S}}} c_s \frac{p_s}{q}.$$

Then if $\mathcal{A}_T$ is run on a q-sub instance of $\mathcal{I}$ with threshold T, at most a total mass of $(2/\beta)q$ of the scenarios pick the outside option box T.

Proof. Consider a policy $\pi_{\mathcal{I}_q}$ which runs $\pi^*_{\mathcal{I}}$ on the instance $\mathcal{I}_q$; and for scenarios with cost $c_s \geqslant t_{q/(\beta\alpha)}$ aborts after spending this cost and chooses the outside option T. The cost of this policy is:

$$c(\pi^*_{\mathcal{I}_q}) \leqslant c(\pi_{\mathcal{I}_q}) = \frac{T + t_{q/(\beta\alpha)}}{\beta\alpha} + \sum_{\substack{c_s \in [t_q, t_{q/(10\alpha)}] \\ s \in \mathcal{S}}} c_s \frac{p_s}{q}, \tag{5.2}$$

By our assumption on T, this cost is at most $2T/\beta\alpha$. On the other hand since $\mathcal{A}_T$ is an α-approximation to the optimal we have that the cost of the algorithm's solution is at most

$$\alpha c(\pi^*_{\mathcal{I}_q}) \leqslant \frac{2T}{\beta}$$

Since the expected cost of $\mathcal{A}_T$ is at most $2T/\beta$, then using Markov's inequality, we get that $\mathbf{Pr}\left[c_s \geqslant T\right] \leqslant (2T/\beta)/T = 2/\beta$. Therefore, $\mathcal{A}_T$ covers at least $1 - 2/\beta$ mass every time. □

Lemma 5.14. (*Optimal Lower Bound*) *Let $\mathcal{I}$ be the instance of PB. For any $q < 1$, any $\alpha > 1$, and $\beta \geqslant 2$, for the optimal policy $\pi^*_{\mathcal{I}}$ for PB it that*

$$\text{cost}(\pi^*_{\mathcal{I}}) \geqslant \sum_{i=0}^{\infty} \frac{1}{\beta\alpha} \cdot (q)^i \, t_{q^i/\beta\alpha}.$$

Proof. In every interval of the form $\mathcal{I}_i = [t_{q^i}, t_{q^i/(\beta\alpha)}]$ the optimal policy for PB covers at least $1/(\beta\alpha)$ of the probability mass that remains. Since the values belong in the interval $\mathcal{I}_i$ in phase i, it follows that the minimum possible value that the optimal policy might pay is t_{q^i}, i.e. the lower end of the interval. Summing up for all intervals, we get the lemma. □

5.A.4 Proofs from subsection 5.3.1.2

Algorithm 11: Expand: rescales and returns an instance of UPB.

```
   Input: Set of scenarios $\mathcal{S}$
1  Scale all probabilities by c such that $c \sum_{s \in \mathcal{S}} p_s = 1$
2  Let $p_{\min} = \min_{s \in \mathcal{S}} p_s$
3  $\mathcal{S}' =$ for each $s \in \mathcal{S}$ create $p_s/p_{\min}$ copies
4  Each copy has probability $1/|\mathcal{S}'|$
5  return $\mathcal{S}'$
```

Main Lemma 5.10. *Given a polynomial-time $\alpha(n, m)$-approximation algorithm for UPB$_{\leqslant T}$, there exists a polynomial-time $O(\alpha(n, m) \log \alpha(n, m))$-approximation for PB.*

Proof. The proof in this case follows the steps of the proof of Theorem 5.11, and we are only highlighting the changes. The process of the reduction is the same as Algorithm 9 with the only difference that we add two extra steps; (1) we initially remove all low probability scenarios (line 3 - remove at most c fraction) and (2) we add them back after running UPB$_{\leqslant T}$ (line 8). The reduction process is formally shown in Algorithm 10.

Calculating the thresholds. For every phase i we choose a threshold T_i such that $T_i = \min\{T : \mathbf{Pr}[c_s > T] \leqslant \delta\}$ i.e. at most δ of the probability mass of the scenarios are not covered, again using binary search as in Algorithm 9. We denote by $\text{Int}_i = [t_{(1-c)(\delta+c)^i}, t_{(1-c)(\delta+c)^i/(\beta\alpha)}]$ the relevant interval of costs at every run of the algorithm, then by Lemma 5.15, we know that for remaining total probability mass $(1-c)(\delta+c)^i$, any threshold which satisfies

$$T_i \geqslant t_{(1-c)(\delta+c)^{i-1}/\beta\alpha} + \beta\alpha \sum_{\substack{s \in \mathcal{S} \\ c_s \in \text{Int}_i}} c_s \frac{p_s}{(1-c)(\delta+c)^i}$$

also satisfies the desired covering property, i.e. at least $(1-2/\beta)(1-c)(\delta+c)$ mass of the current scenarios is covered. Therefore the threshold T_i found by our binary search satisfies

$$T_i = t_{(1-c)(\delta+c)^{i-1}/\beta\alpha} + \beta\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s \frac{p_s}{(1-c)(\delta+c)^i}. \tag{5.3}$$

Following the proof of Theorem 5.11, **Constructing the final policy** and **Accounting for the values** remain exactly the same as neither of them uses the fact that the scenarios are uniform.

Bounding the final cost. Using the guarantee that at the end of every phase we cover $(\delta+c)$ of the scenarios, observe that the algorithm for $\mathrm{PB}_{\leqslant T}$ is run in an interval of the form $\mathrm{Int}_i = [t_{(1-c)(\delta+c)^i}, t_{(1-c)(\delta+c)^i/(\beta\alpha)}]$. Note also that these intervals are overlapping. Bounding the cost of the final policy $\pi_{\mathcal{J}}$ for all intervals we get

$$\begin{aligned}
\pi_{\mathcal{J}} &\leqslant \sum_{i=0}^{\infty}(1-c)(\delta+c)^i T_i \\
&= \sum_{i=0}^{\infty}\left((1-c)(\delta+c)^i t_{(1-c)(\delta+c)^{i-1}/\beta\alpha} + \beta\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s p_s\right) && \text{From equation (5.3)} \\
&\leqslant 2\cdot\beta\alpha\pi^*_{\mathcal{J}} + \beta\alpha\sum_{i=0}^{\infty}\sum_{\substack{s\in\mathcal{S}\\ c_s\in \mathrm{Int}_i}} c_s p_s && \text{Using Lemma 5.14} \\
&\leqslant 2\beta\alpha\log\alpha\cdot\pi^*_{\mathcal{J}},
\end{aligned}$$

where the inequalities follow similarly to the proof of Theorem 5.11. Choosing $c=\delta=0.1$ and $\beta=20$ we get the theorem. □

5.A.5 Proofs from Section 5.4

Claim 5.16. *If there exists an $\alpha(n,m)$-approximation algorithm for DT (UDT) then there exists a $(1+\alpha(n,m))$-approximation algorithm for MSSC$_f$ (resp. UMSSC$_f$).*

Proof of Claim 5.16. Let $\mathcal{J}$ be an instance of MSSC$_f$. We create an instance $\mathcal{J}'$ of DT the following way: for every set s_j we create a scenario s_j with the same probability and for every element e_i we create a test T_{e_i} with the same cost, that gives full feedback whenever an element belongs to a set, otherwise returns only the element's feedback f_{ij}. Formally, the i-test under scenario s_j returns

$$T_{e_i}(s_j) = \begin{cases} \text{“The feedback is } f_{ij}\text{”} & \text{If } e_i \notin s_j \\ \text{“The scenario is } j\text{”} & \text{else ,} \end{cases}$$

therefore the test isolates scenario j when $e_i \in s_j$.

Constructing the policy. Given a policy π' for the instance $\mathcal{I}'$ of DT, we can construct a policy π for $\mathcal{I}$ by selecting the element that corresponds to the test π' chose. When π' finishes, all scenarios are identified and for any scenario s_j either (1) there is a test in π' that corresponds to an element in s_j (in the instance $\mathcal{I}$) or (2) there is no such test, but we can pay an extra $\min_{i \in s_j} c_i$ to select the lowest cost element in this set[8].

Observe also that in this instance of DT if we were given the optimal solution, it directly translates to *a* solution for MSSC$_f$ with the same cost, therefore

$$c(\pi^*_{\mathcal{I}}) \leqslant c(\pi'_{\mathcal{I}'}) = c(\pi^*_{\mathcal{I}'}) \tag{5.4}$$

Bounding the cost of the policy. As we described above the total cost of the policy is

$$\begin{aligned} c(\pi) &\leqslant c(\pi_{\mathcal{I}'}) + \mathbf{E}_{s \in \mathcal{S}}\left[\min_{i \in s} c_i\right] \\ &\leqslant c(\pi_{\mathcal{I}'}) + c(\pi^*_{\mathcal{I}}) \\ &\leqslant a(n,m)c(\pi^*_{\mathcal{I}'}) + c(\pi^*_{\mathcal{I}}) \\ &= (1 + a(n,m))c(\pi^*_{\mathcal{I}}), \end{aligned}$$

where in the last inequality we used equation (5.4).

Note that for this reduction we did not change the probabilities of the scenarios, therefore if we had started with uniform probabilities and had an oracle to UDT, we would still get an $a(n,m) + 1$ algorithm for UMSSC$_f$. □

In the reduction proof of Theorem 5.17, we are using the following two lemmas, that show that the policy constructed for UDT via the reduction is feasible and has

[8]Since the scenario is identified, we know exactly which element this is.

bounded cost.

Lemma 5.22. *Given an instance $\mathcal{I}'$ of UDT and the corresponding instance $\mathcal{I}$ of UMSSC$_f$ in the reduction of Theorem 5.17, the policy $\pi_{\mathcal{I}'}$ constructed for UDT is feasible.*

Proof of Lemma 5.22. It suffices to show that every scenario is isolated. Fix a scenario s_i. Observe that s_i's branch has chosen the isolating element E^i in the UMSSC$_f$ solution, since that is the the only element that belongs to set s_i. Let S be the set of scenarios just before E^i is chosen and note that by definition $s_i \in S$.

If $|S| = 1$, then since $\pi_{\mathcal{I}'}$ runs tests giving the same branching behavior by definition of $\pi_{\mathcal{I}'}$, and s_i is the only scenario left, we have that the branch of $\pi_{\mathcal{I}'}$ isolates scenario s_i.

If $|S| > 1$ then all scenarios/sets in $S \setminus \{s_i\}$ are not covered by choosing element E^i, therefore they are covered at strictly deeper leaves in the tree. By induction on the depth of the tree, we can assume that for each scenario $s_j \in (S \setminus \{s_i\})$ is isolated in $\pi_{\mathcal{I}'}$. We distinguish the following cases based on when we encounter E^i among the isolating elements on s_i's branch.

1. E^i **was the first isolating element chosen on the branch**: then policy $\pi_{\mathcal{I}'}$ ignores element E^i. Since every leaf holds a unique scenario in $S \setminus \{s_i\}$, if we ignore s_i it follows some path of tests and either be isolated or end up in a node that originally would have had only one scenario, as shown in Figure 5.6. Since there are only two scenarios at that node, policy $\pi_{\mathcal{I}'}$ runs the cheapest test distinguishing s_i from that scenario.

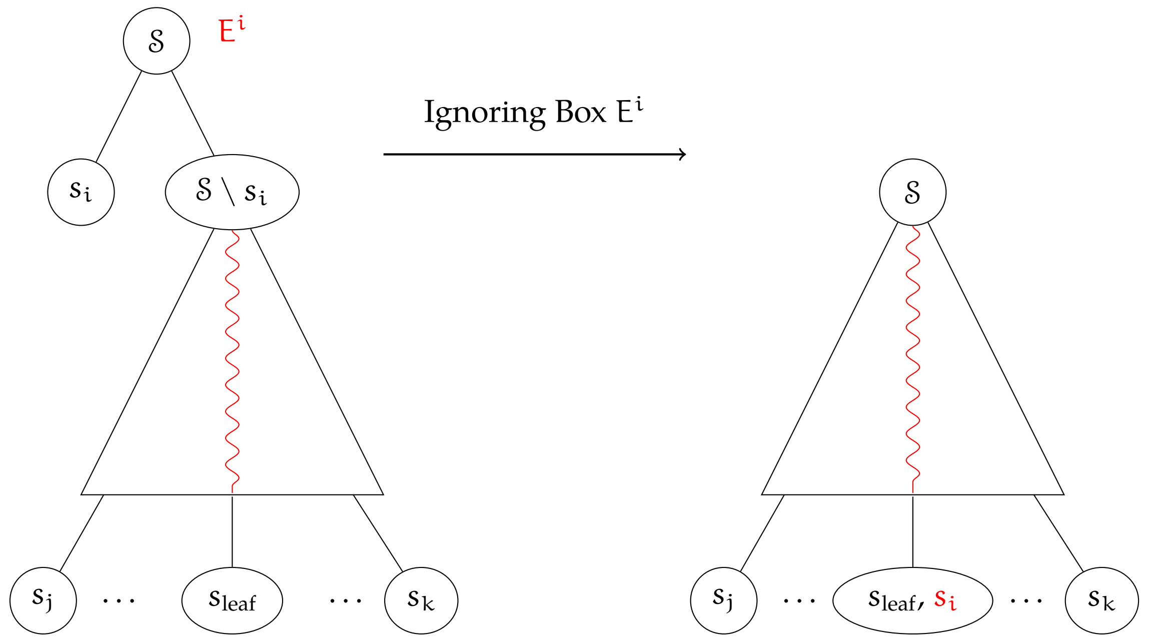

Figure 5.6: Case 1: $\mathcal{S}$ is the set of scenarios remaining when E^i is chosen, s_{leaf} is the scenario that s_i ends up with.

2. **A different element** E^j **was chosen before** E^i: by our construction, instead of ignoring E^i we now run the cheapest test that distinguishes s_i from s_j, causing i and j to go down separate branches, as shown in figure 5.7. We apply the induction hypothesis again to the scenarios in these sub-branches, therefore, both s_i and s_j are either isolated or end up in a node with a single scenario and then get distinguished by the last case of $\pi_{\mathcal{J}'}$'s construction.

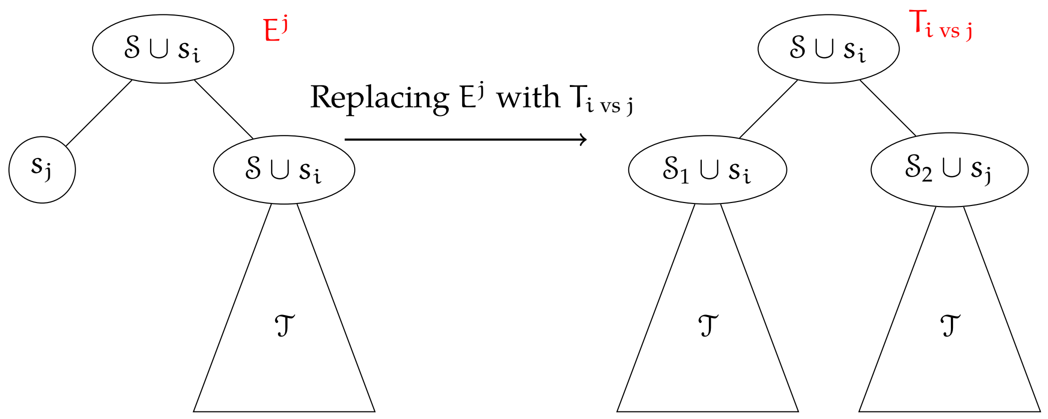

Figure 5.7: Case 2: run test $T_{i \text{ vs } j}$ to distinguish s_i and s_j. Sets $\mathcal{S}_1$ and $\mathcal{S}_2$ partition $\mathcal{S}$

Hence, $\pi_{\mathcal{I}'}$ is isolating for any scenario s_i. Also, notice that any two scenarios that have isolating boxes on the same branch will end up in distinct subtrees of the lower node. □

Lemma 5.23. *Given an instance $\mathcal{I}$ of UMSSC$_f$ and an instance $\mathcal{I}'$ of UDT, in the reduction of Theorem 5.17 it holds that*

$$c(\pi_{\mathcal{I}'}) \leqslant 2c(\pi_{\mathcal{I}}).$$

Proof of Lemma 5.23. Let s_i be any scenario in $\mathcal{S}$. We use induction on the number of isolating boxes along s_i's branch in $\mathcal{I}'$. Initially observe that E^i will always exist in s_i's branch, in any feasible solution to $\mathcal{I}$. We use $c(E^j)$ and $c(T_k)$ to denote the costs of box E^j and test T_k, for any $k \in [n]$ and $j \in [n+m]$.

1. **Only E^i is on the branch**: since E^i will be ignored, we end up with s_i and some other not yet isolated scenario, let s_{leaf} be that scenario. To isolate s_i and s_{leaf} we run the cheapest test that distinguishes between these. From the definition of the cost of E^i we know that $c(T_{s_i \text{ vs } s_{\text{leaf}}}) \leqslant c(E^i)$. Additionally, since $c(s_i) \leqslant c(s_{\text{leaf}})$ and both s_{leaf} and s_i have probability $1/m$, overall we have $c(\pi_{\mathcal{I}}) \leqslant 2c(\pi_{\mathcal{I}'})$. This is also shown in Figure 5.6.

2. **More than one isolating elements are on the branch**: similarly, observe that for any extra isolating element E^j we encounter, we substitute it with a test that distinguishes between s_i and s_j and costs at most $c(E^j)$. Given that $c(s_i) \leqslant c(s_{\text{leaf}})$ and scenarios are uniform, we again have $c(\pi_{\mathcal{I}}) \leqslant 2c(\pi_{\mathcal{I}'})$.

□

Theorem 5.17 (Uniform Decision Tree to UMSSC$_f$)**.** *Given an $\alpha(m, n)$-approximation algorithm for UMSSC$_f$ then there exists an $O(\alpha(n+m, m))$-approximation algorithm for UDT.*

Proof of Theorem 5.17. We begin by giving the construction of the policy in the reduction, and showing the final approximation ratio.

Constructing the policy. Given a policy $\pi_{\mathcal{I}}$ for the instance of UMSSC$_f$, we construct a policy $\pi_{\mathcal{I}'}$. For any test element B_j that $\pi_{\mathcal{I}}$ selects, $\pi_{\mathcal{I}'}$ runs the equivalent test T_j. For the *isolating elements* E^i we distinguish the following cases.

1. If $\pi_{\mathcal{J}}$ selects an *isolating element* E^i for the first time on the current branch, then $\pi_{\mathcal{J}'}$ ignores this element but remembers the set/scenario s_i, which E^i belonged to.

2. If $\pi_{\mathcal{J}}$ selects another *isolating element* E^j after some E^i on the branch, then $\pi_{\mathcal{J}'}$ runs the minimum cost test that distinguishes scenario s_j from s_k where E^k was the most recent *isolating element* chosen on this branch prior to E^j.

3. If we are at the end of $\pi_{\mathcal{J}}$, there can be at most 2 scenarios remaining on the branch, so $\pi_{\mathcal{J}'}$ runs the minimum cost test that distinguishes these two scenarios.

By Lemma 5.22, we have that the above policy is feasible for UDT.

Approximation ratio. From Lemma 5.23 we have that $c(\pi_{\mathcal{J}'}) \leqslant 2c(\pi_{\mathcal{J}})$. For the optimal policy, we have that $c(\pi^*_{\mathcal{J}}) \leqslant 3c(\pi^*_{\mathcal{J}'})$. This holds since if we have an optimal solution to UDT, we can add an *isolating element* at every leaf to make it feasible for UMSSC_f, by only increasing the cost by a factor of 3[9], which means that $c(\pi^*_{\mathcal{J}})$ will be less than this transformed UMSSC_f solution. Overall, if $\pi_{\mathcal{J}}$ is computed from an $\alpha(n, m)$-approximation for UMSSC_f, we have

$$c(\pi_{\mathcal{J}'}) \leqslant 2c(\pi_{\mathcal{J}}) \leqslant 2\alpha(n+m, m)c(\pi^*_{\mathcal{J}}) \leqslant 6\alpha(n+m, m)c(\pi^*_{\mathcal{J}'})$$

□

5.A.6 Boxes with Non-Unit Costs: Revisiting our Results

In the original Pandora's Box problem, denoted by PB^c, each box i has a different known cost $c_i > 0$. Similarly we denote the non-unit cost version of both decision tree-like problems and Min Sum Set Cover-like problems by adding a superscript ^{c} to the problem name. Specifically, we now define DT^c, UDT^c, MSSC^c_f and UMSSC^c_f, where the tests (elements) have non-unit cost for the decision tree (min sum set cover) problems. We revisit our results and describe how our reductions change to incorporate non-unit cost boxes (summary in Figure 5.8).

[9]This is because for every two scenarios, the UDT solution must distinguish between them, but one of these scenarios is the max scenario from the definition of T_j, for which we pay less than T_j

Figure 5.8: Summary of all the reductions with non-unit costs. The only result that needs a changed proof is Corollary 5.25 highlighted in bold (previously Theorem 5.17).

Note also, that even though the known results for Optimal Decision Tree (e.g. Guillory and Bilmes (2009); Gupta et al. (2017)) handle non-unit test costs, the currently known works for Uniform Decision Tree do not. If however there is an algorithm for Uniform Decision Tree with non-unit costs, our reductions will handle this obtaining the same approximation guarantees.

5.A.6.1 Connecting Pandora's Box and MSSC$_f$

$$\begin{tikzcd}
\mathrm{MSSC}_f^c \arrow[dr, dashed, bend right, "\text{Claim 5.7}"'] & & \mathrm{UMSSC}_f^c \arrow[ll, dotted] \\
\mathrm{PB}^c_{\leqslant T} \arrow[u, "\text{Claim 5.9}"] & & \mathrm{UPB}^c_{\leqslant T} \arrow[ll, dotted] \arrow[u, "\text{Claim 5.9}"'] \\
 & \mathrm{PB}^c \arrow[ul, dashed, "\textbf{Lem } 5.11"] \arrow[ur, dashed, "\textbf{Lem } 5.10"'] &
\end{tikzcd}$$

Main Lemma (log factors) $\dashrightarrow$

Claim (const. factors) $\longrightarrow$

Minor Claim $\dashrightarrow$

Subproblem $\cdots\rightarrow$

Figure 5.9: Reductions shown in this section. The solid lines are part of Corollary 5.24.

All the results of this section hold as they are when we change all versions to incorporate costs. We did not use the fact that the costs are unit in any of the proofs of Claim 5.7, Claim 5.9 or Lemmas 5.11, 5.10. We formally restate the main theorem of Section 5.3 as the following corollary, where the only change is that it now holds for the cost versions of the problems.

Corollary 5.24 (Pandora's Box to MSSC$_f$ with non-unit costs)**.** *If there exists an $\alpha(n, m)$ approximation algorithm for MSSC$_f^c$ then there exists a $O(\alpha(n+m, m^2)\log\alpha(n+m, m^2))$-approximation for PB^c. The same result holds if the initial algorithm is for UMSSC$_f^c$.*

5.A.6.2 Connecting MSSC_f and Optimal Decision Tree

In this section the reduction of Theorem 5.17 uses the fact that the costs are uniform. However we can easily circumvent this and obtain corollary 5.25. Using this, the results for the non-unit costs versions are summarized in Figure 5.10.

$$\begin{tikzcd}
\mathrm{DT}^c & \\
& \mathrm{UDT}^c \arrow[ul, dotted] \arrow[d, bend left, "\text{Cor 5.25}"] \\
\mathrm{MSSC}^c_f \arrow[uu, dashed, "\text{Claim 5.16}"] & \mathrm{UMSSC}^c_f \arrow[u, dashed, bend left, "\text{Claim 5.16}"] \arrow[l, dotted]
\end{tikzcd}$$

$$\begin{tikzcd}
{} \arrow[r, "\text{Main Theorem (const. factors)}"] & {} \\
{} \arrow[r, dashed, "\text{Minor Claim}"] & {} \\
{} \arrow[r, dotted, "\text{Subproblem}"] & {}
\end{tikzcd}$$

Figure 5.10: Summary of reductions for non unit cost boxes.

Corollary 5.25 (Uniform Decision Tree with costs to UMSSC^c_f)**.** *Given an $\alpha(m,n)$-approximation algorithm for UMSSC^c_f then there exists an $O(\alpha(n+m,m))$-approximation algorithm for UDT^c.*

Proof. The proof follows exactly the same way as the proof of Theorem 5.17 with one change: the cost of an isolating element is the minimum cost test needed to isolate s_i from scenario s_k where s_k is the scenario that maximizes this quantity. Formally, if $c(i,k) = \min\{c_j | T_j(i) \neq T_j(k)\}$, then $c(B^i) = \max_{k\in[m]} c(i,k)$. The reduction follows the exact steps as the one we described in Section 5.A.5. $\square$

Part II

Robustness in the prior knowledge

6 INTRODUCTION & OVERVIEW OF RESULTS

In the previous part we discussed cases where the prior distribution on our input data is fully known and given to us. This, however, is an unrealistic assumption when we deal with real life situations. In experimental settings, it is usually the case that the instruments used cannot have infinite precision and could have design flaws which lead to uncertain and noisy measurements (Moffat, 1988; Rabinovich, 2006; Hughes and Hase, 2010). In some extreme cases, we may not be able to even obtain data on our input at all, yet we are still required to solve a problem.

We study different cases on the prior information given to us, through the lens of Pandora's Box.

6.1 Overview of Results

We discuss three different regimes of partial knowledge on the prior distribution $\mathcal{D}$. Initially instead of having access to the joint prior distribution we only assume that an oracle can return samples to us (Chapter 7). Then we move on to the fully pessimistic model where we have no information at all on the prior distribution (Chapter 8). Finally in Chapter 9, we discuss a setting that is equivalent to a noisy version of Optimal Decision Tree.

6.1.1 Sample Access

Instead of having a matrix as in Table 2.1, that gives us exactly the probability of all the possible scenarios that can happen, we only have access to an oracle that returns samples from this joint distribution. We first show that against the Fully-Adaptive benchmark we have no hope of obtaining any meaningful approximation. A simple way to see this is via the example shown in Figure 6.1. Imagine that there is a box with a 0 price for every scenario, and the optimal knows an encoding function that gives them the exact 0 box. In this example, the optimal opens boxes i and j, sees some very high values, that cannot potentially pick, and using the most significant bits of these, gives them the location of the box. In order for us to learn this function, and always be able to find the 0 we need to sample every scenario.

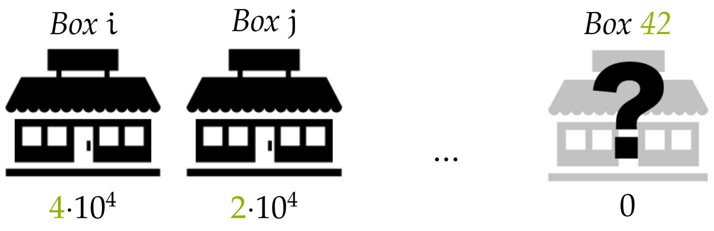

Figure 6.1: Example on how why any strategy will fail against the Fully-Adaptive when we are only given sample access to $\mathcal{D}$.

For the rest of Section 7.1 we revisit our previous results against the Partially-Adaptive benchmark when only given sample access to the joint prior distribution. We initially show how all our results of Chapter 3, even for complex constraints still hold when we draw an adequate number of samples at the beginning. Specifically, we show that given an α-approximation algorithm for a scenario-aware Partially-Adaptive (SPA) strategy, then combining ski-rental with a sampling step before gives us the desired result. The theorem is stated informally below.

Main Theorem (Informal Theorem 7.2)**.** *Suppose there exists an algorithm for* P*ANDORA's Box that returns an SPA strategy achieving an α-approximation. Then, for any $\epsilon > 0$, there exists an algorithm that runs in time polynomial in n and $1/\epsilon$ and returns a PA strategy with competitive ratio $\frac{e}{e-1}(1+\epsilon)\alpha$, where $n = |\mathcal{B}|$.*

Following this, we also show that our simpler algorithm of Chapter 4, when conditioning on $V_b > \sigma_b$, still gives us the same constant approximation within $(1+\varepsilon)$ for a suitable number of samples. We state the main theorem below.

Main Theorem (Informal Theorem 7.3)**.** *For any $\varepsilon, \delta > 0$, using $m = \text{poly}(n, 1/\varepsilon, \log(1/\delta))$ samples from $\mathcal{D}$, Our algorithm obtains a $4.428 + \varepsilon$ approximation policy against the Partially-Adaptive optimal, with probability at least $1 - \delta$, and where $n = |\mathcal{B}|$.*

6.1.2 Unknown Distributions

Switching to our more pessimistic model, we now are required to solve PANDORA's Box for T rounds, while the distribution is unknown but each time we can use the information of the previous rounds. Our goal in this case is to minimize α-approximate regret as defined below, where $\mathcal{A}(t)$ is the cost of our algorithm at round $t \in [T]$ and $\text{OPT}(t)$ the cost of the optimal for the same round. In this setting, the prices inside the boxes can be adversarially selected.

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

As we discussed in Section 1.1, there are two settings arising in this problem, depending on the amount of information we see after the game of each day ends.

- *Full information setting*: at the end of each day we see all the prices inside the boxes.
- *Bandit setting*: at the end of each day we only see the prices in the boxes we chose to open.

Our algorithm first relaxes the initial integer problem a continuous convex one and then uses our main theorem (stated informally below) that allows us to reduce our problem to an online convex optimization one.

Main Theorem (Informal Theorem 8.4)**.** *Given Π 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 αRegret(T) for Π.

Combining this with Theorems 3.5, 3.12 and 3.15 of Chapter:3 we get the following corollary.

Corollary 6.1 (Competing against PA, Corollary 8.7 and 8.11)**.** *In the full information and bandit setting, we give an algorithm that 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***

6.1.2.1 Adding Context

In Section 8.2 we consider adding the extra information of *context* at the beginning of every round. The adversary in this case can select the distributions at each round, and they are independent, however we still have no knowledge of them. We enforce the extra requirement of a *realizability assumption* as a minimal assumption so that the problem is not intractable[1]. Formally the assumption is stated below

Our framework works similarly to the fully unknown distributions case of the previous section, and we reduce our initial Contextual Pandora's Box to a simpler Linear-Quadratic Online Regression one. *Linear-Quadratic Online Regression* is the special case of online regression where the loss function $\ell(z)$ is chosen to be a linear-quadratic function of the form

$$H_c(z) = \frac{1}{2}\mathrm{ReLU}(z)^2 - cz \tag{6.1}$$

for some parameter $c > 0$.

Having reduced our problem to a simpler regression one is not enough though; we also require that using an estimation of Weitzman's reservation values in the original algorithm, we can still get some guarantees. This is shown in the following theorem.

Main Theorem (Informal Theorem 8.19)**.** *For a Pandora's Box instance with n boxes with distributions $\mathcal{D}$, costs $\mathbf{c}$ and optimal reservation values $\boldsymbol{\sigma}^*$, Weitzman's Algorithm, run with reservation values $\boldsymbol{\sigma}$ incurs cost at most*

$$\mathrm{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}; \mathbf{c}) \leqslant \mathrm{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*; \mathbf{c}) + \|\mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma}) - \mathbf{c}\|_1.$$

where $c_{\mathcal{D}}(\sigma)$ are the costs corresponding to the instance where $\boldsymbol{\sigma}$ are the optimal reservation values.

Main Theorem (Informal Theorem 8.17)**.** *Given an oracle with expected regret $r(T)$ for Linear-Quadratic Online Regression, we give an algorithm that achieves $2n\sqrt{Tr(T)}$ regret for the Contextual Pandora's Box problem. In particular, if the regret $r(T)$ is sublinear in T, our algorithm achieves sublinear regret.*

6.1.3 Noisy Data

Finally, in Chapter 9, we switch to Pandora's Box where we given a mixture of m product distributions, and show how this is tightly connected to a noisy version of decision tree. Our benchmark is once again the Fully-Adaptive optimal. We design a

[1]We fully justify this assumption in Section 8.2

dynamic program that finds a constant approximation to this problem ; the theorem is stated below.

Main Theorem (Informal Theorem 9.1)**.** *Let* $c(DP)$ *and OPT be our algorithm's and the optimal's cost respectively. For any* $\beta > 0$*, and* $\text{UB} = \frac{m^2}{\varepsilon^2} \log \frac{m^2 T}{c_{\min} \beta}$*. Then it holds that*

$$c(DP) \leqslant (1 + \beta) OPT.$$

and the DP *runs in time* n^{UB}*, where* n *is the number of boxes and* $c_{\min}$ *is the minimum cost box.*

7 SAMPLE ACCESS TO DISTRIBUTION

In this chapter we are going to revisit our results of Chapters 5, 3 and 4 in the regime where the distributions are not fully known. In real life problems, it is usually the case that we do not fully know the process that generated our data, which implies that the distribution of the data is unknown to us. It is however the case that we have historical data or observations on our problem at hand. This historical data can be thought of as *samples*, from this unknown prior distribution.

We are showing how when competing with the Fully-Adaptive benchmark it is impossible to obtain any meaningful approximation, but when competing with the Partially-Adaptive we are able to extend all our previous results to still hold within $(1+\varepsilon)$ factor, when we gain $m = \text{poly}(n, 1/\epsilon, \log(1/\delta))$ samples.

Competing with the Fully-Adaptive Recall that the optimal solution for PANDORA'S Box is Fully-Adaptive strategy that chooses which box to query each time based on all the costs that have been observed so far. While these are the best strategies one could hope for, they are impossible to find or approximate with samples. For example, it could be the case that the cost in the first few boxes encode the location of a box of cost 0 while every other box has infinite cost. While the best option can be identified with just few queries, any reasonable approximation to the optimal cost would need to accurately learn this mapping. Learning such an arbitrary mapping however is impossible through samples, unless there is significant probability of seeing the exact same combination of costs. Recall from our results in Chapter 5 that for explicitly given distributions, this is not an issue. However as we also showed in that chapter, PANDORA'S Box becomes equivalent to another NP-Hard problem, Uniform Decision Tree, and it is unclear whether there is a constant approximation algorithm for it.

Related work The sampling regime for PANDORA'S BOX, Guo et al. (2021) showed that $\tilde{O}(n^3/\varepsilon^3)$ samples are enough to obtain a ε-additive approximation to the optimal, when the distributions are independent. This bound was later improved by Fu and Lin (2020) to $\tilde{O}(n^2/\varepsilon^2)$. In the related prophet inequalities literature, Azar et al. (2014) first studied the Single Sample Prophet Inequality problem where instead of full access to the prior distributions, we are only given a single sample from each. They gave a reduction to order-oblivious secretary policies. Later Rubinstein et al.

(2020) showed that even in this restrictive setting we can obtain the best possible ratio 2. In the IID setting of this problem, where all the priors are identical, Correa et al. (2019) showed that $e/(e-1)$ is achievable with one sample from each distribution, a bound later improved in Correa et al. (2020, 2024). Finally, Caramanis et al. (2022) address the same problem but under more complex combinatorial constraints.

7.1 Competing with the Partially-Adaptive

Since competing with the Fully-Adaptive is hopeless in the case of samples, in this section we revisit the results presented in Chapter 3 and 4 and show how **they all can be extended to the case where we only are given sample access to the prior distribution**.

7.1.1 Learning from samples: initial approach

Recall that in our initial approach of Chapter 3 we used a reduction to a smaller class of strategies called *Scenario-Aware Partially-Adaptive* (*SPA*). In these strategies the probing order σ is independent of the costs observed in probed boxes, however, the stopping time τ is a function of the instantiated scenario s. In other words, the algorithm fixes a probing order, then learns of the scenario instantiated, and then determines a stopping rule for the chosen probing order based on the revealed scenario.

Therefore, we focus on designing good scenario-aware partially adaptive strategies for Pandora's Box. Observe that once we fix a probing order, determining the optimal scenario-aware stopping time is easy. We now show that in order to optimize over all possible probing orders, it suffices to optimize with respect to a small set of scenarios drawn randomly from the underlying distribution.

Formally, let $\mathcal{D}$ denote the distribution over scenarios and let $\mathcal{S}$ be a collection of m scenarios drawn independently from $\mathcal{D}$, with m being a large enough polynomial in n. Then, we claim that with high probability, for *every* probing order σ, $\text{cost}_{\mathcal{D}}(\sigma)$ is close to $\text{cost}_{\mathcal{S}}(\sigma)$, where $\text{cost}_{\mathcal{D}}(\sigma)$ denotes the total expected cost of the SPA strategy σ over the scenario distribution $\mathcal{D}$, and $\text{cost}_{\mathcal{S}}(\sigma)$ denotes its cost over the uniform distribution over the sample $\mathcal{S}$. The implication is that it suffices for us to optimize for SPA strategies over scenario distributions with finite small support.

Lemma 7.1. *Let $\epsilon, \delta > 0$ be given parameters. Let $\mathcal{S}$ be a set of m scenarios chosen independently at random from $\mathcal{D}$ with $m = poly(n, 1/\epsilon, \log(1/\delta))$. Then, with probability at least $1-\delta$, for all permutations $\pi : [n] \to [n]$, we have*

$$\text{cost}_{\mathcal{S}}(\pi) \in (1 \pm \epsilon)\,\text{cost}_{\mathcal{D}}(\pi).$$

Proof. Fix a permutation π. For scenario s, let $\text{cost}_s(\pi) = \min_i\{i + c_{\pi(i)s}\}$ denote the total cost incurred by SPA strategy π in scenario s. Observe that for any π and any s, we have $\text{cost}_s(\pi) \in [1 + \min_i c_{is}, n + \min_i c_{is}]$. Furthermore, $\text{cost}_{\mathcal{D}}(\pi) = \mathbf{E}_{s\sim\mathcal{D}}[\text{cost}_s(\pi)]$, and $\text{cost}_{\mathcal{S}}(\pi) = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}} \text{cost}_s(\pi)$. The lemma now follows by using the Hoeffding inequality and applying the union bound over all possible permutations π. $\square$

Combining Corollary 3.3 and Lemma 7.1 yields the following theorem.

Theorem 7.2. *Suppose there exists an algorithm for the optimal search problem that runs in time polynomial in the number of boxes n and the number of scenarios m, and returns an SPA strategy achieving an α-approximation. Then, for any $\epsilon > 0$, there exists an algorithm that runs in time polynomial in n and $1/\epsilon$ and returns a PA strategy with competitive ratio $\frac{e}{e-1}(1+\epsilon)\alpha$, where $n = |\mathcal{B}|$.*

Recall that in Chapter 3 we gave competitive algorithms against the Scenario Aware strategies. Now we can, using Theorem 7.2 to extend all these results to the results shown in the following table.

	Approximation Factor
Choose 1 box	$(1+\varepsilon)9.22$ (Theorem 7.2+3.5)
Choose k boxes	$(1+\varepsilon)O(1)$ (Theorem 7.2+3.12)
Choose matroid basis	$(1+\varepsilon)O(\log k)$ (Theorem 7.2+3.15)

Table 7.1: Approximation factors from our initial approach, when selecting $\text{poly}(n, 1/\varepsilon, \log 1/\delta)$ samples. Results hold w.p. $1-\delta$.

7.1.1.1 Ski rental with general rent cost

We also note that even in the case of general rent costs, the results presented in Section 3.A.2.1 still hold via sampling polynomially many scenarios, i.e. Theorem 7.2. Recall that Theorem 7.2 requires two building blocks: Corollary 3.3 which describes a reduction from a general strategy to a scenario-aware strategy and Lemma 7.1 that guarantees that a small sample over scenarios suffices to achieve a good approximation.

We proved Lemma 7.1 by observing that for any probing order π, the cost of any scenario s is bounded in a polynomial range. This still holds since the total probing time is bounded by nP.

7.1.2 Learning from samples: simpler approach

Moving on to our simpler approach of Chapter 4, we show how the results extend to the sampling regime, but only for the Algorithm that updates the prior by conditioning on $V_b > \sigma_b$.

Conditioning on $V_b = v$ The second variant with full Bayesian updates $\mathcal{D}|_{V=v}$ requires full knowledge of the underlying distribution and can only work with sample access if one can learn the full distribution. To see this consider for example an instance where the values are drawn uniformly from $[0,1]^d$. No matter how many samples one draws, it is impossible to know the conditional distribution $\mathcal{D}|_{V=v}$ after opening the first box for fresh samples v, and the Bayesian update is not well defined[1].

Conditioning on $V_b > \sigma_b$ This variant does not face this problem and can be learned from samples if the costs of the boxes are polynomially bounded by n, i.e. if there is a constant $c > 0$ such that for all $b \in \mathcal{B}$, $c_b \in [1, n^c]$. If the weights are unbounded, it is impossible to get a good approximation with few samples. To see this consider the following instance. Box 1 has cost $1/H \to 0$, while every other box has cost H for a very large $H > 0$. Now consider a distribution where with probability $1 - \frac{1}{H} \to 1$, the value in the first box is 0, and with probability $1/H$ is $+\infty$. In this case, with a small number of samples we never observe any scenario where $v_1 \neq 0$ and believe the overall cost is near 0. However, the true cost is at least $H \cdot 1/H \geqslant$ and is determined by how the order of boxes is chosen when the scenario has $v_1 \neq 0$. Without any such samples it is impossible to pick a good order.

Therefore, we proceed to analyze Variant 1 with $\mathcal{D}|_{V>\sigma}$ in the case when the box costs are similar. We show that polynomial, in the number of boxes, samples suffice to obtain an approximately-optimal algorithm, as we formally state in the following

[1]For a discrete distribution example see Section 7.A.1 of this chapter's appendix.

theorem. We present the case where all boxes have cost 1 but the case where the costs are polynomially bounded easily follows.

Theorem 7.3. *Consider an instance of* PANDORA*'s Box with opening costs equal to* 1*. For any given parameters* $\varepsilon, \delta > 0$*, using* $m = \text{poly}(n, 1/\varepsilon, \log(1/\delta))$ *samples from* $\mathcal{D}$*, Algorithm* 6 (*Variant* 1) *obtains a* $4.428 + \varepsilon$ *approximation policy against the partially-adaptive optimal, with probability at least* $1 - \delta$*.*

To prove the theorem, we first note that variant 1 of Algorithm 6 takes a surprisingly simple form, which we call a threshold policy. It can be described by a permutation π of visiting the boxes and a vector of thresholds $\boldsymbol{\tau}$ that indicate when to stop. The threshold for every box corresponds to the reservation value the first time the box is opened. To analyze the sample complexity of Algorithm 6, we study a broader class of algorithms parameterized by a permutation and vector of thresholds given in Algorithm 12.

Algorithm 12: General format of PANDORA's Box algorithm.

```
  Input: Set of boxes, permutation $\pi$, vector of thresholds $\boldsymbol{\tau} \in \mathbb{R}^n$
1 best $\leftarrow \infty$
2 foreach $i \in [n]$ do
3     if best $> \tau_i$ then
4         Open box $\pi_i$, see value $v_i$
5         best $\leftarrow \min(\text{best}, v_i)$
6     else
7         Accept best
8 end
```

Our goal now is to show that polynomially many samples from the distribution $\mathcal{D}$ suffice to learn good parameters for Algorithm 12. We first show a Lemma that bounds the cost of the algorithm calculated in the empirical $\hat{\mathcal{D}}$ instead of the original $\mathcal{D}$ (Lemma 7.4), and a Lemma 7.5 that shows how capping the reservation values by n/ε can also be done with negligible cost.

Lemma 7.4. *Let* $\varepsilon, \delta > 0$ *and let* $\mathcal{D}'$ *be the empirical distribution obtained from poly*$(n, 1/\varepsilon,$ $\log(1/\delta))$ *samples from* $\mathcal{D}$*. Then, with probability* $1 - \delta$*, it holds that*

$$\left| \boldsymbol{E}_{\hat{\mathcal{D}}} \left[ALG(\pi, \tau) - \min_{b \in \mathcal{B}} v_b \right] - \boldsymbol{E}_{\mathcal{D}} \left[ALG(\pi, \tau) - \min_{b \in \mathcal{B}} v_b \right] \right| \leqslant \varepsilon$$

for any permutation π and any vector of thresholds $\boldsymbol{v} \in \left[0, \frac{n}{\varepsilon}\right]^n$

We defer the proof of Lemmas 7.4, 7.5 and that of Theorem 7.3 to Section 7.A.1 of the Appendix.

Lemma 7.5. *Let $\mathcal{D}$ be any distribution of values. Let $\varepsilon > 0$ and consider a permutation π and thresholds $\boldsymbol{\tau}$. Moreover, let τ' be the thresholds capped to n/ε, i.e. setting $\tau'_b = \min\{\tau_b, n/\varepsilon\}$ for all boxes b. Then,*

$$\boldsymbol{E}_{v\sim D}\left[ALG(\pi, \tau')\right] \leqslant (1+\varepsilon)\boldsymbol{E}_{v\sim D}\left[ALG(\pi, \tau)\right].$$

Note on Continuous vs Discrete Distributions. The results of Section 7.1.2 apply for general distributions (discrete or continuous) and show that the partial updates variant leads to good approximation when run on the empirical distribution obtained just with polynomially many samples. In contrast, the full updates variant requires a complete description of the distribution. However, as the approximation factor does not depend on the support size, It can also apply even for continuous distributions with arbitrary large support by taking a limit over a very fine discretization

7.A Appendix for Chapter 7

7.A.1 Proofs from Section 7.1.2

We first present an example of a discrete distribution that shows that one needs exponentially many samples in the number of boxes to learn $D_{V=v}$.

Discrete Distribution Example Consider a distribution that only takes values $0, H, H+1$ for some very large $H > 0$. The scenario is drawn by choosing a random bit $b_i \in \{0, 1\}$ for every box and depending on the realized sequence $\mathbf{b}$ a single box $f(\mathbf{b}) \in [n]$ is chosen for an unknown and arbitrary function f. The value at box i is then chosen to be $H + b_i$ unless i is the box $f(\mathbf{b})$ in which case it is 0. In this case learning the probability $D_{V=v}$ would require learning the unknown function f on all inputs which are exponentially many. In particular, if we only take $s << 2^n$ samples, for any order of choosing boxes after $\approx \log s$ steps, none of the samples in our collection will match the observed sequence of bits, therefore it will not be possible to compute a posterior distribution.

We continue by giving the omitted proofs.

Lemma 7.4. *Let $\varepsilon, \delta > 0$ and let $\mathcal{D}'$ be the empirical distribution obtained from poly$(n, 1/\varepsilon,$ $\log(1/\delta))$ samples from $\mathcal{D}$. Then, with probability $1-\delta$, it holds that*

$$\left| \boldsymbol{E}_{\hat{D}} \left[ALG(\pi, \tau) - \min_{b \in \mathcal{B}} \nu_b \right] - \boldsymbol{E}_{D} \left[ALG(\pi, \tau) - \min_{b \in \mathcal{B}} \nu_b \right] \right| \leqslant \varepsilon$$

for any permutation π and any vector of thresholds $\boldsymbol{\nu} \in \left[0, \frac{n}{\varepsilon}\right]^n$

Proof of Lemma 7.4. We first argue that we can accurately estimate the cost for any vector of thresholds $\boldsymbol{\tau}$ when the order of visiting boxes is fixed.

Consider any fixed permutation $\pi = \pi_1, \pi_2, \ldots, \pi_n$ be any permutation of the boxes, we relabel the boxes without loss of generality so that π_i is box i.

Denote by $\hat{V}_i = \min_{j \leqslant i} \nu_j$, and observe that $\hat{V}_i$ is a random variable that depends on the distribution $\mathcal{D}$. Then we can write the expected cost of the algorithm as the expected sum of the opening cost and the chosen value: $\mathbf{E}_{\mathcal{D}}[\text{ALG}] = \mathbf{E}_{\mathcal{D}}[\text{ALG}_o] + \mathbf{E}_{\mathcal{D}}[\text{ALG}_\nu]$. We have that:

$$\mathbf{E}_{\mathcal{D}}[\text{ALG}_o] = \sum_{i=1}^{n} \mathbf{Pr}_{\mathcal{D}}[\text{reach } i] = \sum_{i=1}^{n} \mathbf{Pr}_{\mathcal{D}} \left[\bigwedge_{j=1}^{i-1} (\hat{V}_j > \tau_{j+1}) \right]$$

Moreover, we denote by $\overline{V}_{\boldsymbol{\tau}}^i = \bigwedge_{j=1}^{i-1} \left(\hat{V}_j > \tau_{j+1} \right)$ and we have

$$\begin{aligned}
\mathbf{E}_{\mathcal{D}} \left[\text{ALG}_\nu - \hat{V}_n \right] &= \sum_{i=1}^{n} \mathbf{E}_{\mathcal{D}} \left[(\hat{V}_i - \hat{V}_n) \cdot \mathbb{1}\{\text{stop at } i\} \right] \\
&= \sum_{i=1}^{n-1} \mathbf{E}_{\mathcal{D}} \left[(\hat{V}_i - \hat{V}_n) \cdot \mathbb{1}\left\{ \overline{V}_{\boldsymbol{\tau}}^i \wedge \left(\hat{V}_i \leqslant \tau_{i+1} \right) \right\} \right] \\
&= \sum_{i=1}^{n-1} \mathbb{E}_{\mathcal{D}} \left[\tau_{i+1} \mathbf{Pr}_{r \sim U[0, \tau_{i+1}]} \left[r < \hat{V}_i - \hat{V}_n \right] \cdot \mathbb{1}\left\{ \overline{V}_{\boldsymbol{\tau}}^i \wedge \left(\hat{V}_i \leqslant \tau_{i+1} \right) \right\} \right] \\
&= \sum_{i=1}^{n-1} \tau_{i+1} \mathbf{Pr}_{\mathcal{D}, r \sim U[0, \tau_{i+1}]} \left[\overline{V}_{\boldsymbol{\tau}}^i \wedge \left(r + \hat{V}_n \leqslant \hat{V}_i \leqslant \tau_{i+1} \right) \right]
\end{aligned}$$

In order to show our result, we use from Blumer et al. (1989) that for a class with VC dimension $d < \infty$ that we can learn it with error at most ε with probability $1-\delta$ using $m = \text{poly}(1/\varepsilon, d, \log(1/\delta))$ samples.

Consider the class $\mathcal{F}_{\boldsymbol{\tau}}(\hat{V}, r) = \bigwedge_{j=1}^{i-1}(\hat{V}_j > \tau_{j+1})$. This defines an axis parallel rectangle in $\mathbb{R}^i$, therefore its VC-dimension is $2i$. Using the observation above we have that using $m = \text{poly}(1/\varepsilon, n, \log(1/\delta))$ samples, , with probability at least $1-\delta$, it holds

$$\left| \mathbf{Pr}_{\mathcal{D}} \left[\mathcal{F}_{\boldsymbol{\tau}}(\hat{V}, r) \right] - \mathbf{Pr}_{\hat{\mathcal{D}}} \left[\mathcal{F}_{\boldsymbol{\tau}}(\hat{V}, r) \right] \right| \leqslant \varepsilon$$

for all $\boldsymbol{\tau} \in \mathbb{R}^n$. Similarly, the class

$$\mathcal{C}_{\boldsymbol{\tau}}(\hat{V}, r) = \bigwedge_{j=1}^{i-1} \left(\hat{V}_j > \tau_{j+1} \right) \wedge \left(r + \hat{V}_n \leqslant \hat{V}_i \leqslant \tau_{i+1} \right)$$

has VC-dimension $O(n)$ since it is an intersection of at most n (sparse) halfspaces. Therefore, the same argument as before applies and for $m = \text{poly}(1/\varepsilon, n, \log(1/\delta))$ samples, we get

$$\left| \mathbf{Pr}_{\mathcal{D}, r \sim U[0, \tau_{i+1}]} \left[\mathcal{C}_{\boldsymbol{\tau}}(\hat{V}, r) \right] - \mathbf{Pr}_{\hat{\mathcal{D}}, r \sim U[0, \tau_{i+1}]} \left[\mathcal{C}_{\boldsymbol{\tau}}(\hat{V}, r) \right] \right| \leqslant \varepsilon$$

for all $\boldsymbol{\tau} \in \mathbb{R}^n$, with probability at least $1-\delta$.

Putting it all together, the error can still be unbounded if the thresholds τ are too large. However, since we assume that $\tau_i \leqslant n/\varepsilon$ for all $i \in [n]$, $\text{poly}(n, 1/\varepsilon, \log(1/\delta))$ samples suffice to get ε error overall, by setting $\varepsilon \leftarrow \frac{\varepsilon^2}{n}$.

While we obtain the result for a fixed permutation, we can directly obtain the result for all $n!$ permutations through a union bound. Setting $\delta \leftarrow \frac{\delta}{n!}$ only introduces an additional factor of $\log(n!) = n \log n$ in the overall sample complexity. $\square$

Lemma 7.5. *Let $\mathcal{D}$ be any distribution of values. Let $\varepsilon > 0$ and consider a permutation π and thresholds $\boldsymbol{\tau}$. Moreover, let τ' be the thresholds capped to n/ε, i.e. setting $\tau'_b = \min\{\tau_b, n/\varepsilon\}$ for all boxes b. Then,*

$$\boldsymbol{E}_{v \sim D}\left[ALG(\pi, \tau') \right] \leqslant (1+\varepsilon) \boldsymbol{E}_{v \sim D}\left[ALG(\pi, \tau) \right].$$

Proof of Lemma 7.5. We compare the expected cost of ALG with the original thresholds and the transformed one ALG′ with the capped thresholds. For any value vector $\boldsymbol{v} \sim \mathcal{D}$, either (1) the algorithms stopped at the same point having the same opening cost and value, or (2) ALG stopped earlier at a threshold $\tau > n/\varepsilon$, while ALG′

continued. In the latter case, the value v that ALG gets is greater than n/ε, while the value v' that ALG$'$ gets is smaller, $v' \leqslant v$. For such a scenario, the opening cost c of ALG, and the opening cost c' of ALG$'$ satisfy $c' \leqslant c + n$. Thus, the total cost is $c' + v' \leqslant c + v + n \leqslant (1+\varepsilon)(c+v)$, where the last inequality follows from $u > n/\varepsilon$. Overall, we get that

$$\mathbf{E}_{\mathcal{D}}\left[\text{ALG}'\right] \leqslant \mathbf{E}_{\mathcal{D}}\left[\text{ALG}\right](1+\varepsilon).$$

□

Theorem 7.3. *Consider an instance of* Pandora*'s Box with opening costs equal to 1. For any given parameters* $\varepsilon, \delta > 0$*, using* $m = \text{poly}(n, 1/\varepsilon, \log(1/\delta))$ *samples from* $\mathcal{D}$*, Algorithm 6 (Variant 1) obtains a* $4.428 + \varepsilon$ *approximation policy against the partially-adaptive optimal, with probability at least* $1 - \delta$*.*

Proof of Theorem 7.3. With $\text{poly}(n, \varepsilon, \log(1/\delta))$ samples from $\mathcal{D}$, we obtain an empirical distribution $\hat{\mathcal{D}}$.

From Lemma 7.4, we have that with probability at least $1 - \delta\varepsilon/\log(1/\delta)$, the following holds

$$\left| \mathbf{E}_{v\sim\hat{D}}\left[\text{ALG}(\pi,\tau) - \min_{b\in\mathcal{B}} v_b\right] - \mathbf{E}_{v\sim D}\left[\text{ALG}(\pi,\tau) - \min_{b\in\mathcal{B}} v_b\right] \right| \leqslant \varepsilon \tag{7.1}$$

for any permutation π and any vector of thresholds $\boldsymbol{v} \in \left[0, \frac{n}{\varepsilon}\right]^n$. This gives us that we can estimate the cost of a threshold policy accurately.

To compare with the set of all partially adaptive policies that may not take the form of a threshold policy, we consider the set of scenario aware policies (SA). These are policies SA(π) parameterized by a permutation π of boxes and are forced to visit the boxes in that order. However, they are aware of all values in the boxes in advance and know precisely when to stop. These are unrealistic policies introduced in Chawla et al. (2020) which serve as an upper bound to the set of all partially adaptive policies.

As shown in Chawla et al. (2020) (Lemma 3.3), scenario-aware policies are also learnable from samples. With probability at least $1 - \delta\varepsilon/\log(1/\delta)$, it holds that for any permutation π

$$\left| \mathbf{E}_{v\sim\hat{D}}\left[SA(\pi) - \min_{b\in\mathcal{B}} v_b\right] - \mathbf{E}_{v\sim D}\left[SA(\pi) - \min_{b\in\mathcal{B}} v_b\right] \right| \leqslant \varepsilon. \tag{7.2}$$

The α-approximation guarantees (with $\alpha \approx 4.428$) of Algorithm 6 hold even against scenario aware policies as there is no restriction on how the partially-adaptive policy may choose to stop. So for the empirical distribution, we can compute a permutation $\hat{\pi}$ and thresholds $\hat{\tau}$ such that:

$$\mathbf{E}_{\hat{D}}\left[\mathrm{ALG}(\hat{\pi},\hat{\tau})\right] \leqslant \alpha \cdot \min_{\pi} \mathbf{E}_{\hat{D}}\left[SA(\pi)\right]$$

Clipping the thresholds to obtain $\hat{\tau}' = \min\{\hat{\tau}, n/\varepsilon\}$, and letting $\Delta = \mathbf{E}_{\nu\sim\hat{D}}\left[\min_{b\in\mathcal{B}} \nu_b\right] - \mathbf{E}_{\nu\sim D}\left[\min_{b\in\mathcal{B}} \nu_b\right]$, we have that:

$$\begin{aligned}
\mathbf{E}_{D}\left[\mathrm{ALG}(\hat{\pi},\hat{\tau}')\right] &\leqslant \mathbf{E}_{\hat{D}}\left[\mathrm{ALG}(\hat{\pi},\hat{\tau}')\right] - \Delta + \varepsilon \\
&\leqslant (1+\varepsilon)\mathbf{E}_{\hat{D}}\left[\mathrm{ALG}(\hat{\pi},\hat{\tau})\right] + \Delta + \varepsilon/4 \\
&\leqslant (1+\varepsilon)\alpha \cdot \min_{\pi} \mathbf{E}_{\hat{D}}\left[SA(\pi)\right] - \Delta + \varepsilon/4 \\
&\leqslant (1+\varepsilon)\alpha \cdot \min_{\pi} \mathbf{E}_{D}\left[SA(\pi)\right] + O(\Delta+\varepsilon)
\end{aligned}$$

By Markov's inequality, we have that

$$\mathbf{Pr}\left[\mathbf{E}_{\nu\sim\hat{D}}\left[\min_{b\in\mathcal{B}} \nu_b\right] \leqslant (1+\varepsilon)\mathbf{E}_{\nu\sim D}\left[\min_{b\in\mathcal{B}} \nu_b\right]\right] \geqslant \frac{\varepsilon}{1+\varepsilon} \geqslant \varepsilon/2.$$

Thus, repeating the sampling process $\frac{O(\log 1/\delta)}{\varepsilon}$ times and picking the empirical distribution with minimum $\mathbf{E}_{\nu\sim\hat{D}}\left[\min_{b\in\mathcal{B}} \nu_b\right]$ satisfies

$$\Delta \leqslant \varepsilon \mathbf{E}_{\nu\sim D}\left[\min_{b\in\mathcal{B}} \nu_b\right]$$

with probability at least $1-\delta$ and simultaneously satisfies equations (7.1) and (7.2).

This shows that $\mathbf{E}_{D}\left[\mathrm{ALG}(\hat{\pi},\hat{\tau}')\right] \leqslant (1+O(\varepsilon))\alpha \cdot \min_{\pi} \mathbf{E}_{D}\left[SA(\pi)\right]$ which completes the proof by rescaling ε by a constant.

□

8 UNKNOWN DISTRIBUTIONS: AN ONLINE PANDORA'S BOX PROBLEM

As we established, PANDORA's Box is a fundamental stochastic optimization framework, which models the trade-off between exploring a set of alternatives and exploiting the already collected information, in environments where data acquisition comes at a cost.

The model captures a variety of different settings where the decision-maker needs to balance the value of the selected alternative and the effort devoted to find it. We include some examples below.

- Consider an online shopping environment where a search engine needs to present users with results on a product they want to buy. Visiting all potential e-shops that sell this product to find the cheapest option would be prohibitive in terms of time needed to present the search results to the user. The search engine needs to explore the different options only up to the extent that would make the marginal improvement in the best price found worthwhile.
- Consider a path planning service provider like Google Maps. Upon request, the provider must search its database for a good path to recommend to the user, but the higher the time spent searching the higher is the server cost. The provider must trade off computation cost with the quality of the result.

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 our examples given above, it is always the case that we have to repeatedly solve these problems giving rise to the question "**Can we use past data to guide our future decisions, even when no information is given to us a priori?**". In the two sections of this chapter we give positive answers to this question by designing no-regret algorithms both for the case of adversarial distributions, and in a contextual setting, where the distributions are independent and unknown, but we have an extra information at each round.

Related work The closest related work to our problem is the paper by Gatmiry et al. (2024) that studies Prophet Inequalities and Pandora's Box problems in an online learning setting, where however, the distributions are independent. Similarly, in a distributional learning setting Guo et al. (2021) showed that $\tilde{O}(n^3/\varepsilon^3)$ samples are enough to obtain a ε-additive approximation to the optimal, when the distributions are independent. This bound was later improved by Fu and Lin (2020) to $\tilde{O}(n^2/\varepsilon^2)$.

Our worst case setting of Section 8.1 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 and Singer (2007a).

Our contextual setting of section 8.2 is closely related to contextual multi-armed bandits, where the contexts provide additional information on the quality of the actions at each round. In particular, in the case of *stochastic linear bandits* (Abe and Long, 1999): the reward of each round is given by a (noisy) a linear function of the context drawn at each round. Optimistic algorithms proposed for this setting rely on maintaining a confidence ellipsoid for estimating the unknown vector (Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Abbasi-Yadkori et al., 2011; Valko et al., 2014). On the other hand, in *adversarial linear bandits*, a context vector is adversarially selected at each round. The loss is characterized by the inner product of the context and the selected action of the round. Common approaches for this setting include variants of the multiplicative-weights algorithm (Hazan et al., 2014; van der Hoeven et al., 2018), as well as, tools from online linear optimization (Blair, 1985; Cesa-Bianchi and Lugosi, 2006) such as Follow The Regularized Leader (FTRL) and mirror descent (see (Bubeck and Eldan, 2015; Abernethy et al., 2008; Shalev-Shwartz and Singer, 2007b; Bubeck et al., 2018) and references therein).

This setting also generalizes the contextual bandits setting, since any instance of contextual bandits can be reduced to Contextual Pandora's Box for box costs selected to be large enough. One work from the contextual bandits literature that is more closely related to ours is the recent work of Foster and Rakhlin (2020). Similarly to our work, they provide a generic reduction from contextual multi-armed bandits to online regression, by showing that any oracle for online regression can be used to obtain a contextual bandits algorithm.

8.1 Worst Case distributions

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 multiplicative 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 8.1.3) 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 8.1.4). 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 8.A.1).

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.

8.1.1 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 and Kertz (1992); Lucier (2017); Correa et al. (2018) for more details on prophet inequalities).

8.1.2 Definitions & Notation

Approximate Regret In this first part, 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

$$\text{Regret}_{\text{OPT}}(\mathcal{A}, T) = \frac{1}{T}\sum_{t=1}^{T}(\mathcal{A}(t) - \text{OPT}(t)) \tag{8.1}$$

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

$$\alpha\text{-Regret}_{\text{OPT}}(\mathcal{A}, T) = \frac{1}{T}\sum_{t=1}^{T}(\mathcal{A}(t) - \alpha\text{OPT}(t)) \,. \tag{8.2}$$

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.

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

8.1.2.1 Relaxations

We are again using the scenario-aware relaxation of Chapter 3, and the fact that it is enough to design a strategy that chooses an ordering of the boxes and performs well, assuming that we know when to stop. We restate the Theorem that proved this fact below for convenience.

Lemma 8.1 (Simplification of Theorem 7.2 from 3)**.** *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.*

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 8.2 (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 $x \in X$, $\overline{g}(x) \leqslant g(x)$.*

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 and 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^s_{it} the indicator of whether box i is selected for scenario s at time t. We define the relaxation $\overline{g}^s(\boldsymbol{x})$ as

$$\begin{aligned} \min_{z\geqslant 0} \quad & \sum_{i\in\mathcal{B}, t\in\mathcal{T}} (t + c^s_i) z^s_{it} && \text{(Relaxation-SPA)} \\ \text{s.t.} \quad & \sum_{t\in\mathcal{T}, i\in\mathcal{B}} z^s_{it} = 1, \end{aligned}$$

$$z^s_{it} \leqslant x_{it}, \qquad\qquad i \in \mathcal{B}, t \in \mathcal{T}.$$

Similarly, we also relax the problem when we are required to pick k boxes (LP-k-cover) and when we are required to pick a matroid basis (LP-matroid). Leveraging our previous results (Chawla et al. (2020)), presented in sections 3.2, 3.3.2 and 3.3.1 of Chapter 3, 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 8.3 (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*

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

8.1.3 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 8.4. *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 αRegret(T) for Π.

Proof of Theorem 8.4. 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

$$\begin{aligned}\mathbf{E}\left[\sum_{t=1}^{T} f_t\big(\mathcal{A}(\overline{x_t})\big)\right] &\leqslant \alpha \sum_{t=1}^{T} \overline{f}_t(\overline{x}_t) \\ &\leqslant \alpha \left(\text{Regret}(T) + \min_{x \in \overline{X}} \sum_{t=1}^{T} \overline{f}_t(x)\right) \\ &\leqslant \alpha \left(\text{Regret}(T) + \min_{x \in X} \sum_{t=1}^{T} f_t(x)\right).\end{aligned}$$

By rearranging the terms, we get the theorem. □

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 8.6) and (2) how to obtain rounding algorithms with good approximation guarantees, using the results of Chawla et al. (2020).

8.1.3.1 Applications to Pandora's Box and MSSC

Applying Theorem 8.4 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, LP-k-cover and LP-matroid.

We denote by $\overline{g}^s(\mathbf{x})$ 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 $\mathbf{x}_t \in \overline{X}$, 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 loss $f^s(\mathbf{x}_t) = \overline{g}^s(\mathbf{x}_t)$. Note that f^s is convex in all cases (Relaxation-SPA, LP-k-cover, LP-matroid).
3. We observe the function f^s for all points $\mathbf{x} \in \overline{X}$.

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 8.5 (Theorem 2.11 from Shalev-Shwartz (2012))**.** *Let $f_1, \dots, f_T$ 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$(FTRL, T) \cdot T \leqslant U_{\max} - U_{\min} + TL^2\eta$*

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 13, and we show how to choose the regularizer $U(\boldsymbol{x})$ in Theorem 8.6.

Algorithm 13: Algorithm $\mathcal{A}$ for the full information case.

```
  Input: $\Pi = (\mathcal{F}, OPT)$ : the problem to solve, $\mathcal{A}_\Pi$ : the rounding algorithm for $\Pi$
1 Denote by $f^s(\boldsymbol{x})$ = fractional objective function
2 Select regularizer $U(\boldsymbol{x})$ according to Theorem 8.6
3 $\overline{X}$ = space of fractional solutions
4 for Each round $t \in [T]$ do
5     Set $\boldsymbol{x}_t = \min_{x \in \overline{X}} \sum_{\tau=1}^{t-1} f^{s_\tau}(\boldsymbol{x}) + U(\boldsymbol{x})$
6     Round $\boldsymbol{x}_t$ to $\boldsymbol{x}_t^{int}$ according to $\mathcal{A}_\Pi$
7     Receive loss $f^s(\boldsymbol{x}_t^{int})$
8 end
```

We show the guarantees of our algorithm above using Theorem 8.5 which provides regret guarantees for *FTRL*. The proof of Theorem 8.6 is deferred to section 8.A.6 of the appendix.

Theorem 8.6. *The average regret of Algorithm 13 is*

$$\textit{Regret}_{PA}(\mathcal{A}, T) \leqslant 2n\sqrt{\frac{\log n}{T}}$$

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 8.4 we get Corollary 8.7 for competing with the partially-adaptive benchmark for all different feasibility constraints (choose 1, choose k or choose a matroid basis).

Corollary 8.7 (Competing against PA, full information)**.** *In the full information setting, Algorithm 13 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 8.8. *In the special case of MSSC, our approach obtains the tight* 4*-approximation of the offline case Feige et al.* (*2004*)*. This result improves on the previous work Fotakis et al.* (*2020*) *who obtain a* 11.713*-approximation.*

8.1.4 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 $x_t \in \overline{X}$ in each round t, we only observe a loss $f^s(x_t)$ at the point x_t we chose to play and not for every $x \in \overline{X}$. This difference prevents us from directly using any online convex optimization algorithm, as in the full information setting of section 8.2.6.1. However, observe that if we decide to open all n boxes, this is equivalent to observing the function f^s for all $x \in \overline{X}$, 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 14, and we show the following guarantees.

Theorem 8.9. *The average regret for Algorithm 14, for* $k = \left(\frac{n}{2L+\sqrt{\log n}}\right)^{2/3} T^{1/3}$ *and loss functions that are* L*-Lipschitz is*

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

To analyze the regret of Algorithm 14 and prove Theorem 8.9, we consider the regret of two related settings.

Algorithm 14: $\mathcal{A}_{PA}$ minimizing regret against PA

```
Get parameter $k$ from Theorem 8.9
Select regularizer $U(\boldsymbol{x})$ according to Theorem 8.9
Split the times $[T]$ into $T/k$ intervals $\mathcal{I}_1 \ldots, \mathcal{I}_{T/k}$
$\mathcal{R} \leftarrow \emptyset$ // Random times for each $\mathcal{I}_i$
for Every interval $\mathcal{I}_i$ do
    Pick a $t_p$ uniformly in $\mathcal{I}_i$
    for All times $t \in \mathcal{I}_i$ do
        if $t = t_p$ then
            $\mathcal{R} \leftarrow \mathcal{R} \cup \{t_p\}$
            Open all boxes
            Get feedback $f^{s_{t_p}}$
        else
            $\boldsymbol{x}_t \leftarrow \text{argmin}_{x \in \overline{X}} \sum_{\tau \in \mathcal{R}} f^{s_\tau}(\boldsymbol{x}) + U(\boldsymbol{x})$
        end
    end
end
```

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.*

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 8.10. *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 8.10. 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 $x^* = \text{argmin}_{x \in \overline{X}} \sum_{t=1}^{T/k} \overline{f}_t(x)$ 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

$$\begin{aligned}
\mathbf{E}\left[\sum_{t=1}^{T/k}|X_t|\right] &\leqslant \left(\mathbf{E}\left[\left(\sum_{t=1}^{T/k}X_t\right)^2\right]\right)^{1/2} \\
&= \left(\mathbf{E}\left[\sum_{t=1}^{T/k}X_t^2\right]\right)^{1/2} \\
&\leqslant n\sqrt{\frac{T}{k}}.
\end{aligned}$$

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. $\mathbf{E}_{X_t|X_1,\dots,X_{t-1}}[=]\,0$, as the choice of the function at time t is independent of the chosen point $\boldsymbol{x}_t$.

We now look at the average regret of the strategy $\boldsymbol{x}_t$ for the average cost setting. We have that

$$\begin{aligned}
\frac{1}{T}\mathbf{E}\left[\sum_t \bar{f}_t(\boldsymbol{x}_t)\right] - R(T) - \frac{n}{\sqrt{kT}} &\leqslant \frac{1}{T}\mathbf{E}\left[\sum_t f_t^r(\boldsymbol{x}_t)\right] - R(T) \\
&\leqslant \frac{1}{T}\mathbf{E}\left[\min_{\boldsymbol{x}}\sum_t f_t^r(\boldsymbol{x})\right] \\
&\leqslant \frac{1}{T}\mathbf{E}\left[\sum_t f_t^r(\boldsymbol{x}^*)\right] \\
&= \sum_t \bar{f}_t(\boldsymbol{x}^*)
\end{aligned}$$

which implies the required regret bound.

□

Given this lemma, we are now ready to show Theorem 8.9.

Proof of Theorem 8.9. 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 8.10, it suffices to bound the regret in the random costs setting. Let $U(\boldsymbol{x}) : [0,1]^{n\times n} \to \mathbb{R}$ be an η/n-strongly convex regularizer used in the *FTRL* algorithm.

We are using $U(\boldsymbol{x}) = \left(\sum_{i=1}^{n}\sum_{t=1}^{n} x_{it}\log x_{it}\right)/\eta$, which is η/n-strongly convex from Lemma 8.37 and is at most $(n\log n)/\eta$ as we observed in corollary 8.7. Then from Theorem 8.5, we get that the average regret for the corresponding random costs setting is $2L\sqrt{\frac{\log n}{kT}}$.

Using Lemma 8.10, we get that the total average regret $R(T)$ of our algorithm is

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

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

Finally, using Theorem 8.9 we can get the same guarantees as the full-information setting, using the α-approximate rounding for each case.

Corollary 8.11 (Competing against PA, bandit)**.** *In the bandit setting, Algorithm 14 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***

8.2 Adding Context

Moving on to our Contextual Pandora's Box setting, recall from the previous chapters, that despite the richness of the setting, Weitzman shows that the optimal policy for any instance of Pandora's Box admits a particularly simple characterization: for each box, one can compute a *reservation value* as a function of its cost and value distribution. Then, the boxes are inspected in increasing order of these values, until a simple termination criterion is fulfilled. This characterization is revealing: obtaining an optimal algorithm for Pandora's Box does not require complete knowledge of the distributions or the costs, yet only access to a *single statistic* for each box.

This raises the important question of how easy it is to learn a near-optimal search strategy, especially in environments where the distributions may not remain fixed across time but can change according to the characteristics of the instance at hand. In the case of online shopping, depending on the type of product we are searching, the e-shops have different product-specific distributions on the prices. We would be interested in a searching strategy that is not tied to a specific product, but is able to minimize the expected cost for any product we may be interested in. Similarly, in the case of path recommendations, the optimal search strategy may depend on the time of day or the day of the year. Motivated by the above question, we extend the PANDORA'S Box model to a contextual online learning setting, where the learner faces a new instance of the problem at each round. At the beginning of each round, the learner observes a context and must choose a search strategy for opening boxes depending on the observation. While the context and the associated opening costs of the boxes of each round are observed, the learner has no access to the value distributions, which may be arbitrary in each round.

Realizability. In the above description, the context may be irrelevant to the realized values. Such an adversarial setting is impossible to solve as it is related to the problem of learning thresholds online. In fact, even in the offline version of the problem, the task would still be computationally hard as it corresponds to agnostic learning (see Section 8.A.7.2 for more details). This naturally raises the following question: *What are the least possible strong assumptions, under which the problem becomes tractable?*

One of the main contributions of this paper is identifying a minimal realizability assumption under which the problem remains tractable. This assumption is parallel to the realizability assumption in contextual bandits (see chapter 19 of (Lattimore and Szepesvári, 2020)). We describe below how the difficulty of problem increases as we move from the strongest (1) to the weakest (4) assumptions possible:

1. **Contexts directly related to values**: in this case any learning algorithm is able to fit the contexts and predict exactly the realized values of the boxes. Such a setting is trivial yet unrealistic.

2. **Contexts directly related to distributions**: this is the case where there exists a learnable mapping from the context to the distribution of values. This is a more realistic setting, but it is still relatively constrained; it requires being able to perfectly determine the distribution family, which would need to be parametric.

3. **Contexts related to sufficient statistic**: in the more general case, instead of the whole value distributions, the contexts give us information about only a sufficient statistic of the problem. This is one of the main contribution of this work; we show that the problem remains tractable when the contexts give us

information on the *reservation values* of the boxes. Observe that in this case, the value distributions on the boxes can be arbitrarily different at each round, as long as they "implement" the correct reservation value based on the context. This model naturally extends the standard realizability assumption made in bandit settings, according to which, the mean of the distributions is predictable from the context. In that case, the sufficient statistic needed in order to select good arms is, indeed, the mean reward of each arm (**?**).

4. **No assumptions**: in this case, as explained before, the problem becomes intractable (see Section 8.A.7.2).

8.2.1 Our Contribution

We introduce a novel contextual online learning variant of the PANDORA's BOX problem, namely the CONTEXTUAL PANDORA's BOX, that captures the problem of learning near-optimal search strategies in a variety of settings.

Our main technical result shows that even when the sequence of contexts and distributions is adversarial, we can find a search strategy with sublinear average regret (compared to an optimal one) as long as no-regret algorithms exist for a much simpler online regression problem with a linear-quadratic loss function.

Main Theorem (Informal)**.** *Given an oracle that achieves expected regret* $r(T)$ *after* T *rounds for Linear-Quadratic Online Regression, there is an algorithm that obtains* $O(\sqrt{Tr(T)})$ *regret for the* CONTEXTUAL PANDORA's BOX *problem.*

The main technical challenge in obtaining the result is that the class of search strategies can be very rich. Even restricting to greedy policies based on reservation values for each box, the cost of the policies is a non-convex function of the reservation values. We manage to overcome this issue by considering a "proxy" function for the expected cost of the search policy which bounds the difference from the optimal cost, based on a novel sensitivity analysis of the original Weitzman's algorithm. The proxy function has a simple linear-quadratic form and thus optimizing it reduces our setting to an instance of *linear-quadratic online regression*. This allows us to leverage existing methods for minimizing regret in online regression problems in a black box manner.

Using the above reduction, we design algorithms with sublinear regret guarantees for two different variants of our problem: the *full information*, where the decision-maker observes the realized values of all boxes at the end of each round, and the *bandit* version, where only the realized values of the opened boxes can be observed. We achieve both results by constructing oracles based on the *Follow the Regularized Leader* family of algorithms.

Beyond the results shown in this paper, an important conceptual contribution of our work is extending the traditional bandit model in the context of stochastic optimization. Instead of trying to learn simple decision rules, in stochastic optimization we are interested in learning complex algorithms tailored to a distribution. Our model can be extended to a variety of such problems beyond the Pandora's Box setting: one concrete such example is the case of designing revenue optimal auctions for selling a single item to multiple buyers given distributional information about their values. Modeling these settings through an online contextual bandit framework allows obtaining results without knowledge of the prior distributions which may change based on the context. Our novel realizability assumption allows one to focus on predicting only the sufficient statistics required for running a specific algorithm. In the case of designing revenue optimal auctions, contexts may refer to the attributes of the item for sale and a sufficient statistic for the bidder value distributions are Myerson's reserve prices Myerson (1981).

8.2.2 Definitions & Notation

We begin by describing the original Pandora's Box formulation. Then, in Section 8.2.2.2 we describe our online extension, which solves a contextual instance of Pandora's Box at each round.

8.2.2.1 Original Pandora's Box Formulation

In Pandora's Box we are given a set $\mathcal{B}$ of n boxes each with cost c_i and value $v_i \sim \mathcal{D}_i$, where the distributions $\mathcal{D}_i$ and the costs c_i for each box are known and the distributions are independent. The goal is to adaptively choose a box of small cost while spending as little as possible in opening costs. When opening box i, the algorithm pays c_i and observes the value $v_i \sim \mathcal{D}_i$ instantiated inside the box. The formal definition follows.

Definition 8.12 (Pandora's Box cost)**.** *Let $\mathcal{P}$ and c_i be the set of boxes opened and the cost* *of the box selected, respectively. The cost of the algorithm is*

$$\boldsymbol{E}_{\mathcal{P},v_i\sim\mathcal{D}_i\forall i\in\mathcal{B}}\left[\min_{i\in\mathcal{P}} v_i + \sum_{i\in\mathcal{P}} c_i\right] \tag{8.3}$$

where the expectation is taken over the distributions of the values v_i and the (potentially random) choice of $\mathcal{P}$ by the algorithm.

Observe that an adaptive algorithm for this problem has to decide on: (a) a (potentially adaptive) order according to which it examines the boxes and, (b) a stopping rule. Weitzman's algorithm, first introduced in Weitzman (1979), and formally presented in Algorithm 15, gives a solution to Pandora's Box. The algorithm uses the order induced by the *reservation values* to open the boxes.

Algorithm 15: Weitzman's algorithm.

```
  Input: n boxes, costs $c_i$ and reservation values $\sigma_i$ for $i \in \mathcal{B}$
1 $\pi \leftarrow$ sort boxes by increasing $\sigma_i$
2 $v_{\min} \leftarrow \infty$
3 for every box $\pi_i$ do
4     Pay $c_i$ and open box $\pi_i$ and observe $v_i$
5     $v_{\min} \leftarrow \min(v_{\min}, v_i)$
6     if $v_{\min} < \sigma_{\pi_{i+1}}$ then
7         Stop and collect $v_{\min}$
8     end
9 end
```

We denote by $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}; \mathbf{c})$ the expected cost of running Weitzman's algorithm using reservation values $\boldsymbol{\sigma}$ on an instance with distribution $\mathfrak{D}$ and costs $\mathbf{c}$. Weitzman showed that this algorithm achieves the optimal expected cost of Equation 8.3 for the following selection of reservation values:

Theorem 8.13 (Weitzman (1979))**.** *Weitzman's algorithm is optimal for Pandora's Box when run with reservation values $\boldsymbol{\sigma}^*$ that satisfy $\boldsymbol{E}_{v_i\sim\mathcal{D}_i}[\text{ReLU}(\sigma_i^* - v)] = c_i$ for every box $i \in \mathcal{B}$, where* $\text{ReLU}(x) = \max\{x, 0\}$.

8.2.2.2 Online Contextual Pandora's Box

We now describe an online contextual extension of Pandora's Box. In Contextual Pandora's Box there is a set $\mathcal{B}$ of n boxes with costs $\mathbf{c} = (c_1, \dots, c_n)$[2]. At each round $t \in [T]$:

1. An (unknown) product distribution $\boldsymbol{\mathcal{D}}_t = (\mathcal{D}_{t,1}, \dots, \mathcal{D}_{t,n})$ is chosen and for every box i, a value $v_{t,i} \sim \mathcal{D}_{t,i}$ is independently realized.
2. A vector of contexts $\boldsymbol{x}_t = (\boldsymbol{x}_{t,1}, \dots \boldsymbol{x}_{t,n})$ is given to the learner, where $\boldsymbol{x}_{t,i} \in \mathbb{R}^d$ and $\|\boldsymbol{x}_{t,i}\|_2 \leqslant 1$ for each box $i \in \mathcal{B}$.
3. The learner decides on a (potentially adaptive) algorithm $\mathcal{A}$ based on past observations.
4. The learner opens the boxes according to $\mathcal{A}$ and chooses the lowest value found.
5. At the end of the round, the learner observes all the realized values of all boxes (*full-information* model), or observes only the values of boxes opened in the round (*bandit* model).

Assumption 8.14 (Realizability)**.** *There exist vectors $\boldsymbol{w}_1^*, \dots, \boldsymbol{w}_n^* \in \mathbb{R}^d$ and a function h, such that for every time $t \in [T]$ and every box $i \in \mathcal{B}$, the optimal reservation value $\sigma_{t,i}^*$ for the distribution $\mathcal{D}_{t,i}$ is equal to $h(\boldsymbol{w}_i^*, \boldsymbol{x}_{t,i})$, i.e.*

$$\boldsymbol{E}\,[v_{t,i} \sim \mathcal{D}_{t,i}]\,\mathrm{ReLU}(h(\boldsymbol{w}_i^*, \boldsymbol{x}_{t,i}) - v_{t,i}) = c_i.$$

The goal is to achieve low expected regret over T rounds compared to an optimal algorithm that has prior knowledge of the vectors $\boldsymbol{w}_i^*$ and, thus, can compute the exact reservation values of the boxes in each round and run Weitzman's optimal policy. The regret of an algorithm is defined as the difference between the cumulative Pandora's Box cost achieved by the algorithm compared to the cumulative cost achieved by running Weitzman's optimal policy at every round. That is:

[2]Every result in the paper holds even if the costs change for each $t \in [T]$ and can be adversarially selected.

Definition 8.15 (Expected Regret)**.** *The expected regret of an algorithm $\mathcal{A}$ that opens boxes $\mathcal{P}_t$ at round t over a time horizon T is*

$$\text{Regret}(\mathcal{A}, T) = \boldsymbol{E}\left[\sum_{t=1}^{T}\left(\min_{i\in\mathcal{P}_t} v_{t,i} + \sum_{i\in\mathcal{P}_t} c_i - \text{WEITZ}_{\mathbf{D}_t}(h(\boldsymbol{w}^*, \boldsymbol{x}_t); \mathbf{c})\right)\right].$$

The expectation is taken over the randomness of the algorithm, the contexts $\boldsymbol{x}_t$, distributions $\mathfrak{D}_t$ and realized values $\boldsymbol{v}_t \sim \mathfrak{D}_t$, over all rounds $t \in [T]$.

Remark. If the learner uses Weitzman's algorithm at every time step t, with reservation values $\sigma_{t,i} = h(\boldsymbol{w}_{t,i}, \boldsymbol{x}_{t,i})$ for some chosen parameter $\boldsymbol{w}_{t,i} \in \mathbb{R}^d$ for every box $i \in \mathcal{B}$, the regret can be written as

$$\text{Regret}(\mathcal{A}, T) = \mathbb{E}\Big[\sum_{t=1}^{T}\Big(\text{WEITZ}_{\mathbf{D}_t}(h(\boldsymbol{w}_t, \boldsymbol{x}_t); \mathbf{c}) - \text{WEITZ}_{\mathbf{D}_t}(h(\boldsymbol{w}^*, \boldsymbol{x}_t); \mathbf{c})\Big)\Big],$$

where $\boldsymbol{w}_t = (\boldsymbol{w}_{t,1}, \dots, \boldsymbol{w}_{t,n})$.

8.2.3 Reduction to Online Regression

In this section we give a reduction from Contextual Pandora's Box problem to an instance of online regression, while maintaining the regret guarantees given by online regression. We begin by formally defining the regression problem:

Definition 8.16 (Linear-Quadratic Online Regression)**.** *Online regression with loss ℓ is defined as follows: at every round t, the learner first chooses a prediction $\boldsymbol{w}_t$, then an adversary chooses an input-output pair $(\boldsymbol{x}_t, y_t)$ and the learner incurs loss $\ell(h(\boldsymbol{w}_t, \boldsymbol{x}_t) - y_t)$.*

In the ***costly feedback*** *setting, the learner may observe the input-output pair at the end of round t, if they choose to pay an information acquisition cost α. The* ***full information*** *setting, corresponds to the case where $\alpha = 0$, in which case the input-output pair is always visible. The regret of the learner after T rounds, when the learner has acquired information k times, is equal to*

$$\sum_{t=1}^{T} \ell(h(\boldsymbol{w}_t, \boldsymbol{x}_t) - y_t) - \min_{\boldsymbol{w}} \sum_{t=1}^{T} \ell(h(\boldsymbol{w}_t, \boldsymbol{x}_t) - y_t) + \alpha k.$$

Linear-Quadratic Online Regression *is the special case of online regression where the loss function $\ell(z)$ is chosen to be a linear-quadratic function of the form*

$$H_c(z) = \frac{1}{2}\text{ReLU}(z)^2 - cz \tag{8.4}$$

for some parameter $c > 0$.

The reduction presented in Algorithm 16 shows how we can use an oracle for linear-quadratic online regression, to obtain an algorithm for Contextual Pandora's Box problem. We show that our algorithm achieves $O(\sqrt{Tr(T)})$ regret when the given oracle has a regret guarantee of $r(T)$.

Theorem 8.17. *Given an oracle that achieves expected regret $r(T)$ for Linear-Quadratic Online Regression, Algorithm 16 achieves $2n\sqrt{Tr(T)}$ regret for the Contextual Pandora's Box problem. In particular, if the regret $r(T)$ is sublinear in T, Algorithm 16 achieves sublinear regret.*

Our algorithm works by maintaining a regression oracle for each box, and using it at each round to obtain a prediction on $w_{t,i}$. Specifically, in the prediction phase of the round the algorithm obtains a prediction and then uses the context $x_{i,t}$ to calculate an estimated reservation value for each box. Then, based on the estimated reservation values, it uses Weitzman's algorithm 15 to decide which boxes to open. Finally, it accumulates the Pandora's Box cost acquired by Weitzman's play at this round and the cost of any extra boxes opened by the oracle in the update phase (in the bandit setting). The update step is used to model the full-information setting (where the value of each box is always revealed at the end of the round) vs the costly feedback setting (where the value inside each box is only revealed if we paid the opening cost).

In the rest of this section we outline the proof of Theorem 8.17. The proof is based on obtaining robustness guarantees for Weitzman's algorithm when it is run with estimates instead of the true reservation values. In this case, we show that the cost incurred by Weitzman's algorithm is proportional to the error of the *approximate costs* of the boxes (8.18). This analysis is found on 8.2.4. Then, in Section 8.2.5 we exploit the form of the Linear-Quadratic loss functions to connect the robustness result with

Algorithm 16: $\mathcal{CPB}$: Contextual Pandora's Box

```
Input: Input:
Oracle $\mathcal{O}$ for Linear-Quadratic Online Regression.
For every box $i \in \mathcal{B}$, instantiate a copy $\mathcal{O}_i$ of the oracle with linear-quadratic
  loss $H_{c_i}$
for each round $t \in T$ do
    # Prediction Phase
    Call oracle $\mathcal{O}_i$ to get a prediction $\boldsymbol{w}_{t,i}$ for each box $i$
    Obtain context $\boldsymbol{x}_t^i$ for each box $i \in \mathcal{B}$
    Run Weitzman's algo 15 with reservation values $\sigma_{t,i} = h(\boldsymbol{w}_{t,i}, \boldsymbol{x}_{t,i})$ for
      each box $i \in \mathcal{B}$

    #Update Phase
    for every box $i \in \mathcal{B}$ do
        if the oracle $\mathcal{O}_i$ requests an input-output pair at this round then
            Observe value $v_{t,i}$ and give the input-output pair $(\boldsymbol{x}_{t,i}, v_{t,i})$
        end
    end
end
```

the regret of the Linear-Quadratic Online Regression problem and conclude our main Theorem 8.17 of this section.

An empirical evaluation of Algorithm 16 can be found in Section 8.A.7.1.

8.2.4 Weitzman's Robustness

We provide guarantees on Weitzman's algorithm 15 performance when instead of the optimal reservation values $\boldsymbol{\sigma}^*$ of the boxes, the algorithm uses estimates $\boldsymbol{\sigma} \neq \boldsymbol{\sigma}^*$. We first define the following:

Definition 8.18 (Approximate Cost)**.** *Given a distribution $\mathcal{D}$ such that $v \sim \mathcal{D}$ and a value σ, the approximate cost with respect to σ and $\mathcal{D}$ is defined as:*

$$c_{\mathcal{D}}(\sigma) = \boldsymbol{E}_{v \sim \mathcal{D}} \left[\mathrm{ReLU}(\sigma - v) \right]. \tag{8.5}$$

Moreover, given n boxes with estimated reservation values $\boldsymbol{\sigma}$ and distributions $\boldsymbol{\mathcal{D}}$ we denote the vector of approximate costs as $\boldsymbol{c}_{\boldsymbol{\mathcal{D}}}(\boldsymbol{\sigma}) = (c_{\mathcal{D}_1}(\sigma_1), \ldots, c_{\mathcal{D}_n}(\sigma_n))$.

Remark. Observe that, in the Pandora's Box setting, if box i has value distribution $\mathcal{D}_i$ opening cost c_i and optimal reservation value σ_i^*, then, by definition, the quantity $c_{\mathcal{D}_i}(\sigma_i^*)$ corresponds to the true cost, c_i, of the box. This also holds for the vector of approximate costs, i.e. $\mathbf{c}_{\mathfrak{D}}(\boldsymbol{\sigma}^*) = \mathbf{c}$.

We now state our robustness guarantee for Weitzman's algorithm. In particular, we show that the extra cost incurred due to the absence of initial knowledge of vectors $\boldsymbol{w}_i^*$ is proportional to the error of the approximate costs the boxes, as follows:

Theorem 8.19. *For a Pandora's Box instance with n boxes with distributions $\mathfrak{D}$, costs $\mathbf{c}$ and corresponding optimal reservation values $\boldsymbol{\sigma}^*$ so that $\mathbf{c} = \mathbf{c}_{\mathfrak{D}}(\boldsymbol{\sigma}^*)$, Weitzman's Algorithm 15, run with reservation values $\boldsymbol{\sigma}$ incurs cost at most*

$$\mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}; \mathbf{c}) \leqslant \mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}^*; \mathbf{c}) + \|\mathbf{c}_{\mathfrak{D}}(\boldsymbol{\sigma}) - \mathbf{c}\|_1.$$

Before showing Theorem 8.19, we prove the following lemma, that connects the optimal Pandora's Box cost of an instance with optimal reservation values $\boldsymbol{\sigma}^*$ to the optimal cost of the instance with optimal reservation values $\boldsymbol{\sigma}$.

Lemma 8.20. *Let $\mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}^*; \mathbf{c})$ and $\mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}; \mathbf{c}_{\mathfrak{D}}(\boldsymbol{\sigma}))$ be the optimal Pandora's Box costs corresponding to instances with optimal reservation values $\boldsymbol{\sigma}^*$ and $\boldsymbol{\sigma}$ respectively. Then*

$$\mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}^*; \mathbf{c}) \geqslant \mathrm{WEITZ}_{\mathfrak{D}}(\boldsymbol{\sigma}; \mathbf{c}_{\mathfrak{D}}(\boldsymbol{\sigma})) - \sum_{i \in \mathcal{B}} \mathrm{ReLU}(c_{\mathcal{D}_i}(\sigma_i) - c_i).$$

The proof of the above Lemma, together with that of Theorem 8.19, are deferred to Section 8.A.7.3.

8.2.5 Proof of Theorem 8.17

Moving on to show our main theorem, we connect the robustness Theorem 8.19 with the performance guarantee of the Linear-Quadratic Online Regression problem. The robustness guarantee of Weitzman's algorithm is expressed in terms of the error of the approximate costs of the boxes, while the regret of the Online Regression problem is measured in terms of the cumulative difference of the linear-quadratic loss functions $H_c(\cdot)$. Thus, we begin with the following lemma:

Lemma 8.21. *For any distribution $\mathcal{D}$ with $c_{\mathcal{D}}(\sigma^*) = c$, it holds that*

$$\boldsymbol{E}_{v\sim\mathcal{D}}\left[H_c(\sigma - v) - H_c(\sigma^* - v)\right] \geqslant \frac{1}{2}(c_{\mathcal{D}}(\sigma) - c_{\mathcal{D}}(\sigma^*))^2$$

The proof of the lemma is deferred to the Appendix.

Proof of Theorem 8.17. Recall that at every step $t \in [T]$, Theorem 16 runs Weitzman's algorithm as a subroutine, using an estimate $\boldsymbol{\sigma}_t$ for the optimal reservation values of the round, $\boldsymbol{\sigma}^*_t$. From the robustness analysis of Weitzman's algorithm we obtain that the regret of Algorithm 16 can be bounded as follows:

$$\begin{aligned}
\text{Regret}(\mathcal{CPB}, T) &= \mathbf{E}\left[\sum_{t\in[T]} \text{WEITZ}_{\mathcal{D}_t}(\boldsymbol{\sigma}_t; \mathbf{c}_t) - \text{WEITZ}_{\mathcal{D}_t}(\boldsymbol{\sigma}^*_t; \mathbf{c}_t)\right] \\
&\leqslant \sum_{t\in[T], i\in\mathcal{B}} |c_{\mathcal{D}_{t,i}}(\sigma_{t,i}) - c_{\mathcal{D}_{t,i}}(\sigma^*_{t,i})| \\
&\leqslant \sqrt{nT}\sqrt{\sum_{t\in[T], i\in\mathcal{B}} (c_{\mathcal{D}_{t,i}}(\sigma_{t,i}) - c_{\mathcal{D}_{t,i}}(\sigma^*_{t,i}))^2},
\end{aligned}$$

where the first inequality follows by Theorem 8.19 and for the last inequality we used that for any k-dimensional vector z we have that $\|z\|_1 \leqslant \sqrt{k}\|z\|_2$ and the fact that the above sum over $T, \mathcal{B}$ is equivalent to ℓ_1 norm on nT dimensions. Moreover, we have that

$$\begin{aligned}
\sum_{t\in[T], i\in\mathcal{B}} (c_{\mathcal{D}_{t,i}}(\sigma_{t,i}) - c_{\mathcal{D}_{t,i}}(\sigma^*_{t,i}))^2 &\leqslant 2\,\mathbf{E}\left[\sum_{t\in[T], i\in\mathcal{B}} \left(H_{c_{t,i}}(\sigma_{t,i} - v_{t,i}) - H_{c_{t,i}}(\sigma^*_{t,i} - v_{t,i})\right)\right] \\
&\leqslant 2 \cdot n \cdot r(T),
\end{aligned}$$

where for the first inequality we used Lemma 8.21, and then the guarantee of the oracle. Thus, we conclude that the total expected regret is at most $2n\sqrt{Tr(T)}$. □

8.2.6 A Special Case: Linear Contextual Pandora's Box

Using the reduction we developed in Section 8.2.3, we design efficient no-regret algorithms for Contextual Pandora's Box in the case where the mapping from contexts to reservation values is linear. That is, we assume that $h(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$.

8.2.6.1 Full Information Setting

In this section we study the full-information version of the Contextual Pandora's Box problem, where the algorithm observes the realized values of all boxes at the end of each round, irrespectively of which boxes were opened. Initially we show that there exists an online regression oracle, that achieves sublinear regret for the full information version of the Linear-Quadratic Online Regression problem. Then, in Corollary 8.23 we combine our reduction of Theorem 8.17 with the online regression oracle guarantee, to conclude that Algorithm 16 using this oracle is no-regret for Contextual Pandora's Box. The lemma and the theorem follow.

Lemma 8.22. *When* $h(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x}$, $\|\boldsymbol{w}\|_2 \leqslant M$ *and* $\|\boldsymbol{x}\|_2 \leqslant 1$, *there exists an oracle for Online Regression with Linear-Quadratic loss* H_c *under* ***full information*** *that achieves regret at most* $\max\{M, c\}\sqrt{2MT}$.

To show Theorem 8.22, we view Linear-Quadratic Online Regression as an instance of Online Convex Optimization and apply the *Follow The Regularized Leader* (FTRL) family of algorithms to obtain the regret guarantees.

Theorem 8.23. *In the full information setting, using the oracle of Theorem 8.22, Algorithm 16 for Contextual Pandora's Box achieves a regret of*

$$\text{Regret}(\mathcal{CPB}, T) \leqslant 3n\sqrt{\max\{M, c_{\max}\}}M^{1/4}T^{\frac{3}{4}},$$

assuming that for all times t *and boxes* $i \in \mathcal{B}$, $\|\boldsymbol{w}^*_{t,i}\|_2 \leqslant M$ *and* $c_{\max} = \max_{i \in \mathcal{B}} c_i$.

The process of using FTRL as an oracle is described in detail in Section 8.A.7.5, alongside the proofs of Theorem 8.22 and Corollary 8.23.

8.2.6.2 Bandit Setting

We move on to extend the results of the previous section to the bandit setting, and show how to obtain a no-regret algorithm for this setting by designing a regression oracle with costly feedback. In this case, the oracle $\mathcal{O}_i$ of Algorithm 16 of each box $i \in \mathcal{B}$ does not necessarily receive information on the value of the box after each round. However, in each round it chooses whether to obtain the information for the box by paying the opening cost c.

We initially show that we can use any regression oracle given for the full-information setting, in the costly feedback setting without losing much in terms of regret guarantees. This is formalized in the following theorem.

Lemma 8.24. *Given an oracle that achieves expected regret $r(T)$ for Online Regression with Linear-Quadratic loss H_c under **full information**, Algorithm 17 is an oracle for Linear-Quadratic Online regression **with costly feedback**, that achieves regret at most $kr(T/k) + cT/k$.*

Algorithm 17 obtains an oracle with costly feedback from a full information oracle. It achieves this by splitting the time interval $[T]$ in intervals of size k, and choosing a uniformly random time per interval to acquire the costly information about the input-output pair. The proof of Lemma 8.24 is included in Section 8.A.7.6 of the Appendix.

Algorithm 17: Costly Feedback oracle from Full Information

```
  Input: Parameter $k$, full information oracle $\mathcal{O}$
1 Split the times $[T]$ into $T/k$ intervals $\mathcal{I}_1 \dots, \mathcal{I}_{T/k}$
2 for Every interval $\mathcal{I}_\tau$ do
3     Pick a $t_p$ uniformly at random from $\mathcal{I}_\tau$
4     # Prediction Phase
5     Call $\mathcal{O}$ to get a vector $\boldsymbol{w}_\tau$. For each $t \in \mathcal{I}_\tau$ predict $\boldsymbol{w}_\tau$
6
7     # Update Phase
8     Obtain feedback for time $t_p \in \mathcal{I}_\tau$ and give input-output pair $(\boldsymbol{x}_{t_p}, y_{t_p})$ to $\mathcal{O}$.
9 end
```

Given that we can convert an oracle for full-information to one with costly feedback using Algorithm 17, we can now present the main theorem of this section (see Section 8.A.7.6 for the proof):

Theorem 8.25. *In the bandit setting, using the oracle of Lemma 8.24 together with the oracle of Theorem 8.22, Algorithm 16 for* Contextual Pandora*'s Box achieves a regret of*

$$\text{Regret}(\mathcal{CPB}, T) \leqslant 2n(2c_{\max} M \max\{M, c_{\max}\}^2)^{1/6} T^{5/6},$$

assuming that for all times t and boxes $i \in \mathcal{B}$, $\|\boldsymbol{w}^*_{t,i}\|_2 \leqslant M$ and $c_{\max} = \max_{i\in\mathcal{B}} c_i$.

8.A Appendix for Chapter 8

8.A.1 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 $\overline{f}(\boldsymbol{x})$ equal to

$$\begin{aligned}
\min\nolimits_{z\geqslant 0} \quad & \sum_{i\in\mathcal{B}} x_i + \frac{1}{|\mathcal{S}|} \sum_{i\in\mathcal{B}, s\in\mathcal{S}} c_i^s z_i^s && && \text{(LP-NA)}\\
\text{s.t.} \quad & \sum_{i\in\mathcal{B}} z_i^s = 1, && \forall s \in \mathcal{S} \\
& z_i^s \leqslant x_i && \forall i \in \mathcal{B}, s \in \mathcal{S}
\end{aligned}$$

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 8.A.5 of the appendix. For selecting 1 box we have Algorithm 19, for selecting k boxes Algorithm 20 and for selecting a matroid basis Algorithm 21.

Algorithm 18: 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             $(x, z) \leftarrow$ Ellipsoid $(k, \mathcal{LP})$
12             $k \leftarrow 2k$
13         until $(x, z)$ 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 $\pi$
18     end
19 end
```

The algorithm works in two steps; in the “explore" step (line 7) opening all boxes results in us exactly learning 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 18 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.

8.A.2 Bounding the mistakes

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

Definition 8.26 (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*

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

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 8.27. *Algorithm 18 has mistake bound*

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

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 8.27 is deferred to section 8.A.1 of the appendix.

8.A.3 Regret for different constraints

Moving on to show regret guarantees of Algorithm 18 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 8.28 (Regret for $0/\infty$)**.** *Algorithm 18, with $p_t = 1/\sqrt{T}$ has the following average regret, when $\mathcal{F} = \{\textit{Select 1 box}\}$ and $c_i \in \{0, \infty\}$.*

$$\boldsymbol{E}\left[\textit{Regret}_{NA}(\mathcal{A}_{\mathcal{F}}, T)\right] \leqslant \textit{OPT} + O\left(\frac{n^2}{\sqrt{T}}\right).$$

Proof of Theorem 8.28. Denote by M the mistake bound term, bounded above in Lemma 8.27. We calculate the total average regret

$$\begin{aligned}\mathbf{E}\left[\mathrm{Regret}_{NA}(\mathcal{A}_{\mathcal{F}}, T)\right] + \mathrm{OPT} &= \frac{1}{T}\left(M + \sum_{t=1}^{T} \mathbf{E}\left[|S_t|\right]\right) \\ &= \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + (1-p_t)\mathbf{E}\left[|S_t|\right]\right) \\ &\leqslant \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + 2\mathrm{OPT}\right) \\ &\leqslant M + 2\mathrm{OPT} + n\sum_{t=1}^{T} p_t \\ &\leqslant 2\mathrm{OPT} + O\left(\frac{n^2}{\sqrt{T}}\right)\end{aligned}$$

where initially we summed up the total regret of Algorithm 18 where the first term is the mistake bound from Lemma 8.27. Then we used the fact that $\mathrm{OPT}_t \leqslant \mathrm{OPT}$ and the solution found by the ellipsoid is within 2, and in the last line we used Since $\sum_{t=1}^{T} p_t \leqslant \sqrt{T}$ 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 8.29. *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 18, with $p_t = 1/\sqrt{T}$ has the following regret.*

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

The proof follows similarly to the $0/\infty$ case, and is deferred to section 8.A.1 of the appendix. Combining the different guarantees against the non-adaptive benchmark with Theorem 8.29 we get the following corollary.

Corollary 8.30 (Competing against NA, bandit setting)**.** *In the bandit setting, when competing with the non-adaptive benchmark, Algorithm 18 is*

- 3.16-*approximate no regret for* ***choosing*** 1 ***box*** (*using Theorem 8.32*)
- 12.64-*approximate no regret for* ***choosing*** k ***boxes*** (*using Theorem 8.34*)
- $O(\log k)$-*approximate no regret for* ***choosing a matroid basis*** (*using Theorem 8.36*)

8.A.4 Missing Proofs of Section 8.A.1

Before moving to the formal proof of Lemma 8.27, we recall the following lemma about the ellipsoid algorithm, bounding the number of steps it takes to find a feasible solution.

Lemma 8.31 (Lemma 3.1.36 from Grötschel et al. (1988))**.** *Given a full dimensional polytope* $P = \{x : Cx \leqslant d\}$*, 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.*

Using the lemma above, we can now prove Lemma 8.27, which we also restate below.

Lemma 8.27. *Algorithm 18 has mistake bound*

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

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 18, 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 18 as follows

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

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

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

of this constraint. Observe that $\{S_1, S_2, \dots S_\ell\} = S_c \subseteq \{\mathcal{C}_1, \mathcal{C}_2, \dots \mathcal{C}_{t_c}\}$. We calculate the probability that $N_T(c)$ is incremented on round k of S

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

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

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

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 8.31 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 8.29. *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 18, with $p_t = 1/\sqrt{T}$ has the following regret.*

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

Proof of Theorem 8.29. Denote by M be the mistake bound term, bounded in Lemma 8.27. Calculating the total average regret we get

$$\begin{aligned}
\mathbf{E}\left[\text{Regret}_{NA}(\mathcal{A}_\mathcal{F}, T)\right] + \text{OPT} &= \frac{1}{T}\left(M + \sum_{t=1}^{T}\mathbf{E}\left[S_t\right]\right) && \text{Definition}\\
&= \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t(n + \min_{i\in\mathcal{F}} c_i) + (1-p_t)\mathbf{E}\left[S_t\right]\right) && \text{Algorithm 18}\\
&\leqslant \frac{1}{T}\left(M + \sum_{t=1}^{T} p_t n + p_t \min_{i\in\mathcal{F}} c_i + 2\beta\text{OPT}_t\right) && \mathcal{A}_\mathcal{F}\text{, ellipsoid's loss}\\
&\leqslant (2\beta+1)\text{OPT} + \frac{1}{T}\left(M + n\sum_{t=1}^{T} p_t\right)\\
&\leqslant (2\beta+1)\text{OPT} + \frac{n}{\sqrt{T}}\\
&\leqslant (2\beta+1)\text{OPT} + O\left(\frac{n^2}{\sqrt{T}}\right) && \text{Lemma 8.27.}
\end{aligned}$$

where in the third to last inequality we used that $\min_{i\in\mathcal{F}} c_i \leqslant \mathrm{OPT}_t \leqslant \mathrm{OPT}$, then in the second to last we used that $\sum_{t=1}^{T} p_t \leqslant \sqrt{T}$ from Alabi et al. (2019). Therefore, subtracting OPT from both sides we get the theorem. □

8.A.5 Rounding LPs against the Non-Adaptive

8.A.5.1 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 19.

Theorem 8.32 (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 19: SPA vs NA fromChawla et al. (2020)

```
    Input: Solution $x, z$ to program (LP-NA); scenario $s$
1   $\sigma :=$ For $t \geqslant 1$, 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 $\frac{z_i^s}{x_i}$.
```

8.A.5.2 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 18 in the case we have to select k boxes. The guarantees are given in Theorem 8.34 and the rounding is presented in Algorithm 20. This extends the results of Chawla et al. (2020) for the case of selecting k items against the non-adaptive.

Lemma 8.33. *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 8.34.

Theorem 8.34. *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 8.33, we formulate our problem as a linear program as follows. The formulation is the same as LP-NA, we introduce constraints 8.6, since we need to pick k boxes instead of 1.

$$\begin{aligned} \text{minimize} \quad & \sum_{i\in\mathcal{B}} x_i + \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S}} c_i^s z_i^s & & \text{(LP-NA-k)} \\ \text{subject to} \quad & \sum_{i\in\mathcal{B}} z_i^s = k, & \forall s\in\mathcal{S} \quad & (8.6) \\ & z_i^s \leqslant x_i, & \forall i\in\mathcal{B}, s\in\mathcal{S} & \\ & x_i, z_i^s \in [0,1] & \forall i\in\mathcal{B}, s\in\mathcal{S} & \end{aligned}$$

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 20: 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 $x_i \geqslant 1/\beta$, from $\mathcal{X}_{\text{low}}$ select each box $i$ w.p. $\frac{x_i}{X}$
3
4  Denote by $k'$ and $\text{OPT}'_c$ the values of $\text{OPT}_c$ and $k$ restricted in the set $\mathcal{X}_{\text{low}}$
5  $\tau_s :=$ select all boxes that $z_i^s \geqslant 1/\beta$
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 8.33. 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 $x_i \geqslant 1/\beta$, select all that $z_i^s \geqslant 1/\beta$. This step only incurs at most $\beta(\text{OPT}_p + \text{OPT}_c)$ cost. The algorithm's value cost is $\text{ALG}_c =$

$\sum_{i:z_i^s \geqslant 1/\beta} c_i$ while $\text{OPT}_c = \sum_i z_i^s c_i \geqslant \sum_{i:z_i^s \geqslant 1/\beta} c_i z_i^s \geqslant 1/\beta \sum_{i:z_i^s \geqslant 1/\beta} c_i = 1/\beta \text{ALG}_c$. 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 $\mathcal{L}_s = \{i : c_i \leqslant \alpha \text{OPT}'_c/k'\}$.
  - **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

$$\begin{aligned}
\textbf{Pr}\left[\text{choose j'th}|\text{have chosen } j-1\right] &\geqslant \frac{\sum_{i \in L_s} x_i - j/\beta}{X} && x_i \leqslant 1/\beta \forall x_i \in \mathcal{X}_{\text{low}} \\
&\geqslant \frac{\sum_{i \in L_s} z_i^s - j/\beta}{X} && \text{LP constraint} \\
&\geqslant \frac{(1-1/\alpha)k' - j/\beta}{\text{OPT}'_p} && \text{Markov's Inequality} \\
&\geqslant \frac{(1-1/\alpha)k' - k'/\beta}{\text{OPT}'_p} && \text{Since } j \leqslant k' \\
&\geqslant \frac{(\alpha\beta - \beta - \alpha)k'}{\alpha\beta \text{OPT}'_p}
\end{aligned}$$

Therefore the expected time until we choose k' boxes is

$$\begin{aligned}
\textbf{E}\left[\text{ALG}_p\right] &= \sum_{j=1}^{k'} \frac{1}{\textbf{Pr}\left[\text{choose j'th}|\text{have chosen } j-1\right]} \\
&\leqslant \sum_{j=1}^{k'} \alpha\beta \frac{\text{OPT}'_p}{(\alpha\beta - \alpha - \beta)k'} \\
&= \alpha\beta \frac{\text{OPT}'_p}{\alpha\beta - \alpha - \beta}
\end{aligned}$$

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

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

8.A.5.3 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 8.36 and the rounding presented in Algorithm 21. This case also extends the results of Chawla et al. (2020).

Lemma 8.35. *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 8.36.

Theorem 8.36. *We can efficiently find a partially-adaptive strategy for optimal search over a matroid of rank* k *that is* $O(\log k)$*-competitive against the optimal non-adaptive strategy.*

In order to present the proof for Lemma 8.35, 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 8.6 with constraint 8.7 which ensures we select at most $r(A)$ number of elements for every set and constraint (8.8) ensures that whatever set A of boxes is already chosen, there still enough elements to cover the rank constraint.

$$\begin{array}{llcll}
\text{minimize} & \sum_{i\in\mathcal{B}} x_i & + & \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{B},s\in\mathcal{S}} c_i^s z_i^s & \qquad \text{(LP-NA-matroid)} \\
\text{subject to} & \sum_{i\in\mathcal{B}} z_i^s & \leqslant & r(A), \qquad \forall s\in\mathcal{S}, A\subseteq\mathcal{B} & \qquad (8.7) \\
& \sum_{i\in A} z_i^s & \geqslant & r([n]) - r(A) \quad \forall A\subseteq\mathcal{B}, \forall s\in\mathcal{S} & \qquad (8.8) \\
& z_i^s & \leqslant & x_i, \qquad \forall i\in\mathcal{B}, s\in\mathcal{S} & \qquad (8.9) \\
& x_i, z_i^s & \in & [0,1] \qquad \forall i\in\mathcal{B}, s\in\mathcal{S} &
\end{array}$$

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 $\text{ALG}_p, \text{ALG}_c$ the respective algorithm's costs.

Algorithm 21: 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 $\mathcal{X}_{low} = \{i : x_i < 1/\beta\}$ and $X = \sum_{i\in\mathcal{X}_{low}} x_i$
2 $\sigma :=$ open all boxes that $x_i \geqslant 1/\beta$, from $\mathcal{X}_{low}$ select each box $i$ w.p. $\frac{x_i}{X}$
3
4 Denote by $k^j$ and $\text{OPT}_c^j$ the values of $\text{OPT}_c$ and $k$ restricted in the set $\mathcal{X}_{low}$
    when $j$ boxes are selected.
5 $\tau_s :=$ select all boxes that $z_i^s \geqslant 1/\beta$
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 8.35. Similarly to Lemma 8.33, 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 $x_i \geqslant 1/\beta$, select all that $z_i^s \geqslant 1/\beta$. This step only incurs at most $\beta(\text{OPT}_p + \text{OPT}_c)$ cost.
- **Step 2**: let $\mathcal{X}_{low} = \{i : x_i < 1/\beta\}$. Denote by OPT'_c and k' the new values of OPT_c and k restricted on $\mathcal{X}_{low}$. At every step, after having selected j boxes, we restrict our search to the set of low cost boxes $\mathcal{L}_s^j = \{i : v_i \leqslant \alpha\text{OPT}_c^j/k^j\}$ 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 $\mathcal{X}_{low}$. Observe that the probability of choosing the j'th box from L_s given

that we already have chosen $j-1$ is

$$\begin{aligned}
\mathbf{Pr}\left[\text{choose } j\text{'th}|\text{have chosen } j-1\right] &\geqslant \frac{\sum_{i\in L_s^{j-1}} x_i}{X} \\
&\geqslant \frac{\sum_{i\in L_s^{j-1}} z_i^s}{X} && \text{From LP constraint (8.9)} \\
&\geqslant \frac{k-(k-j)}{X} && \text{From LP constraint (8.8)} \\
&= \frac{j}{\text{OPT}_p'}
\end{aligned}$$

Therefore the expected time until we choose k' boxes is

$$\begin{aligned}
\mathbf{E}\left[\text{ALG}_c\right] &= \sum_{j=1}^{k'} \frac{1}{\mathbf{Pr}\left[\text{choose } j\text{'th}|\text{have chosen } j-1\right]} \\
&\leqslant \text{OPT}_p' \sum_{j=1}^{k'} \frac{1}{j} \\
&\leqslant \log k \cdot \text{OPT}_p
\end{aligned}$$

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

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

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

8.A.6 Missing Proofs of Section 8.1

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

Lemma 8.37 (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 $\boldsymbol{z}\nabla^2 U(\boldsymbol{x})\boldsymbol{z} \geqslant ||\boldsymbol{z}||_1^2$ 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

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

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

Theorem 8.6. *The average regret of Algorithm 13 is*

$$\text{Regret}_{PA}(\mathcal{A}, T) \leqslant 2n\sqrt{\frac{\log n}{T}}$$

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 8.37 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 8.5 we obtain the guarantee of the theorem, by setting $\eta = \frac{\sqrt{\log n}}{\sqrt{T}}$. □

8.A.7 Appendix of Section 8.2

8.A.7.1 Experiments

We simulate the $\mathcal{CPB}$ algorithm on synthetic data, where the box values implement uniform distributions on different intervals at each round. We simulate $n = 10$ boxes with identical costs $c_i = 1$ for all $i \in \mathcal{B}$, for $T = 300$ rounds. The value distributions of the boxes are generated as follows:

- Each box i corresponds to a fixed random vector $\boldsymbol{w}_i^* \in \mathbb{R}^d$ with $d = 5$, selected uniformly at the beginning of the simulation, such that $\|\boldsymbol{w}_i^*\|_2 \leqslant M = 4$.
- At each round $t \in [T]$ and for each box $i \in \mathcal{B}$, a context $\boldsymbol{x}_{t,i}$ is uniformly drawn in $\mathbb{R}^d$ such that $\|\boldsymbol{x}_{t,i}\|_2 \leqslant 1$.
- The value $v_{i,t}$ of each box i at time t is drawn from a uniform distribution on the interval $[0, B_{t,i}]$. The right-bound $B_{t,i}$ is computed such that the reservation value $\sigma_{t,i}^*$ of the uniform distribution satisfies the realizability assumption, that is $\sigma_{t,i}^* = \boldsymbol{x}_{t,i}^T \boldsymbol{w}_i^*$.

We implement the $\mathcal{CPB}$ algorithm for the estimation of vector $\boldsymbol{w}_i^*$ of each box from the observations $(\boldsymbol{x}_{t,i}^T, v_{i,t})_{t\in[T]}$ using the FTRL oracle. We compare its performance with the performance of linear regression applied to observations $(\boldsymbol{x}_{t,i}^T, v_{i,t})_{t\in[T]}$. The performance of the two methods (averaged over 20 repetitions) is depicted in Figure 8.1, along with the error bars.

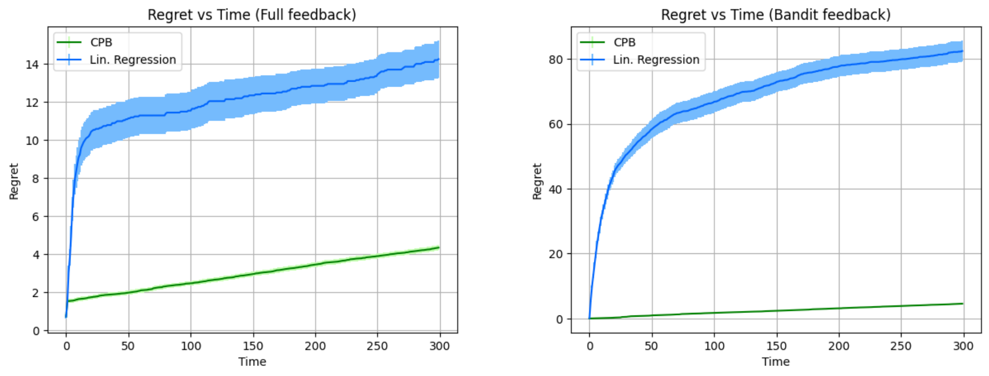

Figure 8.1: The regret as a function of time in a) Full feedback setting, and b) Bandit feedback setting.

In our first experiment, full feedback is available, i.e. the samples $(\boldsymbol{x}_{t,i}^T, v_{i,t})$ are available to both algorithms for all $i \in \mathcal{B}, t \in [T]$. In the second experiment the algorithms have bandit feedback. That is, a sample $(\mathbf{x}_{t,i}^T, v_{i,t})$ is available for an algorithm only if box i was opened at time t by the algorithm. We plot the regret (defined in (8.4)) of the algorithms as a function of time t. As expected, the regret of both algorithms is smaller under full feedback, since there is more information available at each round. In addition, the regret increases with sublinear rate as t grows, which is compatible with our theoretical guarantees in both the full-feedback and the bandit setting. Finally, in both settings, using the $\mathcal{CPB}$ algorithm leads to significantly smaller regret compared to linear regression.

8.A.7.2 Impossibility beyond Realizability

We now provide an example to indicate that without the realizability assumption, our setting is not only computationally hard (even in the offline case) (**?**), but also becomes information-theoretically hard in the online case:

Consider the simple setting of two boxes, with costs either 0 or 1. The context provides us with some information on where is the 0 in each round, and our objective it to select 0 the maximum amount of times. Clearly, in the offline case the above setting coincides with that of agnostic learning, which is known to be computationally hard, even for linear functions (e.g., linear classification).

In the online setting, where an adversary can choose the context to give us at each round, the problem is information-theoretically hard. Similarly, assume that a context is a real number in $[0, 1]$ and that there exists a threshold x, which decides which box gives 0 and which 1. The adversary can always give us a threshold in the uncertainty region, and force us to make a mistake at every round, thus accumulating linear cost over time, while the cost accumulated by the optimal algorithm (which knows the context-value relation) is 0.

8.A.7.3 Proofs from Section 8.2.4

Theorem 8.19. *For a* PANDORA's BOX *instance with n boxes with distributions $\mathcal{D}$, costs $\mathbf{c}$ and corresponding optimal reservation values $\boldsymbol{\sigma}^*$ so that $\mathbf{c} = \mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma}^*)$, Weitzman's Algorithm 15, run with reservation values $\boldsymbol{\sigma}$ incurs cost at most*

$$\mathrm{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}; \mathbf{c}) \leqslant \mathrm{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*; \mathbf{c}) + \|\mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma}) - \mathbf{c}\|_1.$$

Proof. Assume we are using Weitzman's Algorithm with reservation values $\boldsymbol{\sigma}$ in an instance with optimal reservation values $\boldsymbol{\sigma}^*$ and costs $\mathbf{c}$. Let $\mathcal{P}$ be the set of boxes opened during the algorithm's run. Then, the PANDORA's Box cost incurred at the end of the run can be bounded as follows:

$$\begin{aligned}
\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma};\mathbf{c}) &= \mathbf{E}_{v\sim\mathcal{D}}\left[\min_{i\in\mathcal{P}} v_i + \sum_{i\in\mathcal{P}} c_i\right] \\
&= \mathbf{E}_{v\sim\mathcal{D}}\left[\min_{i\in\mathcal{P}} v_i + \sum_{i\in\mathcal{P}} c_i + c_{\mathcal{D}_i}(\sigma_i) - c_{\mathcal{D}_i}(\sigma_i)\right] \\
&= \text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma};\mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma})) + \mathbf{E}_{v\sim\mathcal{D}}\left[\sum_{i\in\mathcal{P}} (c_i - c_{\mathcal{D}_i}(\sigma_i))\right] \\
&\leqslant \text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}) + \\
&\qquad \mathbf{E}_{v\sim\mathcal{D}}\left[\sum_{i\in B} \text{ReLU}(c_{\mathcal{D}_i}(\sigma_i) - c_i) + \sum_{i\in\mathcal{P}} (c_i - c_{\mathcal{D}_i}(\sigma_i))\right] \\
&\leqslant \text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}) + \mathbf{E}_{v\sim\mathcal{D}}\left[\sum_{i\in\mathcal{B}} |c_i - c_{\mathcal{D}_i}(\sigma_i)|\right],
\end{aligned}$$

where in the first inequality we used Lemma 8.20 and in the last inequality that every box from each of the sets $\mathcal{B}$ and $\mathcal{P}$ can contribute at most once in the sum, because of the ReLU function. □

Lemma 8.20. *Let* $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c})$ *and* $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma};\mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma}))$ *be the optimal PANDORA's Box costs corresponding to instances with optimal reservation values* $\boldsymbol{\sigma}^*$ *and* $\boldsymbol{\sigma}$ *respectively. Then*

$$\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}) \geqslant \text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma};\mathbf{c}_{\mathcal{D}}(\boldsymbol{\sigma})) - \sum_{i\in\mathcal{B}} \text{ReLU}(c_{\mathcal{D}_i}(\sigma_i) - c_i).$$

Proof. Consider the optimal strategy $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c})$ for the instance with reservation values $\boldsymbol{\sigma}^*$, and assume we use the same strategy in the instance where the 'actual reservation values are $\hat{\boldsymbol{\sigma}}$. That means our algorithm $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}_{\mathcal{D}}(\hat{\boldsymbol{\sigma}}))$ orders the boxes according to the reservation values $\boldsymbol{\sigma}^*$ and stops when $\min_i v_i \leqslant \sigma^*_{i+1}$. Denote by $\mathcal{P}$ the set of boxes probed (opened) by the algorithm $\text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}_{\mathcal{D}}(\hat{\boldsymbol{\sigma}}))$, then the cost is

$$\text{WEITZ}_{\mathcal{D}}(\hat{\boldsymbol{\sigma}};\mathbf{c}_{\mathcal{D}}(\hat{\boldsymbol{\sigma}})) \leqslant \text{WEITZ}_{\mathcal{D}}(\boldsymbol{\sigma}^*;\mathbf{c}_{\mathcal{D}}(\hat{\boldsymbol{\sigma}}))$$

$$\begin{aligned}
&= \mathbf{E}_{\boldsymbol{v}\sim \mathbf{D}}\left[\min_{i\in\mathcal{P}} v_i + \sum_{i\in\mathcal{P}} c_{\mathcal{D}_i}(\hat{\sigma}_i)\right] \\
&= \mathbf{E}_{\boldsymbol{v}\sim \mathbf{D}}\left[\min_{i\in\mathcal{P}} v_i\right] + \mathbf{E}_{\boldsymbol{v}\sim \mathbf{D}}\left[\sum_{i\in\mathcal{P}} c_{\mathcal{D}_i}(\hat{\sigma}_i) - c_{\mathcal{D}_i}(\sigma_i^*) + c_{\mathcal{D}_i}(\sigma_i^*)\right] \\
&= \text{WEITZ}_{\boldsymbol{\mathcal{D}}}(\boldsymbol{\sigma}^*; \mathbf{c}) + \sum_{i\in\mathcal{P}} (c_{\mathcal{D}_i}(\hat{\sigma}_i) - c_{\mathcal{D}_i}(\sigma_i^*)) \\
&\leqslant \text{WEITZ}_{\boldsymbol{\mathcal{D}}}(\boldsymbol{\sigma}^*; \mathbf{c}) + \sum_{i\in\mathcal{P}} \text{ReLU}(c_{\mathcal{D}_i}(\hat{\sigma}_i) - c_{\mathcal{D}_i}(\sigma_i^*)),
\end{aligned}$$

where the first inequality follows by the definition of the optimal, and the first equality is the actual cost of our algorithm, since in our instance we pay $c_{\mathcal{D}_i}(\hat{\sigma}_i)$ for each box. □

8.A.7.4 Proofs from Section 8.2.5

Lemma 8.21. *For any distribution $\mathcal{D}$ with $c_{\mathcal{D}}(\sigma^*) = c$, it holds that*

$$\boldsymbol{E}_{v\sim\mathcal{D}}\left[H_c(\sigma - v) - H_c(\sigma^* - v)\right] \geqslant \frac{1}{2}(c_{\mathcal{D}}(\sigma) - c_{\mathcal{D}}(\sigma^*))^2$$

Proof. Recall from Definition 8.16 that

$$H_c(\sigma - v) = \frac{1}{2}\text{ReLU}(\sigma - v)^2 - c(\sigma - v) = \left\{ \begin{array}{ll} \frac{1}{2}(\sigma - v)^2 - c(\sigma - v) & \text{if } \sigma \geqslant v \\ -c(\sigma - v), & \text{if } \sigma < v \end{array} \right\}$$

and

$$H_c'(\sigma - v) = \text{ReLU}(\sigma - v) - c.$$

We also use that for $\sigma = \sigma^*$ using Weitzman's theorem for optimal reservation values (Theorem 8.13) we have

$$\mathbf{E}_v\left[H_c'(\sigma^* - v)\right] = 0.$$

We need to compare $H_c(\sigma - v) - H_c(\sigma^* - v)$ to $|c_D(\sigma) - c_D(\sigma^*)|$. Using that $\int_a^b f'(x)\,dx = f(b) - f(a)$ and changing the order between expectation and integration, we obtain

$$\mathbf{E}_{v\sim D}\left[H_c(\sigma - v) - H_c(\sigma^* - v)\right] = \mathbf{E}_{v\sim D}\left[\int_{\sigma^*}^{\sigma} H_c'(\sigma' - v)\,d\sigma'\right]$$

$$
\begin{aligned}
&= \int_{\sigma^*}^{\sigma} \mathbf{E}_{\nu\sim D}\left[H_c'(\sigma'-\nu)\right]\, d\sigma' \\
&= \int_{\sigma^*}^{\sigma} \mathbf{E}_{\nu\sim D}\left[H_c'(\sigma'-\nu)\right] - \mathbf{E}_{\nu\sim D}\left[H_c'(\sigma^*-\nu)\right]\, d\sigma' \\
&= \int_{\sigma^*}^{\sigma} \mathbf{E}_{\nu\sim D}\left[\mathrm{ReLU}(\sigma'-\nu)\right] - \mathbf{E}_{\nu\sim D}\left[\mathrm{ReLU}(\sigma^*-\nu)\right]\, d\sigma' \\
&= \int_{\sigma^*}^{\sigma} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma', \qquad (8.10)
\end{aligned}
$$

where the last inequality above is by definition of $c_D(\cdot)$. We now distinguish between the following cases for the reservation value σ compared to the optimal reservation value σ^*.

Case $\sigma \geqslant \sigma^*$: observe that c_D is 1-Lipschitz, therefore we have that $|c_D(\sigma) - c_D(\sigma')| < |\sigma - \sigma'|$. Moreover, when $\sigma' \geqslant \sigma^*$ we have $c_D(\sigma') \geqslant c_D(\sigma^*)$. Thus Equation 8.10 becomes:

$$
\begin{aligned}
\mathbf{E}_{\nu\sim D}\left[H_c(\sigma-\nu) - H_c(\sigma^*-\nu)\right] &= \int_{\sigma^*}^{\sigma} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma' \\
&\geqslant \int_{\sigma-(c_D(\sigma)-c_D(\sigma^*))}^{\sigma} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma' \\
&\geqslant \int_{\sigma-(c_D(\sigma)-c_D(\sigma^*))}^{\sigma} (c_D(\sigma) - c_D(\sigma^*) - (\sigma-\sigma'))\, d\sigma' \\
&= \frac{1}{2}(c_D(\sigma) - c_D(\sigma^*))^2,
\end{aligned}
$$

where for the first inequality we used that $\sigma - \sigma^* \geqslant c_D(\sigma) - c_D(\sigma^*)$ and for the second that $\sigma - \sigma' \geqslant c_D(\sigma) - c_D(\sigma')$.

Case $\sigma < \sigma^*$: as before, Equation 8.10 can be written as

$$
\begin{aligned}
\mathbf{E}_{y\sim D}\left[H_c(\sigma-\nu) - H_c(\sigma^*-\nu)\right] &= \int_{\sigma^*}^{\sigma} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma' \\
&= -\int_{\sigma}^{\sigma^*} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma' \\
&\geqslant -\int_{\sigma}^{\sigma-(c_D(\sigma)-c_D(\sigma^*))} (c_D(\sigma') - c_D(\sigma^*))\, d\sigma' \\
&\geqslant \int_{\sigma}^{\sigma-(c_D(\sigma)-c_D(\sigma^*))} (-c_D(\sigma) + c_D(\sigma^*) - (\sigma'-\sigma))\, d\sigma'
\end{aligned}
$$

$$= \frac{1}{2}(c_D(\sigma) - c_D(\sigma^*))^2,$$

where for the first inequality we used that $\sigma^* - \sigma \geqslant c_D(\sigma^*) - c_D(\sigma)$ and the fact that $c_D(\sigma') - c_D(\sigma^*) \leqslant 0$. For the second we used that $-c_D(\sigma') \geqslant -c_D(\sigma) - (\sigma' - \sigma)$.

□

8.A.7.5 Regression via FTRL and Proofs of Section 8.2.6.1

Recall our regression function defined in equation (8.4). We define $f_t : \mathbb{R}^d \to \mathbb{R}$, as the loss function at time $t \in [T]$ as

$$f_t(\boldsymbol{w}) = H_c(\boldsymbol{w}^T \boldsymbol{x}_t - y_t) = \frac{1}{2}(\boldsymbol{w}^T \boldsymbol{x}_t - y_t)_+^2 - c(\boldsymbol{w}^T \boldsymbol{x}_t - y_t).$$

As these functions are convex, we can treat the problem as an online convex optimization. Specifically, the problem we solve is the following.

1. At every round t, we pick a vector $\boldsymbol{w}_t \in \mathbb{R}^d$.
2. An adversary picks a convex function $f_t : \mathbb{R}^d \to \mathbb{R}$ induced by the input-output pair $(\boldsymbol{x}_t, y_t)$ and we incur loss $f_t(\sigma_t)$.
3. At the end of the round, we observe the function f_t.

In order to solve this problem, we use a family of algorithms called Follow The Regularized Leader (FTRL). In these algorithms, at every step t we pick the solution $\boldsymbol{w}_t$ that would have performed best so far while adding a regularization term $U(\boldsymbol{w}_t)$ for stability reasons. That is, we choose

$$\boldsymbol{w}_t = \arg \min_{\boldsymbol{w}:\|\boldsymbol{w}\|_2 \leqslant M} \sum_{\tau=1}^{t-1} f_t(\boldsymbol{w}) + U(\boldsymbol{w}).$$

The guarantees for these algorithms are presented in the following lemma.

Lemma 8.38 (Theorem 2.11 from Shalev-Shwartz (2012))**.** *Let $f_1, \dots, f_T$ 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)$\cdot T \leqslant U_{\max} - U_{\min} + TL^2\eta$*

Observe that in order to achieve the guarantees of Lemma 8.38, we need the functions to have some convexity and Lipschitzness properties, for which we show the following lemma.

Lemma 8.39. *The function* $f_t(\boldsymbol{w}) = H_c(\boldsymbol{w}^T\boldsymbol{x}_t - y_t)$ *is convex and* $\max\{M, c\}$*-Lipschitz when* $\|x_t\|_2 \leqslant 1$ *and* $\|\boldsymbol{w}\|_2 \leqslant M$*.*

Proof of Lemma 8.39. We first show convexity and Lipschitzness for the function $H_c(z) = \frac{1}{2}\text{ReLU}(z)^2 - cz$ for $z \in \mathbb{R}$.

Consider the derivative $H_c'(z) = \text{ReLU}(z) - c$ and the second derivative $H_c''(z) = \mathbf{1}(z \geqslant 0)$ and notice that the second derivative is always non-negative which implies convexity.

To bound the Lipschitzness of H_c, we consider the maximum absolute value of the derivative which is at most $\max\{c, \text{ReLU}(z)\}$.

We now turn our attention to f_t and notice that f_t is convex as a composition of a convex function with a linear function. To show Lipschitzness we must bound the norm of the gradient of f_t which is:

$$\|\nabla f_t(\boldsymbol{w})\|_2 = |H_c'(\boldsymbol{w}^T\boldsymbol{x}_t - y_t)|\|\boldsymbol{x}\|_2 \leqslant \max\{c, M\},$$

where the last inequality follows as the maximum value of $\boldsymbol{w}^T\boldsymbol{x}_t - y_t$ is at most M. □

Lemma 8.22. *When* $h(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}$, $\|\boldsymbol{w}\|_2 \leqslant M$ *and* $\|\boldsymbol{x}\|_2 \leqslant 1$*, there exists an oracle for Online Regression with Linear-Quadratic loss* H_c *under* ***full information*** *that achieves regret at most* $\max\{M, c\}\sqrt{2MT}$*.*

Proof. We use the following regularizer $U(\boldsymbol{w}) = \frac{1}{2\eta}\|\boldsymbol{w}\|_2^2$, and observe that this is η-strongly convex with $U_{\min} = 0$ and $U_{\max} = M/(2\eta)$. From Lemma 8.39 we have that the loss function is convex and $\max\{c, M\}$-Lipschitz, therefore using Lemma 8.38 with $\eta = \frac{\sqrt{M}}{\max\{c,M\}\sqrt{2T}}$ we get that $\text{Regret}(\text{FTRL}, T) \leqslant \frac{M}{2\eta} + \eta(\max\{c, M\})^2 T = \max\{c, M\}\sqrt{2MT}$. □

Theorem 8.23. *In the full information setting, using the oracle of Theorem 8.22, Algorithm 16 for* Contextual Pandora*'s Box achieves a regret of*

$$\text{Regret}(\mathcal{CPB}, T) \leqslant 3n\sqrt{\max\{M, c_{\max}\}}M^{1/4}T^{\frac{3}{4}},$$

assuming that for all times t *and boxes* $i \in \mathcal{B}$, $\|\boldsymbol{w}^*_{t,i}\|_2 \leqslant M$ *and* $c_{\max} = \max_{i\in\mathcal{B}} c_i$.

Proof. The regret of Contextual Pandora's Box can be upper bounded as follows:

$$\begin{aligned}\text{Regret}(\mathcal{CPB}, T) &\leqslant 2n\sqrt{\text{Tr}(T)} && \text{by Theorem 8.17}\\ &\leqslant 3n\sqrt{\max\{M, c\}}M^{1/4}T^{\frac{3}{4}}. && \text{by Lemma 8.22}\end{aligned}$$

□

8.A.7.6 Proofs from Section 8.2.6.2

In this section we show how to obtain the guarantees of FTRL in the case of costly feedback proving Lemma 8.24.

While the proof is standalone, the analysis uses ideas from Gergatsouli and Tzamos (2022) (in particular Lemma 4.2 and Algorithm 2). We give a simplified presentation for the case of online regression with improved bounds.

Lemma 8.24. *Given an oracle that achieves expected regret* $r(T)$ *for Online Regression with Linear-Quadratic loss* H_c *under* ***full information****, Algorithm 17 is an oracle for Linear-Quadratic Online regression* ***with costly feedback****, that achieves regret at most* $kr(T/k) + cT/k$.

Proof. Recall that in the online regression problem, we obtain loss f_t at any time step where $f_t(\boldsymbol{w}) = H_c(\boldsymbol{w}^T\boldsymbol{x}_t - y_t) = \frac{1}{2}\text{ReLU}(\boldsymbol{w}^T\boldsymbol{x}_t - y_t)^2 - c(\boldsymbol{w}^T\boldsymbol{x}_t - y_t)$.

To analyze the regret for the costly feedback setting, we consider the regret of two related settings for a full-information online learner but with smaller number of time steps T/k.

1. **Average costs setting**: the learner observes at each round τ a single function

   $$\bar{f}_\tau = \frac{1}{k}\sum_{t\in\mathcal{I}_\tau} f_t,$$

   which is the average of the k functions in the corresponding interval $\mathcal{I}_\tau$.

2. **Random costs setting**: the learner observes at each round τ a single function

   $$f^r_\tau = f_{t_p} \quad \text{with} \quad t_p \sim \text{Uniform}(\mathcal{I}_\tau)$$

sampled uniformly among the k functions f_t for $t \in \mathcal{I}_\tau$.

The guarantee of the full information oracle implies that for the random costs setting we obtain regret $r(T/k)$. That is, the oracle chooses a sequence $\boldsymbol{w}_1, \dots, \boldsymbol{w}_{T/k}$ such that

$$\sum_{\tau=1}^{T/k} f_\tau^r(\boldsymbol{w}_\tau) \leqslant \min_{\boldsymbol{w}} \sum_{\tau=1}^{T/k} f_\tau^r(\boldsymbol{w}) + r(T/k).$$

Denote by $\boldsymbol{w}^* = \operatorname{argmin}_{\boldsymbol{w} \in \mathbb{R}^d} \sum_{\tau=1}^{T/k} \bar{f}_\tau(\boldsymbol{w})$ be the minimizer of the $\bar{f}_t$ over the T/k rounds. Note that this is also the minimizer of $\sum_{t=1}^{T} f_t(\boldsymbol{w})$. From the above regret guarantee, we get that

$$\sum_{\tau=1}^{T/k} (f_\tau^r(\boldsymbol{w}_\tau) - f_\tau^r(\boldsymbol{w}^*)) \leqslant r(T/k).$$

Since $\mathbf{E}\left[f_\tau^r(\boldsymbol{z}_\tau)\right] = \bar{f}_\tau(\boldsymbol{z}_\tau)$ when the expectation is over the choice of $t_p \sim \text{Uniform}(\mathcal{I}_\tau)$. Taking expectation in the above, we get that

$$\sum_{\tau=1}^{T/k} (\bar{f}_\tau(\boldsymbol{w}_\tau) - \bar{f}_\tau(\boldsymbol{w}^*)) \leqslant r(T/k),$$

which implies that the regret in the average costs setting is also $r(T/k)$.

We now obtain the final result by noticing that the regret of the algorithm is at most k times the regret of the average costs setting and incurs an additional overhead of $\frac{T}{k}c$ for the information acquisition cost in the T/k rounds where the input-output pairs are queried. Thus, overall the regret is bounded by $kr(T/k) + cT/k$. $\square$

Theorem 8.25. *In the bandit setting, using the oracle of Lemma 8.24 together with the oracle of Theorem 8.22, Algorithm 16 for* Contextual Pandora*'s Box achieves a regret of*

$$\text{Regret}(\mathcal{CPB}, T) \leqslant 2n(2c_{\max} M \max\{M, c_{\max}\}^2)^{1/6} T^{5/6},$$

assuming that for all times t and boxes $i \in \mathcal{B}$, $\|\boldsymbol{w}_{t,i}^*\|_2 \leqslant M$ and $c_{\max} = \max_{i \in \mathcal{B}} c_i$.

Proof. Initially observe that by combining Theorem 8.22 and Lemma 8.24 we get that there exists an oracle for online regression with loss H_c under the costly feedback setting that guarantees

$$\text{Regret}(\mathcal{O}_{\text{costly}}, T) \leqslant L\sqrt{2kMT} + cT/k$$

with $L = \max\{M, c\}$. Setting $k = \left(\frac{Tc^2}{2ML^2}\right)^{1/3}$, we obtain regret $(2cML^2)^{1/3}T^{2/3}$. Further combining this with Theorem 8.17 that connects the regret guarantees of regression with the regret for our CONTEXTUAL PANDORA'S BOX algorithm, we obtain regret

$$2n(2c_{\max}M\max\{M, c_{\max}\}^2)^{1/6}T^{5/6}.$$

□

8.A.7.7 Discussion on the Lower Bounds

In this work our goal was to formulate the online contextual extension of the Pandora's Box problem and design no-regret algorithms for this problem, and therefore we left the lower bounds as a future work direction. We are however including here a brief discussion on lower bounds implied by previous work.

In Gatmiry et al. (2024) the authors study a special case of our setting where the value distributions and contexts are fixed at every round and for each alternative. The results of Gatmiry et al. (2024) imply a lower bound for this very special case of our problem, which however does not correspond to an adversarial setting, as our general formulation does. Notice, also, that a $\sqrt{T}$ lower bound for our problem can be directly obtained by the fact that our setting generalizes the stochastic multi-armed bandit problem. Observe that if all costs are chosen to be identical and large enough, then both the player and the optimal solution must select exactly one alternative per round (and their inspection costs cancel out in the regret). Interestingly, in that case Weitzman's algorithm indeed selects the alternative of the smallest mean reward.

In another related setting, multi-armed bandits with paid observations, Seldin et al. (2014) show a $T^{2/3}$ lower bound on the regret in the adversarial case. Although their setting is different (e.g. in our setting multiple actions are allowed at each round and there is contextual information involved) we believe that it has similarities to ours in terms of costly options and information acquisition. Therefore, we believe that tighter lower bounds could hold for our general adversarial problem.

9 NOISY OBSERVATIONS

9.1 Noisy Decision Tree

In this chapter we present an approximation for the mixture of distributions model and connect it to Noisy Decision Tree.

Having established the general purpose reductions between PANDORA's Box and Decision Tree(DT) in Chapter 5, we turn to the mixture of product distributions model of correlation. This special case of PANDORA's Box interpolates between Weitzman's independent values setting and the fully general correlated values setting. In this setting, we use the term "scenario" to denote the different product distributions in the mixture. The information gathering component of the problem is now about determining which product distribution in the mixture the box values are realized from. Once the algorithm has determined the realized scenario (a.k.a. product distribution), the remaining problem amounts to implementing Weitzman's strategy for that scenario.

We observe that this model of correlation for PANDORA's Box is related to the noisy version of DT, where the results of some tests for a given realized hypothesis are not deterministic. One challenge for DT in this setting is that any individual test may give us very little information distinguishing different scenarios, and one needs to combine information across sequences of many tests in order to isolate scenarios. This challenge is inherited by PANDORA's Box.

Previous work on noisy DT obtained algorithms whose approximations and run-times depend on the amount of noise. In contrast, we consider settings where the level of noise is arbitrary, but where the mixtures satisfy a *separability assumption*. In particular, we assume that for any given box, if we consider the marginal distributions of the value in the box under different scenarios, these distributions are either identical or sufficiently different (e.g., at least ε in TV distance) across different scenarios. Under this assumption, we design a constant-factor approximation for PANDORA's Box that runs in $n^{\tilde{O}(m^2/\varepsilon^2)}$ (Theorem 9.1), where n is the number of boxes.

Related work The noisy version of optimal decision tree was first studied in Golovin et al. (2010)[1], which gave an algorithm with runtime that depends exponentially on

[1] This result is based on a result from Golovin and Krause (2011) which turned out to be wrong (Nan and Saligrama, 2017). The correct results are presented in Golovin and Krause (2017)

the number of noisy outcomes. Subsequently, Jia et al. (2019) gave an $(\min(r, h) + \log m)$-approximation algorithm, where r (resp. h) is the maximum number of different test results per test (resp. scenario) using a reduction to Adaptive Submodular Ranking problem (Kambadur et al., 2017). In the case of large number of noisy outcome they obtain a $\log m$ approximation exploiting the connection to Stochastic Set Cover (Liu et al., 2008; Im et al., 2016).

Model & Definitions We describe again for convenience the explicit distributions setting, also described in Section 5.1. In the Pandora's Box problem we are given n boxes, each with cost $c_i \geqslant 0$ and value v_i. The values $\{v_i\}_{i\in[n]}$ are distributed according to known distribution $\mathcal{D}$. We assume that $\mathcal{D}$ is an arbitrary correlated distribution over vectors $\{v_i\}_{i\in[n]} \in \mathbb{R}^n$. We call vectors of values *scenarios* and use $s = \{v_i\}_{i\in[n]}$ to denote a possible realization of the scenario. As in DT, nature picks a scenario from the distribution D and this realization is a priori unknown to the algorithm. The goal of the algorithm is to pick a box of small value. The algorithm can observe the values realized in the boxes by opening any box i at its respective costs c_i.

Output. The output of the algorithm is an adaptive policy π for opening boxes and a stopping condition. The policy π takes as input a subset of the boxes and their associated values, and either returns the index of a box $i \in [n]$ to be opened next or stops and selects the minimum value seen so far. That is, $\pi : \cup_{X\subseteq[n]}\mathbb{R}^X \to [n] \cup \{\perp\}$ where $\perp$ denotes stopping.

Objective. For a given policy π, let $\pi(s)$ denote the set of boxes opened by the policy prior to stopping when the realized scenario is s. The objective of the algorithm is to minimize the expected cost of the boxes opened plus the minimum value discovered, where the expectation is taken over all possible realizations of the values in each box. Formally the objective is given by

$$\mathbf{E}_{s\sim\mathcal{D}}\left[\min_{i\in\pi(s)} v_{is} + \sum_{i\in\pi(s)} c_i\right],$$

In this chapter we consider a more general setting, where $\mathcal{D}$ is a mixture of m product distributions. Specifically, each scenario j is a product distribution; instead of giving a deterministic value for every box i, the result is drawn from distribution $\mathcal{D}_{ij}$. This setting is a generalization of the explicit distributions setting described before.

9.2 Mixture of Product Distributions

In this section we switch gears and consider the case where we are given a mixture of m product distributions. Observe that using the tool described in Section 5.3.1.1, we can reduce this problem to $PB_{\leqslant T}$. This now is equivalent to the noisy version of DT (Golovin and Krause, 2017; Jia et al., 2019) where for a specific scenario, the result of each test is not deterministic and can get different values with different probabilities.

Comparison with previous work: previous work on noisy decision tree, considers limited noise models or the runtime and approximation ratio depends on the type of noise. For example in the main result of Jia et al. (2019), the noise outcomes are binary with equal probability. The authors mention that it is possible to extend the following ways:

- to probabilities within $[\delta, 1-\delta]$, incurring an extra $1/\delta$ factor in the approximation
- to non-binary noise outcomes, incurring an extra at most m factor in the approximation

Additionally, their algorithm works by expanding the scenarios for every possible noise outcome (e.g. to 2^m for binary noise). In our work the number of noisy outcomes does not affect the number of scenarios whatsoever.

In our work, we obtain a **constant approximation** factor, that does not depend in any way on the type of the noise. Additionally, the outcomes of the noisy tests can be arbitrary, and do not affect either the approximation factor or the runtime. We only require a *separability* condition to hold ; the distributions either differ *enough* or are exactly the same. Formally, we require that for any two scenarios $s_1, s_2 \in \mathcal{S}$ and for every box i, the distributions $\mathcal{D}_{is_1}$ and $\mathcal{D}_{is_2}$ satisfy $|\mathcal{D}_{is_1} - \mathcal{D}_{is_2}| \in \mathbb{R}_{\geqslant \varepsilon} \cup \{0\}$, where $|\mathcal{A} - \mathcal{B}|$ is the total variation distance of distributions $\mathcal{A}$ and $\mathcal{B}$.

9.2.1 A DP Algorithm for noisy $\mathrm{PB}_{\leqslant T}$

We move on to designing a dynamic programming algorithm to solve the $\mathrm{PB}_{\leqslant T}$ problem, in the case of a mixtures of product distributions. The guarantees of our dynamic programming algorithm are given in the following theorem.

Theorem 9.1. *For any $\beta > 0$, let π_{DP} and π^* be the policies produced by Algorithm* DP(β) *described by Equation* (9.1) *and the optimal policy respectively and* $\mathrm{UB} = \frac{m^2}{\varepsilon^2} \log \frac{m^2 T}{c_{\min} \beta}$. *Then it holds that*

$$c(\pi_{DP}) \leqslant (1 + \beta) c(\pi^*).$$

and the DP *runs in time n^{UB}, where n is the number of boxes and $c_{\min}$ is the minimum cost box.*

Using the reduction described in Section 5.3.1.1 and the previous theorem we can get a constant-approximation algorithm for the initial PB problem given a mixture of product distributions. Observe that in the reduction, for every instance of $\mathrm{PB}_{\leqslant T}$ it runs, the chosen threshold T satisfies that $T \leqslant (\beta + 1) c(\pi_T^*)/0.2$ where π_T^* is the optimal policy for the threshold T. The inequality holds since the algorithm for the threshold T is a $(\beta + 1)$ approximation and it covers 80% of the scenarios left (i.e. pays $0.2T$ for the rest). This is formalized in the following corollary.

Corollary 9.2. *Given an instance of PB on m scenarios, and the DP algorithm described in Equation* (9.1)*, then using Algorithm* 9 *we obtain an* $O(1)$*-approximation algorithm for PB that runs in* $n^{\tilde{O}(m^2/\varepsilon^2)}$.

Observe that the naive DP, that keeps track of all the boxes and possible outcomes, has space exponential in the number of boxes, which can be very large. In our DP, we exploit the separability property of the distributions by distinguishing the boxes in two different types based on a given set of scenarios. Informally, the *informative* boxes help us distinguish between two scenarios, by giving us enough TV distance, while the *non-informative* always have zero TV distance. The formal definition follows.

Definition 9.3 (Informative and non-informative boxes)**.** *Let $S \subseteq \mathcal{S}$ be a set of scenarios. Then we call a box k* informative *if there exist $s_i, s_j \in \mathcal{S}$ such that*

$$|\mathcal{D}_{k s_i} - \mathcal{D}_{k s_j}| \geqslant \varepsilon.$$

We denote the set of all informative *boxes by* $\text{IB}(S)$*. Similarly, the boxes for which the above does not hold are called* non-informative *and the set of these boxes is denoted by* $\text{NIB}(S)$*.*

Recursive calls of the DP: Our dynamic program chooses at every step one of the following options:

1. open an **informative** box: this step contributes towards *eliminating* improbable scenarios. From the definition of informative boxes, every time such a box is opened, it gives TV distance at least ε between at least two scenarios, making one of them more probable than the other. We show (Lemma 9.4) that it takes a finite amount of these boxes to decide, with high probability, which scenario is the one realized (i.e. eliminating all but one scenarios).

2. open a **non-informative** box: this is a greedy step; the best non-informative box to open next is the one that maximizes the probability of finding a value smaller than T. Given a set S of scenarios that are not yet eliminated, there is a unique next non-informative box which is best. We denote by $\text{NIB}^*(S)$ the function that returns this next best non-informative box. Observe that the non-informative boxes do not affect the greedy ordering of which is the next best, since they do not affect which scenarios are eliminated.

State space of the DP: the DP keeps track of the following three quantities:

1. **a list** M which consists of sets of informative boxes opened and numbers of non-informative ones opened in between the sets of informative ones. Specifically, M has the following form: $M = S_1|x_1|S_2|x_2|\dots|S_L|x_L$[2] where S_i is a set of informative boxes, and $x_i \in \mathbb{N}$ is the number of non-informative boxes opened exactly after the boxes in set S_i. We also denote by $\text{IB}(M)$ the informative boxes in the list M.

   In order to update M at every recursive call, we either append a new informative box b_i opened (denoted by $M|b_i$) or, when a non-informative box is opened, we add 1 at the end, denoted by $M + 1$.

2. **a list** E of m^2 tuples of integers (z_{ij}, t_{ij}), one for each pair of distinct scenarios (s_i, s_j) with $i, j \in [m]$. The number z_{ij} keeps track of the number of informative

[2]If b_i for $i \in [n]$ are boxes, the list M looks like this: $b_3 b_6 b_{13}|5|b_{42} b_1|6|b_2$

boxes between s_i and s_j that the value discovered had higher probability for scenario s_i, and the number t_{ij} is the total number of informative for scenarios s_i and s_j opened. Every time an informative box is opened, we increase the t_{ij} variables for the scenarios the box was informative and add 1 to the z_{ij} if the value discovered had higher probability in s_i. When a non-informative box is opened, the list remains the same.We denote this update by E^{++}.

3. **a list** S of the scenarios not yet eliminated. Every time an informative test is performed, and the list E updated, if for some scenario s_i there exists another scenario s_j such that $t_{ij} > 1/\varepsilon^2 \log(1/\delta)$ and $|z_{ij} - \mathbf{E}[z_{ij}|s_i]| \leqslant \varepsilon/2$ then s_j is removed from S, otherwise s_i is removed[3]. This update is denoted by S^{++}.

Base cases: if a value below T is found, the algorithm stops. The other base case is when $|S| = 1$, which means that the scenario realized is identified, we either take the outside option T or search the boxes for a value below T, whichever is cheapest. If the scenario is identified correctly, the DP finds the expected optimal for this scenario. We later show that we make a mistake only with low probability, thus increasing the cost only by a constant factor. We denote by $\text{Nat}(\cdot,\cdot,\cdot)$ the "nature's" move, where the value in the box we chose is realized, and $\text{Sol}(\cdot,\cdot,\cdot)$ is the minimum value obtained by opening boxes. The recursive formula is shown below.

$$
\text{Sol}(M, E, S) = \begin{cases} \min(T, c_{\text{NIB}^*(S)} + \text{Nat}(M{+}1, E, S)) & \text{if } |S| = 1 \\ \min\Big(T, \min\limits_{i \in \text{IB}(M)} (c_i + \text{Nat}(M|i, E, S)) & \\ \qquad\qquad , c_{\text{NIB}^*(S)} + \text{Nat}(M{+}1, E, S)\Big) & \text{else} \end{cases} \tag{9.1}
$$
$$
\text{Nat}(M, E, S) = \begin{cases} 0 & \text{if } v_{\text{last box opened}} \leqslant T \\ \text{Sol}(M, E^{++}, S^{++}) & \text{else} \end{cases}
$$

The final solution is $\text{DP}(\beta) = \text{Sol}(\emptyset, E^0, \mathcal{S})$, where E^0 is a list of tuples of the form $(0, 0)$, and in order to update S we set $\delta = \beta c_{\min}/(m^2 T)$.

Lemma 9.4. *Let $s_1, s_2 \in \mathcal{S}$ be any two scenarios. Then after opening $\frac{\log(1/\delta)}{\varepsilon^2}$ informative boxes, we can eliminate one scenario with probability at least $1 - \delta$.*

We defer the proof of this lemma and Theorem 9.1 to the Appendix that follows.

[3]This is the process of elimination in the proof of Lemma 9.4

9.A Appendix for Chapter 9

Lemma 9.4. *Let $s_1, s_2 \in \mathcal{S}$ be any two scenarios. Then after opening $\frac{\log(1/\delta)}{\varepsilon^2}$ informative boxes, we can eliminate one scenario with probability at least $1-\delta$.*

Proof. Let $s_1, s_2 \in \mathcal{S}$ be any two scenarios in the instance of PB and let v_i be the value returned by opening the i'th informative box, which has distributions $\mathcal{D}_{is_1}$ and $\mathcal{D}_{is_2}$ for scenarios s_1 and s_2 respectively. Then by the definition of informative boxes for every such box opened, there is a set of values v for which $\mathbf{Pr}_{\mathcal{D}_{is_1}}[v] \geqslant \mathbf{Pr}_{\mathcal{D}_{is_2}}[v]$ and a set for which the reverse holds. Denote these sets by $M_i^{s_1}$ and $M_i^{s_2}$ respectively. We also define the indicator variables $X_i^{s_1} = \mathbb{1}\{v_i \in M_i^{s_1}\}$. Define $\overline{X} = \sum_{i\in[k]} X_i^{s_1}/k$, and observe that $\mathbf{E}\left[\overline{X}|s_1\right] = \sum_{i\in[k]} \mathbf{Pr}\left[M_i^{s_1}\right]/k$. Since for every box we have an ε gap in TV distance between the scenarios s_1, s_2 we have that

$$\left|\mathbf{E}\left[\overline{X}|s_1\right] - \mathbf{E}\left[\overline{X}|s_2\right]\right| \geqslant \varepsilon,$$

therefore if $\left|\overline{X} - \mathbf{E}\left[\overline{X}|s_1\right]\right| \leqslant \varepsilon/2$ we conclude that scenario s_2 is eliminated, otherwise we eliminate scenario s_1. The probability of error is $\mathbf{Pr}_{\mathcal{D}_{is_1}}\left[\overline{X} - \mathbf{E}\left[\overline{X}|s_1\right] > \varepsilon/2\right] \leqslant e^{-2k(\varepsilon/2)^2}$, where we used Hoeffding's inequality since $X_i \in \{0,1\}$. Since we want the probability of error to be less than δ, we need to open $O\left(\frac{\log 1/\delta}{\varepsilon^2}\right)$ informative boxes. □

Proof of Theorem 9.1. We describe how to bound the final cost, and calculate the runtime of the DP. Denote by $L = m^2/\varepsilon^2 \log 1/\delta$ where we show that in order to get $(1+\beta)$-approximation we set $\delta = \frac{\beta c_{\min}}{m^2 T}$.

Cost of the final solution. Observe that the only case where the DP limits the search space is when $|S| = 1$. If the scenario is identified correctly, the DP finds the optimal solution by running the greedy order; every time choosing the box with the highest probability of a value below T[4].

In order to eliminate all scenarios but one, we should eliminate all but one of the m^2 pairs in the list E. From Lemma 9.4, and a union bound on all m^2 pairs, the probability of the last scenario being the wrong one is at most $m^2\delta$. By setting $\delta = \beta c_{\min}/(m^2 T)$, we get that the probability of error is at most $\beta c_{\min}/T$, in which case we pay at most T, therefore getting an extra $\beta c_{\min} \leqslant \beta c(\pi^*)$ factor.

[4]When there is only one scenario, this is exactly Weitzman's algorithm.

Runtime. The DP maintains a list M of sets of informative boxes opened, and numbers of non informative ones. Recall that M has the following form $M = S_1|x_1|S_2|x_2|\dots|S_k|x_k$, where $k \leqslant L$ from Lemma 9.4 and the fact that there are m^2 pairs in E. There are in total n boxes, and L "positions" for them, therefore the size of the state space is $\binom{n}{L} = O(n^L)$. There is also an extra n factor for searching in the list of informative boxes at every step of the recursion. Observe that the numbers of non-informative boxes also add a factor of at most n in the state space. The list E adds another factor at most n^{m^2}, and the list S a factor of 2^m making the total runtime to be $n^{\tilde{O}(m^2/\varepsilon^2)}$. $\square$

REFERENCES

Abbasi-Yadkori, Yasin, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. In *Advances in neural information processing systems 24: 25th annual conference on neural information processing systems 2011. proceedings of a meeting held 12-14 december 2011, granada, spain*, ed. John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, 2312–2320.

Abe, Naoki, and Philip M. Long. 1999. Associative reinforcement learning using linear probabilistic concepts. In *Proceedings of the sixteenth international conference on machine learning (ICML 1999), bled, slovenia, june 27 - 30, 1999*, ed. Ivan Bratko and Saso Dzeroski, 3–11. Morgan Kaufmann.

Abernethy, Jacob D., Elad Hazan, and Alexander Rakhlin. 2008. Competing in the dark: An efficient algorithm for bandit linear optimization. In *21st annual conference on learning theory - COLT 2008, helsinki, finland, july 9-12, 2008*, ed. Rocco A. Servedio and Tong Zhang, 263–274. Omnipress.

Adler, Micah, and Brent Heeringa. 2012. Approximating optimal binary decision trees. *Algorithmica* 62(3-4):1112–1121.

Agarwal, Alekh, Ofer Dekel, and Lin Xiao. 2010. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT 2010 - the 23rd conference on learning theory, haifa, israel, june 27-29, 2010*, ed. Adam Tauman Kalai and Mehryar Mohri, 28–40. Omnipress.

Ailon, Nir, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. 2006. Self-improving algorithms. In *Proceedings of the seventeenth annual ACM-SIAM symposium on discrete algorithms, SODA 2006, miami, florida, usa, january 22-26, 2006*, 261–270.

Alabi, Daniel, Adam Tauman Kalai, Katrina Ligett, Cameron Musco, Christos Tzamos, and Ellen Vitercik. 2019. Learning to prune: Speeding up repeated computations. In *Conference on learning theory, COLT 2019, 25-28 june 2019, phoenix, az, USA*, 30–33.

Aouad, Ali, Jingwei Ji, and Yaron Shaposhnik. 2020. The pandora's box problem with sequential inspections. *Available at SSRN 3726167*.

Atsidakou, Alexia, Constantine Caramanis, Evangelia Gergatsouli, Orestis Papadigenopoulos, and Christos Tzamos. 2024. Contextual pandora's box. In *Proceedings of the aaai conference on artificial intelligence*, vol. 38, 10944–10952.

Azar, Pablo Daniel, Robert Kleinberg, and S. Matthew Weinberg. 2014. Prophet inequalities with limited information. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on discrete algorithms, SODA 2014, portland, oregon, usa, january 5-7, 2014*, ed. Chandra Chekuri, 1358–1377. SIAM.

Azar, Yossi, and Iftah Gamzu. 2011. Ranking with submodular valuations. In *Proceedings of the twenty-second annual ACM-SIAM symposium on discrete algorithms, SODA 2011, san francisco, california, usa, january 23-25, 2011*, 1070–1079.

Azar, Yossi, Iftah Gamzu, and Xiaoxin Yin. 2009. Multiple intents re-ranking. In *Proceedings of the 41st annual ACM symposium on theory of computing, STOC 2009, bethesda, md, usa, may 31 - june 2, 2009*, 669–678.

Balcan, Maria-Florina, Travis Dick, Tuomas Sandholm, and Ellen Vitercik. 2018a. Learning to branch. In *Proceedings of the 35th international conference on machine learning, ICML 2018, stockholmsmässan, stockholm, sweden, july 10-15, 2018*, 353–362.

Balcan, Maria-Florina, Travis Dick, and Ellen Vitercik. 2018b. 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*, 603–614.

Balcan, Maria-Florina, Vaishnavh Nagarajan, Ellen Vitercik, and Colin White. 2017. 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*, 213–274.

Bansal, Nikhil, Jatin Batra, Majid Farhadi, and Prasad Tetali. 2023. On min sum vertex cover and generalized min sum set cover. *SIAM Journal on Computing* 52(2): 327–357. https://doi.org/10.1137/21M1434052.

Bansal, Nikhil, Anupam Gupta, and Ravishankar Krishnaswamy. 2010. 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*, 1539–1545.

Basiak, Mateusz, Marcin Bienkowski, and Agnieszka Tatarczuk. 2023. An improved deterministic algorithm for the online min-sum set cover problem. In *International workshop on approximation and online algorithms*, 45–58. Springer.

Bechtel, Curtis, Shaddin Dughmi, and Neel Patel. 2022. Delegated pandora's box. In *Proceedings of the 23rd acm conference on economics and computation*, 666–693.

Berger, Ben, Tomer Ezra, Michal Feldman, and Federico Fusco. 2023. Pandora's problem with combinatorial cost. In *Proceedings of the 24th acm conference on economics and computation*, 273–292. EC '23, New York, NY, USA: Association for Computing Machinery.

Bertsimas, Dimitris, and Jack Dunn. 2017. Optimal classification trees. *Mach. Learn.* 106(7):1039–1082.

Beyhaghi, Hedyeh, and Linda Cai. 2023a. Pandora's problem with nonobligatory inspection: Optimal structure and a PTAS. In *Proceedings of the 55th annual ACM symposium on theory of computing, STOC 2023, orlando, fl, usa, june 20-23, 2023*, ed. Barna Saha and Rocco A. Servedio, 803–816. ACM.

———. 2023b. Recent developments in pandora's box problem: Variants and applications. *SIGecom Exchanges* 21(1):20–34.

Beyhaghi, Hedyeh, and Robert Kleinberg. 2019. Pandora's problem with nonobligatory inspection. In *Proceedings of the 2019 ACM conference on economics and computation, EC 2019, phoenix, az, usa, june 24-28, 2019*, ed. Anna Karlin, Nicole Immorlica, and Ramesh Johari, 131–132. ACM.

Bienkowski, Marcin, and Marcin Mucha. 2023. An improved algorithm for online min-sum set cover. In *Proceedings of the aaai conference on artificial intelligence*, vol. 37, 6815–6822.

Blair, Charles E. 1985. Problem complexity and method efficiency in optimization (a. s. nemirovsky and d. b. yudin). *Siam Review* 27:264–265.

Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1989. Learnability and the vapnik-chervonenkis dimension. *J. ACM* 36(4):929–965.

Boodaghians, Shant, Federico Fusco, Philip Lazos, and Stefano Leonardi. 2020. Pandora's box problem with order constraints. In *EC '20: The 21st ACM conference on* *economics and computation, virtual event, hungary, july 13-17, 2020*, ed. Péter Biró, Jason D. Hartline, Michael Ostrovsky, and Ariel D. Procaccia, 439–458. ACM.

Bubeck, Sébastien, Michael B. Cohen, and Yuanzhi Li. 2018. Sparsity, variance and curvature in multi-armed bandits. In *Algorithmic learning theory, ALT 2018, 7-9 april 2018, lanzarote, canary islands, spain*, ed. Firdaus Janoos, Mehryar Mohri, and Karthik Sridharan, vol. 83 of *Proceedings of Machine Learning Research*, 111–127. PMLR.

Bubeck, Sébastien, and Ronen Eldan. 2015. The entropic barrier: a simple and optimal universal self-concordant barrier. In *Proceedings of the 28th conference on learning theory, COLT 2015, paris, france, july 3-6, 2015*, ed. Peter Grünwald, Elad Hazan, and Satyen Kale, vol. 40 of *JMLR Workshop and Conference Proceedings*, 279. JMLR.org.

Caramanis, Constantine, Paul Dütting, Matthew Faw, Federico Fusco, Philip Lazos, Stefano Leonardi, Orestis Papadigenopoulos, Emmanouil Pountourakis, and Rebecca Reiffenhäuser. 2022. Single-sample prophet inequalities via greedy-ordered selection. In *Proceedings of the 2022 ACM-SIAM symposium on discrete algorithms, SODA 2022, virtual conference / alexandria, va, usa, january 9 - 12, 2022*, ed. Joseph (Seffi) Naor and Niv Buchbinder, 1298–1325. SIAM.

Cesa-Bianchi, Nicolò, and Gábor Lugosi. 2006. *Prediction, learning, and games*.

Chakaravarthy, Venkatesan T., Vinayaka Pandit, Sambuddha Roy, Pranjal Awasthi, and Mukesh K. Mohania. 2011. Decision trees for entity identification: Approximation algorithms and hardness results. *ACM Trans. Algorithms* 7(2):15:1–15:22.

Chakaravarthy, Venkatesan T., Vinayaka Pandit, Sambuddha Roy, and Yogish Sabharwal. 2009. Approximating decision trees with multiway branches. In *Automata, languages and programming, 36th international colloquium, ICALP 2009, rhodes, greece, july 5-12, 2009, proceedings, part I*, ed. Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, vol. 5555 of *Lecture Notes in Computer Science*, 210–221. Springer.

Charikar, Moses, Ronald Fagin, Venkatesan Guruswami, Jon M. Kleinberg, Prabhakar Raghavan, and Amit Sahai. 2000. 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*, 582–591.

Chawla, Shuchi, Evangelia Gergatsouli, Jeremy McMahan, and Christos Tzamos. 2023. Approximating pandora's box with correlations. In *Approximation, randomization, and combinatorial optimization. algorithms and techniques, APPROX/RANDOM 2023, september 11-13, 2023, atlanta, georgia, USA*, ed. Nicole Megow and Adam D. Smith, vol. 275 of *LIPIcs*, 26:1–26:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Chawla, Shuchi, Evangelia Gergatsouli, Yifeng Teng, Christos Tzamos, and Ruimin Zhang. 2020. 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*, 1214–1225. IEEE.

Chen, Yuxin, S. Hamed Hassani, Amin Karbasi, and Andreas Krause. 2015a. 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*, 338–363.

Chen, Yuxin, Shervin Javdani, Amin Karbasi, J. Andrew Bagnell, Siddhartha S. Srinivasa, and Andreas Krause. 2015b. Submodular surrogates for value of information. In *Proceedings of the twenty-ninth AAAI conference on artificial intelligence, january 25-30, 2015, austin, texas, USA.*, 3511–3518.

Cicalese, Ferdinando, Tobias Jacobs, Eduardo Sany Laber, and Marco Molinaro. 2010. On greedy algorithms for decision trees. In *Algorithms and computation - 21st international symposium, ISAAC 2010, jeju island, korea, december 15-17, 2010, proceedings, part II*, ed. Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park, vol. 6507 of *Lecture Notes in Computer Science*, 206–217. Springer.

Clarkson, Kenneth L., Wolfgang Mulzer, and C. Seshadhri. 2010. 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*, 1546–1565.

Correa, José, Andrés Cristi, Boris Epstein, and José A Soto. 2024. Sample-driven optimal stopping: From the secretary problem to the iid prophet inequality. *Mathematics of Operations Research* 49(1):441–475.

Correa, José R., Andrés Cristi, Boris Epstein, and José A. Soto. 2020. The two-sided game of googol and sample-based prophet inequalities. In *Proceedings of the 2020*

ACM-SIAM symposium on discrete algorithms, SODA 2020, salt lake city, ut, usa, january 5-8, 2020, ed. Shuchi Chawla, 2066–2081. SIAM.

Correa, José R., Paul Dütting, Felix A. Fischer, and Kevin Schewior. 2019. Prophet inequalities for I.I.D. random variables from an unknown distribution. In *Proceedings of the 2019 ACM conference on economics and computation, EC 2019, phoenix, az, usa, june 24-28, 2019*, ed. Anna Karlin, Nicole Immorlica, and Ramesh Johari, 3–17. ACM.

Correa, José R., Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. 2018. Recent developments in prophet inequalities. *SIGecom Exchanges* 17(1): 61–70.

Dani, Varsha, Thomas P. Hayes, and Sham M. Kakade. 2008. Stochastic linear optimization under bandit feedback. In *21st annual conference on learning theory - COLT 2008, helsinki, finland, july 9-12, 2008*, ed. Rocco A. Servedio and Tong Zhang, 355–366. Omnipress.

Dasgupta, Sanjoy. 2004. Analysis of a greedy active learning strategy. In *Advances in neural information processing systems 17 [neural information processing systems, NIPS 2004, december 13-18, 2004, vancouver, british columbia, canada]*, 337–344.

Dinur, Irit, and David Steurer. 2014. Analytical approach to parallel repetition. In *Symposium on theory of computing, STOC 2014, new york, ny, usa, may 31 - june 03, 2014*, 624–633.

Doval, Laura. 2018. Whether or not to open pandora's box. *Journal of Economic Theory* 175:127–158.

Drygala, Marina, Sai Ganesh Nagarajan, and Ola Svensson. 2023. Online algorithms with costly predictions. In *Proceedings of the 26th international conference on artificial intelligence and statistics*, ed. Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, vol. 206 of *Proceedings of Machine Learning Research*, 8078–8101. PMLR.

Dumitriu, Ioana, Prasad Tetali, and Peter Winkler. 2003. On playing golf with two balls. *SIAM Journal on Discrete Mathematics* 16(4):604–615.

Esfandiari, Hossein, Mohammad Taghi Hajiaghayi, Brendan Lucier, and Michael Mitzenmacher. 2019. 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*, 1885–1892. AAAI Press.

Feige, Uriel, László Lovász, and Prasad Tetali. 2002. Approximating min-sum set cover. In *Approximation algorithms for combinatorial optimization, 5th international workshop, APPROX 2002, rome, italy, september 17-21, 2002, proceedings*, 94–107.

Feige, Uriel, László Lovász, and Prasad Tetali. 2004. Approximating min sum set cover. *Algorithmica* 40(4):219–234.

Flaxman, Abraham, Adam Tauman Kalai, and H. Brendan McMahan. 2005. 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*, 385–394. SIAM.

Foster, Dylan J., and Alexander Rakhlin. 2020. Beyond UCB: optimal and efficient contextual bandits with regression oracles. In *Proceedings of the 37th ICML 2020, 13-18 july 2020, virtual event*, vol. 119 of *Proceedings of Machine Learning Research*, 3199–3210. PMLR.

Fotakis, Dimitris, Thanasis Lianeas, Georgios Piliouras, and Stratis Skoulakis. 2020. Efficient online learning of optimal rankings: Dimensionality reduction via gradient descent. In *Neurips 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*, ed. Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin.

Fu, Hao, Jian Li, and Pan Xu. 2018. A PTAS for a Class of Stochastic Dynamic Programs. In *Icalp'18: 45th international colloquium on automata, languages, and programming, prague, czech republic*, 56:1–56:14.

Fu, Hu, Jiawei Li, and Daogao Liu. 2023. Pandora box problem with nonobligatory inspection: Hardness and approximation scheme. In *Proceedings of the 55th annual ACM symposium on theory of computing, STOC 2023, orlando, fl, usa, june 20-23, 2023*, ed. Barna Saha and Rocco A. Servedio, 789–802. ACM.

Fu, Hu, and Tao Lin. 2020. Learning utilities and equilibria in non-truthful auctions. *Advances in Neural Information Processing Systems* 33:14231–14242.

Garey, M. R., and Ronald L. Graham. 1974. Performance bounds on the splitting algorithm for binary testing. *Acta Informatica* 3:347–355.

Gatmiry, Khashayar, Thomas Kesselheim, Sahil Singla, and Yifan Wang. 2024. Bandit algorithms for prophet inequality and pandora's box. In *Proceedings of the 2024 annual acm-siam symposium on discrete algorithms (soda)*, 462–500. SIAM.

Gergatsouli, Evangelia, and Christos Tzamos. 2022. Online learning for min sum set cover and pandora's box. In *International conference on machine learning, ICML 2022, 17-23 july 2022, baltimore, maryland, USA*, ed. Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, vol. 162 of *Proceedings of Machine Learning Research*, 7382–7403. PMLR.

———. 2024. Weitzman's rule for pandora's box with correlations. *Advances in Neural Information Processing Systems* 36.

Gittins, J.C., and D.M. Jones. 1974. A dynamic allocation index for the sequential design of experiments. *Progress in Statistics* 241–266.

Gittins, John, Kevin Glazebrook, and Richard Weber. 2011. *Multi-armed bandit allocation indices*. John Wiley & Sons.

Golovin, Daniel, and Andreas Krause. 2011. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res.* 42: 427–486.

———. 2017. Adaptive submodularity: A new approach to active learning and stochastic optimization. *CoRR* abs/1003.3967. 1003.3967.

Golovin, Daniel, Andreas Krause, and Debajyoti Ray. 2010. Near-optimal bayesian active learning with noisy observations. In *Advances in neural information processing systems 23: 24th annual conference on neural information processing systems 2010. proceedings of a meeting held 6-9 december 2010, vancouver, british columbia, canada*, ed. John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, 766–774. Curran Associates, Inc.

Grötschel, Martin, László Lovász, and Alexander Schrijver. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* 1(2): 169–197.

———. 1988. *Geometric algorithms and combinatorial optimization*, vol. 2 of *Algorithms and Combinatorics*. Springer.

Guha, Sudipto, Kamesh Munagala, and Saswati Sarkar. 2008. Information acquisition and exploitation in multichannel wireless networks. *CoRR* abs/0804.1724.

Guillory, Andrew, and Jeff A. Bilmes. 2009. Average-case active learning with costs. In *Algorithmic learning theory, 20th international conference, ALT 2009, porto, portugal, october 3-5, 2009. proceedings*, ed. Ricard Gavaldà, Gábor Lugosi, Thomas Zeugmann, and Sandra Zilles, vol. 5809 of *Lecture Notes in Computer Science*, 141–155. Springer.

Guo, Chenghao, Zhiyi Huang, Zhihao Gavin Tang, and Xinzhi Zhang. 2021. Generalizing complex hypotheses on product distributions: Auctions, prophet inequalities, and pandora's problem. In *Conference on learning theory, COLT 2021, 15-19 august 2021, boulder, colorado, USA*, ed. Mikhail Belkin and Samory Kpotufe, vol. 134 of *Proceedings of Machine Learning Research*, 2248–2288. PMLR.

Gupta, Anupam, Haotian Jiang, Ziv Scully, and Sahil Singla. 2019. 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*, 233–246.

Gupta, Anupam, and Amit Kumar. 2001. 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*, 416–425.

Gupta, Anupam, Viswanath Nagarajan, and R. Ravi. 2017. Approximation algorithms for optimal decision trees and adaptive TSP problems. *Math. Oper. Res.* 42(3): 876–896.

Gupta, Rishi, and Tim Roughgarden. 2017. A PAC approach to application-specific algorithm selection. *SIAM J. Comput.* 46(3):992–1017.

Hazan, Elad, Zohar Karnin, and Raghu Meka. 2014. Volumetric spanners: an efficient exploration basis for learning. In *Proceedings of the 27thcolt*, ed. Maria Florina Balcan, Vitaly Feldman, and Csaba Szepesvári, vol. 35 of *Proceedings of Machine Learning Research*, 408–422. Barcelona, Spain: PMLR.

Hill, Theodore P., and Robert P. Kertz. 1992. A survey of prophet inequalities in optimal stopping theory. *Contemporary Mathematics* 125.

van der Hoeven, Dirk, Tim van Erven, and Wojciech Kotlowski. 2018. The many faces of exponential weights in online learning. In *Conference on learning theory, COLT 2018, stockholm, sweden, 6-9 july 2018*, ed. Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, vol. 75 of *Proceedings of Machine Learning Research*, 2067–2092. PMLR.

Holzer, Harry J. 1988. Search method use by unemployed youth. *Journal of labor economics* 6(1):1–20.

Hughes, Ifan, and Thomas Hase. 2010. *Measurements and their uncertainties: a practical guide to modern error analysis*. OUP Oxford.

Hyafil, Laurent, and Ronald L. Rivest. 1976. Constructing optimal binary decision trees is np-complete. *Inf. Process. Lett.* 5(1):15–17.

Im, Sungjin, Viswanath Nagarajan, and Ruben van der Zwaan. 2016. Minimum latency submodular cover. *ACM Trans. Algorithms* 13(1):13:1–13:28.

Im, Sungjin, Maxim Sviridenko, and Ruben Van Der Zwaan. 2014. Preemptive and non-preemptive generalized min sum set cover. *Mathematical Programming* 145(1-2): 377–401.

Jia, Su, Viswanath Nagarajan, Fatemeh Navidi, and R. Ravi. 2019. Optimal decision tree with noisy outcomes. In *Advances in neural information processing systems 32: Annual conference on neural information processing systems 2019, neurips 2019, december 8-14, 2019, vancouver, bc, canada*, ed. Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, 3298–3308.

Kambadur, Prabhanjan, Viswanath Nagarajan, and Fatemeh Navidi. 2017. Adaptive submodular ranking. In *Integer programming and combinatorial optimization - 19th international conference, IPCO 2017, waterloo, on, canada, june 26-28, 2017, proceedings*, 317–329.

Karlin, Anna R., Mark S. Manasse, Lyle A. McGeoch, and Susan S. Owicki. 1990. 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*, ed. David S. Johnson, 301–309. SIAM.

Karni, Edi, and Aba Schwartz. 1977. Search theory: The case of search with uncertain recall. *Journal of Economic Theory* 16(1):38–52.

Ke, T. Tony, and J. Miguel Villas-Boas. 2019. Optimal learning before choice. *J. Econ. Theory* 180:383–437.

Kleinberg, Robert, Kevin Leyton-Brown, and Brendan Lucier. 2017. 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*, 2023–2031.

Kleinberg, Robert, Bo Waggoner, and E. Glen Weyl. 2016. Descending price optimally coordinates search. In *Proceedings of the 2016 acm conference on economics and computation*, 23–24. EC '16, New York, NY, USA: Association for Computing Machinery.

Kosaraju, S. Rao, Teresa M. Przytycka, and Ryan S. Borgstrom. 1999. On an optimal split tree problem. In *Algorithms and data structures, 6th international workshop, WADS '99, vancouver, british columbia, canada, august 11-14, 1999, proceedings*, ed. Frank K. H. A. Dehne, Arvind Gupta, Jörg-Rüdiger Sack, and Roberto Tamassia, vol. 1663 of *Lecture Notes in Computer Science*, 157–168. Springer.

Lattimore, Tor, and Csaba Szepesvári. 2020. *Bandit algorithms*. Cambridge University Press.

Li, Ray, Percy Liang, and Stephen Mussmann. 2020. A tight analysis of greedy yields subexponential time approximation for uniform decision tree. In *Proceedings of the 2020 ACM-SIAM symposium on discrete algorithms, SODA 2020, salt lake city, ut, usa, january 5-8, 2020*, ed. Shuchi Chawla, 102–121. SIAM.

Liu, Zhen, Srinivasan Parthasarathy, Anand Ranganathan, and Hao Yang. 2008. Near-optimal algorithms for shared filter evaluation in data stream systems. In *Proceedings of the ACM SIGMOD international conference on management of data, SIGMOD 2008, vancouver, bc, canada, june 10-12, 2008*, ed. Jason Tsong-Li Wang, 133–146. ACM.

Loveland, Donald W. 1985. Performance bounds for binary testing with arbitrary weights. *Acta Informatica* 22(1):101–114.

Lucier, Brendan. 2017. An economic view of prophet inequalities. *SIGecom Exchanges* 16(1):24–47.

Ma, Mingchen, and Christos Tzamos. 2023. Buying information for stochastic optimization. In *International conference on machine learning, ICML 2023, 23-29 july 2023, honolulu, hawaii, USA*, ed. Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, vol. 202 of *Proceedings of Machine Learning Research*, 23388–23411. PMLR.

McCall, Brian, and John McCall. 2007. *The economics of search*. Routledge.

McCall, John Joseph. 1970. Economics of information and job search. *The Quarterly Journal of Economics* 84(1):113–126.

Miller, Robert A. 1984. Job matching and occupational choice. *Journal of Political economy* 92(6):1086–1120.

Moffat, Robert J. 1988. Describing the uncertainties in experimental results. *Experimental thermal and fluid science* 1(1):3–17.

Moorthy, Sridhar, Brian T Ratchford, and Debabrata Talukdar. 1997. Consumer information search revisited: Theory and empirical analysis. *Journal of consumer research* 23(4):263–277.

Mortensen, Dale T. 1986. Job search and labor market analysis. *Handbook of labor economics* 2:849–919.

Munagala, Kamesh, Shivnath Babu, Rajeev Motwani, and Jennifer Widom. 2005. The pipelined set cover problem. In *Database theory - ICDT 2005, 10th international conference, edinburgh, uk, january 5-7, 2005, proceedings*, ed. Thomas Eiter and Leonid Libkin, vol. 3363 of *Lecture Notes in Computer Science*, 83–98. Springer.

Myerson, Roger B. 1981. Optimal auction design. *Math. Oper. Res.* 6(1):58–73.

Nan, Feng, and Venkatesh Saligrama. 2017. Comments on the proof of adaptive stochastic set cover based on adaptive submodularity and its implications for the group identification problem in "group-based active query selection for rapid diagnosis in time-critical situations". *IEEE Trans. Inf. Theory* 63(11):7612–7614.

Pattipati, Krishna R., and Mahesh Dontamsetty. 1992. On a generalized test sequencing problem. *IEEE Trans. Syst. Man Cybern.* 22(2):392–396.

Podgorelec, Vili, Peter Kokol, Bruno Stiglic, and Ivan Rozman. 2002. Decision trees: An overview and their use in medicine. *Journal of medical systems* 26:445–63.

Quan, Daniel C, and John M Quigley. 1991. Price formation and the appraisal function in real estate markets. *The Journal of Real Estate Finance and Economics* 4: 127–146.

Rabinovich, Semyon G. 2006. *Measurement errors and uncertainties: theory and practice*. Springer Science & Business Media.

Ratchford, Brian T. 1982. Cost-benefit models for explaining consumer choice and information seeking behavior. *Management Science* 28(2):197–212.

Rokach, Lior, and Oded Maimon. 2014. *Data mining with decision trees - theory and applications. 2^{nd} edition*, vol. 81 of *Series in Machine Perception and Artificial Intelligence*. WorldScientific.

Rothschild, Michael. 1978. Models of market organization with imperfect information: A survey. In *Uncertainty in economics*, 459–491. Elsevier.

Roughgarden, Tim. 2021. *Beyond the worst-case analysis of algorithms*. Cambridge University Press.

Rubinstein, Aviad, Jack Z. Wang, and S. Matthew Weinberg. 2020. Optimal single-choice prophet inequalities from samples. In *11th innovations in theoretical computer science conference, ITCS 2020, january 12-14, 2020, seattle, washington, USA*, ed. Thomas Vidick, vol. 151 of *LIPIcs*, 60:1–60:10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Rusmevichientong, Paat, and John N. Tsitsiklis. 2010. Linearly parameterized bandits. *Math. Oper. Res.* 35(2):395–411.

Safavian, S.R., and D. Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics* 21(3):660–674.

Segev, Danny, and Sahil Singla. 2021. Efficient approximation schemes for stochastic probing and prophet problems. In *EC '21: The 22nd ACM conference on economics and* *computation, budapest, hungary, july 18-23, 2021*, ed. Péter Biró, Shuchi Chawla, and Federico Echenique, 793–794. ACM.

Seldin, Yevgeny, Peter Bartlett, Koby Crammer, and Yasin Abbasi-Yadkori. 2014. Prediction with limited advice and multiarmed bandits with paid observations. In *Proceedings of the 31st international conference on machine learning*, ed. Eric P. Xing and Tony Jebara, vol. 32 of *Proceedings of Machine Learning Research*, 280–287. Bejing, China: PMLR.

Shalev-Shwartz, Shai. 2012. Online learning and online convex optimization. *Found. Trends Mach. Learn.* 4(2):107–194.

Shalev-Shwartz, Shai, and Yoram Singer. 2007a. A primal-dual perspective of online learning algorithms. *Mach. Learn.* 69(2-3):115–142.

———. 2007b. A primal-dual perspective of online learning algorithms. *Mach. Learn.* 69(2-3):115–142.

Simonson, Itamar, Joel Huber, and John Payne. 1988. The relationship between prior brand knowledge and information acquisition order. *Journal of consumer research* 14(4):566–578.

Singla, Sahil. 2018. 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*, 2523–2532.

Skutella, Martin, and David P. Williamson. 2011. A note on the generalized min-sum set cover problem. *Oper. Res. Lett.* 39(6):433–436.

Stigler, George J. 1961. The economics of information. *Journal of political economy* 69(3):213–225.

Tsitsiklis, John N. 1994. A short proof of the gittins index theorem. *The Annals of Applied Probability* 194–199.

Valko, Michal, Rémi Munos, Branislav Kveton, and Tomáš Kocák. 2014. Spectral bandits for smooth graph functions. In *Proceedings of the 31th international conference on machine learning, ICML 2014, beijing, china, 21-26 june 2014*, vol. 32 of *JMLR Workshop and Conference Proceedings*, 46–54. JMLR.org.

Weber, Richard. 1992. On the gittins index for multiarmed bandits. *The Annals of Applied Probability* 1024–1033.

Weisz, Gellert, Andras Gyorgy, and Csaba Szepesvari. 2018. Leapsandbounds: A method for approximately optimal algorithm configuration. In *International conference on machine learning*, 5257–5265.

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

Whittle, Peter. 1980. Multi-armed bandits and the gittins index. *Journal of the Royal Statistical Society: Series B (Methodological)* 42(2):143–149.

Wilde, Louis L. 1980. The economics of consumer information acquisition. *Journal of Business* S143–S158.