Approximating Pandora's Box with Correlations

Shuchi Chawla
University of Texas – Austin, TX, USA

Evangelia Gergatsouli
University of Wisconsin – Madison, WI, USA

Jeremy McMahan
University of Wisconsin – Madison, WI, USA

Christos Tzamos
University of Wisconsin – Madison, WI, USA
University of Athens, Greece

Abstract

We revisit the classic Pandora's Box (PB) problem under correlated distributions on the box values. Recent work of [13] obtained constant approximate algorithms for a restricted class of policies for the problem that visit boxes in a fixed order. In this work, we study the complexity of approximating the optimal policy which may adaptively choose which box to visit next based on the values seen so far.

Our main result establishes an approximation-preserving equivalence of PB to the well studied Uniform Decision Tree (UDT) problem from stochastic optimization and a variant of the Min-Sum Set Cover (MSSC_f) problem. For distributions of support m, UDT admits a $\log m$ approximation, and while a constant factor approximation in polynomial time is a long-standing open problem, constant factor approximations are achievable in subexponential time [43]. Our main result implies that the same properties hold for PB and MSSC_f.

We also study the case where the distribution over values is given more succinctly as a mixture of m product distributions. This problem is again related to a noisy variant of the Optimal Decision Tree which is significantly more challenging. We give a constant-factor approximation that runs in time $n^{\tilde{O}(m^2/\varepsilon^2)}$ when the mixture components on every box are either identical or separated in TV distance by ε.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Pandora's Box, Min Sum Set Cover, stochastic optimization, approximation preserving reduction

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2023.26

Category APPROX

Related Version *Full Version*: `https://arxiv.org/pdf/2108.12976.pdf`

Funding This work was funded in part by NSF awards CCF-2225259 and CCF-2217069.

1 Introduction

Many everyday tasks involve making decisions under uncertainty; for example driving to work using the fastest route or buying a house at the best price. Although we don't know how the future outcomes of our current decisions will turn out, we can often use some prior information to facilitate the decision making process. For example, having driven on the possible routes to work before, we know which is usually the busiest one. It is also common in such cases that we can remove part of the uncertainty by paying some additional cost. This type of online decision making in the presence of costly information can be modeled as the so-called Pandora's Box problem, first formalized by Weitzman in [52]. In this

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023).
Editors: Nicole Megow and Adam D. Smith; Article No. 26; pp. 26:1–26:24
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

problem, the algorithm is given n alternatives called *boxes*, each containing a value from a known distribution. The exact value is not known, but can be revealed at a known *opening* cost specific to the box. The goal of the algorithm is to decide which box to open next and whether to select a value and stop, such that the total *opening cost plus the minimum value revealed* is minimized. In the case of independent distributions on the boxes' values, this problem has a very elegant and simple optimal solution, as described by Weitzman [52]: calculate an index for each box[1], open the boxes in increasing order of index, and stop when the expected gain is worse than the value already obtained.

Weitzman's model makes the crucial assumption that the distributions on the values are independent across boxes. This, however, is not always the case in practice and as it turns out, the simple algorithm of the independent case fails to find the optimal solution under correlated distributions. Generally, the complexity of the Pandora's Box with correlations is not yet well understood. **In this work we develop the first approximately-optimal policies for the Pandora's Box problem with correlated values.**

We consider two standard models of correlation where the distribution over values can be specified explicitly in a succinct manner. In the first, the distribution over values has a small support of size m. In the second the distribution is a mixture of m product distributions, each of which can be specified succinctly. We present approximations for both settings.

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.

Contribution 1: Connection to Decision Tree and a general purpose approximation

Some aspects of the Pandora's Box problem have been studied separately in other contexts. For example, in the Optimal Decision Tree problem (DT) [30, 43], 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 [21], 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

[1] This is a special case of Gittins index [25].

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 with Feedback (MSSC_f). These connections are summarized in Figure 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.

$$\underbrace{\mathrm{PB} \xleftrightarrow{\text{Section 4}} \mathrm{UMSSC}_f}_{\text{Log-log factors}} \underbrace{\xleftrightarrow{\text{Section 5}} \mathrm{UDT}}_{\text{Constant factors}}$$

Figure 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 with Feedback. 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.

Contribution 2: Approximation for the mixture of distributions model

Having established the general purpose reductions between Pandora's Box and DT, 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 runtimes 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 18), where n is the number of boxes. The formal result and the algorithm is presented in Section 6.

1.1 Related work

The Pandora's Box problem was first introduced by Weitzman in the Economics literature [52]. Since then, there has been a long line of research studying Pandora's Box and its many variants ; non-obligatory inspection [19, 8, 7, 22], with order constraints [37, 9], with correlation [13, 24], with combinatorial costs [6], competitive information design [18], delegated version [5], and finally in an online setting [20]. Multiple works also study the generalized setting where more information can be obtained for a price [12, 32, 15, 14] and in settings with more complex combinatorial constraints [50, 26, 33, 1, 35, 36, 31].

Chawla et al. [13] were the first to study Pandora's Box with correlated values, but they designed approximations relative to a simpler benchmark, namely the optimal performance achievable using a so-called *Partially Adaptive* strategy that cannot adapt the order in which it opens boxes to the values revealed. In general, optimal strategies can decide both the ordering of the boxes and the stopping time based on the values revealed. [13] designed an algorithm with performance no more than a constant factor worse than the optimal Partially Adaptive strategy.

In Min Sum Set Cover the line of work was initiated by [21], and continued with improvements and generalizations to more complex constraints by [3, 46, 4, 51].

Optimal decision tree is an old problem studied in a variety of settings ([49, 48, 30, 29]), while its most notable application is in active learning settings. It was proven to be NP-Hard by Hyafil and Rivest [38]. Since then the problem of finding the best approximation algorithm was an active one [23, 45, 42, 17, 10, 11, 30, 34, 16, 2], where finally a greedy $\log m$ for the general case was given by [30]. This approximation ratio is proven to be the best possible [10]. 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 [10]. The recent work of Li et al. [43] showed that there is an algorithm strictly better than $\log m$ for the uniform decision tree.

The noisy version of optimal decision tree was first studied in [29][2], which gave an algorithm with runtime that depends exponentially on the number of noisy outcomes. Subsequently, Jia et al. in [40] 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 [41]. In the case of large number of noisy outcome they obtain a $\log m$ approximation exploiting the connection to Stochastic Set Cover [44, 39].

[2] This result is based on a result from [27] which turned out to be wrong [47]. The correct results are presented in [28]

2 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 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 $c_i \geq 0$ 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.[3] Formally the objective is given by

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

For simplicity of presentation, from now on we assume that $c_i = 1$ for all boxes, but we show in the Appendix of the full version 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 $\mathbb{E}_{s\sim\mathcal{D}}[\text{cost}_s(\pi)]$.

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, $\pi : \cup_{X\subseteq[n]}\mathbb{R}^X \to [n]$.

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 $\mathbb{E}_{s\sim\mathcal{D}}[\text{cost}_s(\pi)]$.

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 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 PB and elements in MSSC_f;

[3] In the original version of the problem studied by Weitzman [52] 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.

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 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.

2.1 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.

Mixture of Distributions. We also 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.

3 Roadmap of the Reductions and Implications

In Figure 2, we give an overview of all the main technical reductions shown in Sections 4 and 5. 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).

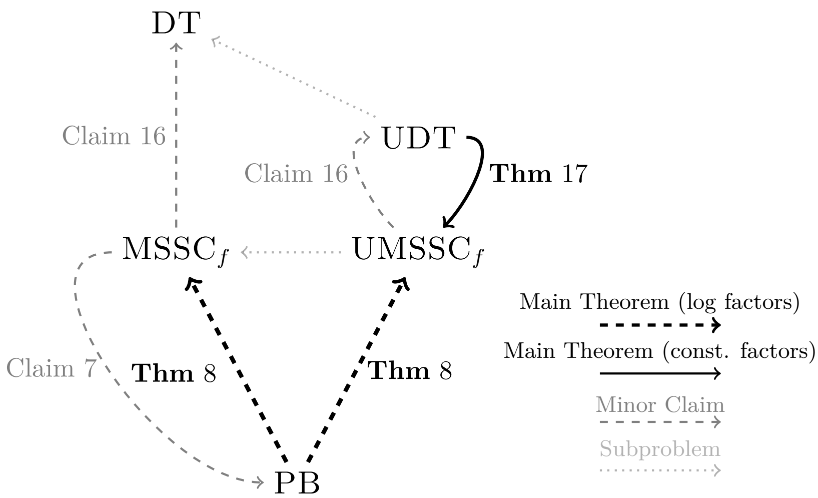

Figure 2 Summary of all our reductions. Bold black lines denote our main theorems, gray dashed are minor claims, and dotted lines are trivial reductions.

3.1 Approximating Pandora's Box

Given our reductions and using the best known results for Uniform Decision Tree from [43] we immediately obtain efficient approximation algorithms for Pandora's Box. We repeat the results of [43] below.

▶ **Theorem 2** (Theorems 3.1 and 3.2 from [43]).

- *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 2 combined with Theorem 8 and Claim 16 we get the following corollary.

▶ **Corollary 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*[4]*.*
- *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 4.** *It is not NP-hard to achieve any superconstant approximation for PB assuming the Exponential Time Hypothesis.*

Observe that the logarithmic approximation achieved in Corollary 3 loses a $\log\log m$ factor (hence the $\tilde{O}$) as it relies on the more complex reduction of Theorem 8. If we choose to use the more direct naive reduction (given in the full version of our paper) to the Optimal Decision Tree where the tests have non-unit costs (which also admits a $O(\log m)$-approximation [34, 41]), we get the following corollary.

▶ **Corollary 5.** *There exists an efficient algorithm that is* $O(\log m)$*-approximate for Pandora's Box and with or without unit-cost boxes.*

3.2 Constant approximation for Partially Adaptive PB

Moving on, we show how our reduction can be used to obtain and improve the results of [13]. Recall that in [13] 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 4 can be used to reduce PB to the standard Min Sum Set Cover (i.e. without feedback), which admits a 4-approximation [21].

▶ **Corollary 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 [46]. The case of non-uniform opening costs has also been considered for Pandora's Box by [13] but only provide an algorithm to handle polynomially bounded opening costs.

[4] 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 $\mathrm{OPT} \geq \log_K m$, as observed by [43].

4 Connecting Pandora's Box and MSSC$_f$

In this section we establish the connection between Pandora's Box and Min Sum Set Cover with Feedback. 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 with Feedback is a special case of Pandora's Box. Note that in both problems we examine boxes/elements in an adaptive order. In PB 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 PB and a PB algorithm can be used to solve it. We formally describe this reduction in Section A of the Appendix.

▷ Claim 7. If there exists an $\alpha(n, m)$-approximation algorithm for PB then there exists a $\alpha(n, m)$-approximation for MSSC$_f$.

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

▶ **Theorem 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 PB.*

Guessing a stopping rule and an intermediate problem

The feedback structure in 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 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 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 PB$_{\leq T}$. In this version the objective is to minimize the cost of finding a value $\leq T$, 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$_{\leq T}$ with a logarithmic loss in approximation factor, and then PB$_{\leq T}$ can be reduced to Min Sum Set Cover with Feedback with a constant factor loss. The following two results capture the details of these reductions.

▷ Claim 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$_{\leq T}$.

▶ **Main Lemma 10.** *Given a polynomial-time $\alpha(n,m)$-approximation algorithm for UPB$_{\leq T}$, there exists a polynomial-time $O(\alpha(n,m)\log\alpha(n,m))$-approximation for PB.*

The relationship between PB$_{\leq T}$ and MIN SUM SET COVER WITH FEEDBACK is relatively straightforward and requires explicitly relating the structure of feedback in the two problems. We describe the details in Section A of the Appendix.

Putting it all together. The proof of Theorem 8 follows by combining Claim 9 with Lemmas 11 and 10 presented in the following sections. Proofs of Claims 7, 9 deferred to Section A of the Appendix. The rest of this section is devoted to proving Lemmas 11 and 10.

4.1 Reducing Pandora's Box to PB$_{\leq T}$

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$_{\leq T}$ 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 WITH FEEDBACK in Section 4.1.1 and then improve upon that to reduce from the **uniform** version of MSSC$_f$ (Section 4.1.2).

4.1.1 Main Tool

The high level idea in this reduction is that we repeatedly run the algorithm for PB$_{\leq T}$ 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 the full version of the paper. Formally, we show the following lemma.

▶ **Main Lemma 11.** *Given a polynomial-time $\alpha(n,m)$-approximation algorithm for PB$_{\leq T}$, there exists a polynomial-time $O(\alpha(n,m)\log\alpha(n,m))$-approximation for PB.*

Algorithm 1 Reduction from PB to PB$_{\leq T}$.

Input: Oracle $\mathcal{A}(T)$ for PB$_{\leq T}$, set of all scenarios $\mathcal{S}$.
$i \leftarrow 0$ `// Number of current Phase`
while $\mathcal{S} \neq \emptyset$ **do**
  Use $\mathcal{A}$ to find smallest T_i via Binary Search s.t. **Pr** [accepting the outside option T_i] ≤ 0.2
  Call the oracle $\mathcal{A}(T_i)$ on set $\mathcal{S}$ to obtain policy π_i
  $\mathcal{S} \leftarrow \mathcal{S} \setminus$ {scenarios with total cost $\leq T_i$}
end
for $i \leftarrow 0$ *to* ∞ **do**
  Run policy π_i until it terminates and selects a box, or accumulates probing cost T_i.
end

We will now analyze the policy produced by this algorithm.

Proof of Main Lemma 11. We start with some notation. Given an instance $\mathcal{I}$ of PB, we repeatedly run $\text{PB}_{\leq T}$ in *phases*. Phase i consists of running $\text{PB}_{\leq T}$ 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[5], and run $\text{PB}_{\leq T}$ 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 1.

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 $\leq 2\sum_{i=0}^{\infty}(0.2)^i T_i$. 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 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*

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

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 lemmas are deferred to Section A.1 of the Appendix. We first formally define a *sub-instance* of the given Pandora's Box instance.

▶ **Definition 13** (Sub-instance). *Let $\mathcal{I}$ be an instance of $\{PB_{\leq T}, PB\}$ 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 14** (Optimal Lower Bound). *Let $\mathcal{I}$ be the instance of PB. For any $q < 1$, any $\alpha > 1$, and $\beta \geq 2$, for the optimal policy $\pi^*_{\mathcal{I}}$ for PB it that*

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

▶ **Lemma 15.** *Given an instance $\mathcal{I}$ of PB; an α-approximation algorithm $\mathcal{A}_T$ to $PB_{\leq T}$; and any $q < 1$ and $\beta \geq 2$, suppose that the threshold T satisfies*

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

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.

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

Calculating the thresholds. For every phase i we choose a threshold T_i such that $T_i = \min\{T : \mathbf{Pr}\left[c_s > T\right] \leq 0.2\}$ 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 $\text{PB}_{\leq T}$ 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 15 for $\beta = 10$, we know that for remaining total probability mass $(0.2)^i$, any threshold which satisfies

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

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

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

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 (1) to optimal costs of scenarios corresponding to intervals of the form $\text{Int}_i = [t_{(0.2)^i}, t_{(0.2)^i/(10\alpha)}]$. Note that these intervals are overlapping. We therefore get

$$\begin{aligned}
\text{cost}(\pi_{\mathcal{I}}) &\leq 2\sum_{i=0}^{\infty}(0.2)^i T_i \\
&= 2\sum_{i=0}^{\infty}\left((0.2)^i t_{(0.2)^{i-1}/10a} + 10\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in\text{Int}_i}} c_s p_s\right) && \text{From equation (1)} \\
&\leq 4\cdot 10\alpha\pi^*_{\mathcal{I}} + 20\alpha\sum_{i=0}^{\infty}\sum_{\substack{s\in\mathcal{S}\\ c_s\in\text{Int}_i}} c_s p_s && \text{Using Lemma 14 for } \beta = 10, q = 0.2 \\
&\leq 40\alpha\log\alpha\cdot\pi^*_{\mathcal{I}}.
\end{aligned}$$

Where 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.

4.1.2 An Even Stronger Tool

Moving one step further, we show that if we instead of $\text{PB}_{\leq T}$ we had an α-approximation algorithm for $\text{UPB}_{\leq T}$ we can obtain the same guarantees as the ones described in Lemma 11. Observe that we cannot directly use Algorithm 1 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 10.** *Given a polynomial-time $\alpha(n,m)$-approximation algorithm for $UPB_{\leq T}$, 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 11, and show how to change Algorithm 1 to work with the new oracle, that requires the scenarios to have uniform probability. The function **Expand** shown in Algorithm 2 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 3 in Section A.2 of the Appendix, alongside the proof of Main Lemma 10.

Algorithm 2 Reduction from PB to $\text{UPB}_{\leq T}$.

Input: Oracle $\mathcal{A}(T)$ for $\text{UPB}_{\leq T}$, set of all scenarios $\mathcal{S}$, $c = 1/10, \delta = 0.1$.

$i \leftarrow 0$ `// Number of current Phase`
while $\mathcal{S} \neq \emptyset$ **do**
  Let $\mathcal{L} = \left\{ s \in \mathcal{S} : p_s \leq c \cdot \frac{1}{|\mathcal{S}|} \right\}$ `// Remove low probability scenarios`
  $\mathcal{S}' = \mathcal{S} \setminus \mathcal{L}$
  $\mathcal{UI} = \text{Expand}(\mathcal{S}')$
  In instance $\mathcal{UI}$ use $\mathcal{A}$ to find smallest T_i via Binary Search s.t. $\mathbf{Pr}[\text{accepting } T_i] \leq \delta$
  Call the oracle $\mathcal{A}(T_i)$
  $\mathcal{S} \leftarrow \left(\mathcal{S}' \setminus \{ s \in \mathcal{S}' : c_s \leq T_i \} \right) \cup \mathcal{L}$
end

5 Connecting MSSC_f and Optimal Decision Tree

In this section we establish the connection between Min Sum Set Cover with Feedback and Optimal Decision Tree. We show that the uniform versions of these problems are equivalent up to constant factors in approximation ratio. The proofs of this section are deferred to the full version of the paper in ArXiv.

▷ Claim 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 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 the full version, 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 16.

Proof Sketch of Theorem 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.

6 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 4.1.1, we can reduce this problem to $\mathrm{PB}_{\leq T}$. This now is equivalent to the noisy version of DT [28, 40] 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 [40], 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 $|\mathcal{D}_{is_1} - \mathcal{D}_{is_2}| \in \mathbb{R}_{\geq \varepsilon} \cup \{0\}$, where $|\mathcal{A} - \mathcal{B}|$ is the total variation distance of distributions $\mathcal{A}$ and $\mathcal{B}$.

6.1 A DP Algorithm for noisy PB$_{\leq T}$

We move on to designing a dynamic programming algorithm to solve the PB$_{\leq T}$ problem, in the case of a mixtures of product distributions. The guarantees of our dynamic programming algorithm are given in the following theorem.

▶ **Theorem 18.** *For any $\beta > 0$, let π_{DP} and π^* be the policies produced by Algorithm* DP(β) *described by Equation* (2) *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*

$$c(\pi_{\mathrm{DP}}) \leq (1+\beta) c(\pi^*).$$

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 4.1.1 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 PB$_{\leq T}$ it runs, the chosen threshold T satisfies that $T \leq (\beta+1) c(\pi_T^*)/0.2$ 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 19.** *Given an instance of PB on m scenarios, and the DP algorithm described in Equation* (2)*, then using Algorithm 1 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 20** (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*

$$|\mathcal{D}_{ks_i} - \mathcal{D}_{ks_j}| \geq \varepsilon.$$

We denote the set of all informative *boxes by* 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* 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 21) 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 $\mathrm{NIB}^*(S)$ 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 $\boldsymbol{M}$** 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: $M = S_1|x_1|S_2|x_2|\dots|S_L|x_L$[6] 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 $\mathrm{IB}(M)$ 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 $\boldsymbol{E}$** 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 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 $\boldsymbol{S}$** 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 $|z_{ij} - \mathbb{E}[z_{ij}|s_i]| \le \varepsilon/2$ then s_j is removed from S, otherwise s_i is removed[7]. 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 $\mathrm{Nat}(\cdot,\cdot,\cdot)$ the "nature's" move, where the value in the box we chose is realized, and $\mathrm{Sol}(\cdot,\cdot,\cdot)$ is the minimum value obtained by opening boxes. The recursive formula is shown below.

$$\mathrm{Sol}(M,E,S) = \begin{cases} \min(T, c_{\mathrm{NIB}^*(S)} + \mathrm{Nat}(M{+}1, E, S)) & \text{if } |S| = 1 \\ \min\Big(T, \min\limits_{i \in \mathrm{IB}(M)} (c_i + \mathrm{Nat}(M|i, E, S)) & \\ \qquad , c_{\mathrm{NIB}^*(S)} + \mathrm{Nat}(M{+}1, E, S)\Big) & \text{else} \end{cases}$$

[6] If b_i for $i \in [n]$ are boxes, the list M looks like this: $b_3 b_6 b_{13}|5|b_{42} b_1|6|b_2$
[7] This is the process of elimination in the proof of Lemma 21

$$\mathrm{Nat}(M,E,S) = \begin{cases} 0 & \text{if } v_{\text{last box opened}} \le T \\ \mathrm{Sol}(M,E^{++},S^{++}) & \text{else} \end{cases} \tag{2}$$

The final solution is $\mathrm{DP}(\beta) = \mathrm{Sol}(\emptyset, E^0, \mathcal{S})$, where E^0 is a list of tuples of the form $(0,0)$, and in order to update S we set $\delta = \beta c_{\min}/(m^2T)$.

▶ **Lemma 21.** *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 $1-\delta$.*

References

1 Marek Adamczyk, Maxim Sviridenko, and Justin Ward. Submodular stochastic probing on matroids. *Math. Oper. Res.*, 41(3):1022–1038, 2016. doi:10.1287/moor.2015.0766.

2 Micah Adler and Brent Heeringa. Approximating optimal binary decision trees. *Algorithmica*, 62(3-4):1112–1121, 2012. doi:10.1007/s00453-011-9510-9.

3 Yossi Azar, Iftah Gamzu, and Xiaoxin Yin. Multiple intents re-ranking. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 – June 2, 2009*, pages 669–678. ACM, 2009. doi:10.1145/1536414.1536505.

4 Nikhil Bansal, Anupam Gupta, and Ravishankar Krishnaswamy. 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*, pages 1539–1545, 2010. doi:10.1137/1.9781611973075.125.

5 Curtis Bechtel, Shaddin Dughmi, and Neel Patel. Delegated pandora's box. In David M. Pennock, Ilya Segal, and Sven Seuken, editors, *EC '22: The 23rd ACM Conference on Economics and Computation, Boulder, CO, USA, July 11 – 15, 2022*, pages 666–693. ACM, 2022. doi:10.1145/3490486.3538267.

6 Ben Berger, Tomer Ezra, Michal Feldman, and Federico Fusco. Pandora's problem with combinatorial cost. *CoRR*, abs/2303.01078, 2023. doi:10.48550/arXiv.2303.01078.

7 Hedyeh Beyhaghi and Linda Cai. Pandora's problem with nonobligatory inspection: Optimal structure and a PTAS. *CoRR*, abs/2212.01524, 2022. doi:10.48550/arXiv.2212.01524.

8 Hedyeh Beyhaghi and Robert Kleinberg. Pandora's problem with nonobligatory inspection. In Anna Karlin, Nicole Immorlica, and Ramesh Johari, editors, *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019*, pages 131–132. ACM, 2019. doi:10.1145/3328526.3329626.

9 Shant Boodaghians, Federico Fusco, Philip Lazos, and Stefano Leonardi. Pandora's box problem with order constraints. In Péter Biró, Jason D. Hartline, Michael Ostrovsky, and Ariel D. Procaccia, editors, *EC '20: The 21st ACM Conference on Economics and Computation, Virtual Event, Hungary, July 13-17, 2020*, pages 439–458. ACM, 2020. doi:10.1145/3391403.3399501.

10 Venkatesan T. Chakaravarthy, Vinayaka Pandit, Sambuddha Roy, Pranjal Awasthi, and Mukesh K. Mohania. Decision trees for entity identification: Approximation algorithms and hardness results. *ACM Trans. Algorithms*, 7(2):15:1–15:22, 2011. doi:10.1145/1921659.1921661.

11 Venkatesan T. Chakaravarthy, Vinayaka Pandit, Sambuddha Roy, and Yogish Sabharwal. Approximating decision trees with multiway branches. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, volume 5555 of *Lecture Notes in Computer Science*, pages 210–221. Springer, 2009. doi:10.1007/978-3-642-02927-1_19.

12 Moses Charikar, Ronald Fagin, Venkatesan Guruswami, Jon M. Kleinberg, Prabhakar Raghavan, and Amit Sahai. 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*, pages 582–591, 2000. doi:10.1145/335305.335382.

13 Shuchi Chawla, Evangelia Gergatsouli, Yifeng Teng, Christos Tzamos, and Ruimin Zhang. 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*, pages 1214–1225. IEEE, 2020. doi:10.1109/FOCS46700.2020.00116.

14 Yuxin Chen, S. Hamed Hassani, Amin Karbasi, and Andreas Krause. 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*, pages 338–363, 2015. URL: http://proceedings.mlr.press/v40/Chen15b.html.

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

16 Ferdinando Cicalese, Tobias Jacobs, Eduardo Sany Laber, and Marco Molinaro. On greedy algorithms for decision trees. In Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park, editors, *Algorithms and Computation – 21st International Symposium, ISAAC 2010, Jeju Island, Korea, December 15-17, 2010, Proceedings, Part II*, volume 6507 of *Lecture Notes in Computer Science*, pages 206–217. Springer, 2010. doi:10.1007/978-3-642-17514-5_18.

17 Sanjoy Dasgupta. 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]*, pages 337–344, 2004. URL: https://proceedings.neurips.cc/paper/2004/hash/c61fbef63df5ff317aecdc3670094472-Abstract.html.

18 Bolin Ding, Yiding Feng, Chien-Ju Ho, Wei Tang, and Haifeng Xu. Competitive information design for pandora's box. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 353–381. SIAM, 2023. doi:10.1137/1.9781611977554.ch15.

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

20 Hossein Esfandiari, Mohammad Taghi Hajiaghayi, Brendan Lucier, and Michael Mitzenmacher. 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*, pages 1885–1892. AAAI Press, 2019. doi:10.1609/aaai.v33i01.33011885.

21 Uriel Feige, László Lovász, and Prasad Tetali. Approximating min sum set cover. *Algorithmica*, 40(4):219–234, 2004. doi:10.1007/s00453-004-1110-5.

22 Hu Fu, Jiawei Li, and Daogao Liu. Pandora box problem with nonobligatory inspection: Hardness and improved approximation algorithms. *CoRR*, abs/2207.09545, 2022. doi:10.48550/arXiv.2207.09545.

23 M. R. Garey and Ronald L. Graham. Performance bounds on the splitting algorithm for binary testing. *Acta Informatica*, 3:347–355, 1974. doi:10.1007/BF00263588.

24 Evangelia Gergatsouli and Christos Tzamos. Weitzman's rule for pandora's box with correlations. *CoRR*, abs/2301.13534, 2023. doi:10.48550/arXiv.2301.13534.

25 J.C. Gittins and D.M. Jones. A dynamic allocation index for the sequential design of experiments. *Progress in Statistics*, pages 241–266, 1974.

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

27 Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *J. Artif. Intell. Res.*, 42:427–486, 2011. doi:10.1613/jair.3278.

28 Daniel Golovin and Andreas Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. *CoRR*, abs/1003.3967, 2017. arXiv:1003.3967.

29 Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, editors, *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*, pages 766–774. Curran Associates, Inc., 2010. URL: https://proceedings.neurips.cc/paper/2010/hash/1e6e0a04d20f50967c64dac2d639a577-Abstract.html.

30 Andrew Guillory and Jeff A. Bilmes. Average-case active learning with costs. In Ricard Gavaldà, Gábor Lugosi, Thomas Zeugmann, and Sandra Zilles, editors, *Algorithmic Learning Theory, 20th International Conference, ALT 2009, Porto, Portugal, October 3-5, 2009. Proceedings*, volume 5809 of *Lecture Notes in Computer Science*, pages 141–155. Springer, 2009. doi:10.1007/978-3-642-04414-4_15.

31 Anupam Gupta, Haotian Jiang, Ziv Scully, and Sahil Singla. 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*, pages 233–246, 2019. doi:10.1007/978-3-030-17953-3_18.

32 Anupam Gupta and Amit Kumar. 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*, pages 416–425, 2001. doi:10.1109/SFCS.2001.959916.

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

34 Anupam Gupta, Viswanath Nagarajan, and R. Ravi. Approximation algorithms for optimal decision trees and adaptive TSP problems. *Math. Oper. Res.*, 42(3):876–896, 2017. doi:10.1287/moor.2016.0831.

35 Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1731–1747, 2016. doi:10.1137/1.9781611974331.ch120.

36 Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Adaptivity gaps for stochastic probing: Submodular and XOS functions. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1688–1702, 2017. doi:10.1137/1.9781611974782.111.

37 Noam Hazon, Yonatan Aumann, Sarit Kraus, and David Sarne. Physical search problems with probabilistic knowledge. *Artif. Intell.*, 196:26–52, 2013. doi:10.1016/j.artint.2012.12.003.

38 Laurent Hyafil and Ronald L. Rivest. Constructing optimal binary decision trees is np-complete. *Inf. Process. Lett.*, 5(1):15–17, 1976. doi:10.1016/0020-0190(76)90095-8.

39 Sungjin Im, Viswanath Nagarajan, and Ruben van der Zwaan. Minimum latency submodular cover. *ACM Trans. Algorithms*, 13(1):13:1–13:28, 2016. doi:10.1145/2987751.

40 Su Jia, Viswanath Nagarajan, Fatemeh Navidi, and R. Ravi. Optimal decision tree with noisy outcomes. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3298–3308, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/85f007f8c50dd25f5a45fca73cad64bd-Abstract.html.

41 Prabhanjan Kambadur, Viswanath Nagarajan, and Fatemeh Navidi. Adaptive submodular ranking. In *Integer Programming and Combinatorial Optimization – 19th International Conference, IPCO 2017, Waterloo, ON, Canada, June 26-28, 2017, Proceedings*, pages 317–329, 2017. doi:10.1007/978-3-319-59250-3_26.

42 S. Rao Kosaraju, Teresa M. Przytycka, and Ryan S. Borgstrom. On an optimal split tree problem. In Frank K. H. A. Dehne, Arvind Gupta, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Algorithms and Data Structures, 6th International Workshop, WADS '99, Vancouver, British Columbia, Canada, August 11-14, 1999, Proceedings*, volume 1663 of *Lecture Notes in Computer Science*, pages 157–168. Springer, 1999. doi:10.1007/3-540-48447-7_17.

43 Ray Li, Percy Liang, and Stephen Mussmann. A tight analysis of greedy yields subexponential time approximation for uniform decision tree. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 102–121. SIAM, 2020. doi:10.1137/1.9781611975994.7.

44 Zhen Liu, Srinivasan Parthasarathy, Anand Ranganathan, and Hao Yang. Near-optimal algorithms for shared filter evaluation in data stream systems. In Jason Tsong-Li Wang, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 133–146. ACM, 2008. doi:10.1145/1376616.1376633.

45 Donald W. Loveland. Performance bounds for binary testing with arbitrary weights. *Acta Informatica*, 22(1):101–114, 1985. doi:10.1007/BF00290148.

46 Kamesh Munagala, Shivnath Babu, Rajeev Motwani, and Jennifer Widom. The pipelined set cover problem. In Thomas Eiter and Leonid Libkin, editors, *Database Theory – ICDT 2005, 10th International Conference, Edinburgh, UK, January 5-7, 2005, Proceedings*, volume 3363 of *Lecture Notes in Computer Science*, pages 83–98. Springer, 2005. doi:10.1007/978-3-540-30570-5_6.

47 Feng Nan and Venkatesh Saligrama. 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, 2017. doi:10.1109/TIT.2017.2749505.

48 Krishna R. Pattipati and Mahesh Dontamsetty. On a generalized test sequencing problem. *IEEE Trans. Syst. Man Cybern.*, 22(2):392–396, 1992. doi:10.1109/21.148415.

49 Vili Podgorelec, Peter Kokol, Bruno Stiglic, and Ivan Rozman. Decision trees: An overview and their use in medicine. *Journal of medical systems*, 26:445–63, November 2002. doi:10.1023/A:1016409317640.

50 Sahil Singla. 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*, pages 2523–2532, 2018. doi:10.1137/1.9781611975031.161.

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

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

A Proofs from Section 4

▷ Claim 7. If there exists an $\alpha(n,m)$-approximation algorithm for PB then there exists a $\alpha(n,m)$-approximation for MSSC$_f$.

Proof of Claim 7. Let $\mathcal{I}$ be an instance of MSSC$_f$. We create an instance $\mathcal{I}'$ of 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 PB instance gives a solution to the MSSC$_f$ at the same cost and vice versa. ◁

▷ Claim 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$_{\leq T}$.

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$_{\leq T}$ 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 9. Given an instance $\mathcal{I}$ of UPB$_{\leq T}$ 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 [8].

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 $v_{ij} \leq T$. 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 $\leq T$ in the initial UPB$_{\leq T}$ 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 22 we have that $c(\pi_{\mathcal{I}}(s_i)) \leq 3c(\pi_{\mathcal{I}'}(s_i))$.

   Putting it all together we get

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

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$_{\leq T}$, 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 $T \leq m$, since otherwise the initial policy would never take the outside option. ◁

▶ **Lemma 22.** *Let $\mathcal{I}$ be an instance of UPB$_{\leq T}$, and $\mathcal{I}'$ the instance of UMSSC$_f$ constructed by the reduction of Claim 9. For a scenario s_i, if there is at least one outside option test run in $\pi_{\mathcal{I}}$, then $c(\pi_{\mathcal{I}}(s_i)) \leq 3c(\pi_{\mathcal{I}'}(s_i))$.*

[8] 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 $\pi_{\mathcal{I}'}$. Note that the smallest cost is achieved if all the outside option elements are chosen first[9]. 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. $N \geq T/2$: 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 $\pi_{\mathcal{I}'}$ then is at least
$$\begin{aligned}\sum_{j=1}^{K_i}\sum_{k=1}^{T/2}\frac{cp_\ell}{T}k + \sum_{j=1}^{K_i}\sum_{k=T/2+1}^{T}\frac{cp_\ell}{T}(T/2+M) &= cp_i\frac{\frac{T}{2}(\frac{T}{2}+1)}{2T} + cp_i\frac{\frac{T}{2}(T/2+M)}{T}\\ &\geq cp_i(3T/8 + M/2)\\ &\geq \frac{3}{8}p_i(T+M)\end{aligned}$$
   Since $N \geq T/2$, policy $\pi_{\mathcal{I}}$ 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 $\pi_{\mathcal{I}}$ is at most $p_i(M+T)$ in this case. Hence, we have that s_i's contribution in $\pi_{\mathcal{I}}$ is at most $\frac{8}{3} \leq 3$ times s_i's contribution in $\pi_{\mathcal{I}'}$.
2. $N < T/2$: policy $\pi_{\mathcal{I}}$ 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 $c(\pi_{\mathcal{I}})$ 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
$$\sum_{j=1}^{K_i}\sum_{k=N}^{T}\frac{cp_\ell}{T}M = cp_i\frac{T-N}{T}M \geq cp_iM/2,$$
   therefore, the contribution s_i has on $c(\pi_{\mathcal{I}'})$ is at least $cp_iM/2$. Hence, we have $c(\pi_{\mathcal{I}}) \leq 3c(\pi_{\mathcal{I}'})$. ◀

A.1 Proofs from subsection 4.1.1

▶ **Lemma 15.** *Given an instance $\mathcal{I}$ of PB; an α-approximation algorithm $\mathcal{A}_T$ to PB$_{\leq T}$; and any $q < 1$ and $\beta \geq 2$, suppose that the threshold T satisfies*
$$T \geq t_{q/(\beta\alpha)} + \beta\alpha \sum_{\substack{c_s \in [t_q, t_{q/(\beta\alpha)}]\\ s\in\mathcal{S}}} c_s\frac{p_s}{q}.$$
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.

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

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

$$c(\pi^*_{\mathcal{I}_q}) \leq c(\pi_{\mathcal{I}_q}) = \frac{T + t_{q/(\beta\alpha)}}{\beta\alpha} + \sum_{\substack{c_s \in [t_q, t_{q/(10\alpha)}] \\ s \in \mathcal{S}}} c_s \frac{p_s}{q}, \tag{3}$$

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

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

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

▶ **Lemma 14** (Optimal Lower Bound). *Let $\mathcal{I}$ be the instance of PB. For any $q < 1$, any $\alpha > 1$, and $\beta \geq 2$, for the optimal policy $\pi^*_{\mathcal{I}}$ for PB it that*

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

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. ◀

A.2 Proofs from subsection 4.1.2

Algorithm 3 Expand: rescales and returns an instance of UPB.

Input: Set of scenarios $\mathcal{S}$

Scale all probabilities by c such that $c \sum_{s \in \mathcal{S}} p_s = 1$

Let $p_{\min} = \min_{s \in \mathcal{S}} p_s$

$\mathcal{S}' =$ for each $s \in \mathcal{S}$ create $p_s/p_{\min}$ copies

Each copy has probability $1/|\mathcal{S}'|$

return $\mathcal{S}'$

▶ **Main Lemma 10.** *Given a polynomial-time $\alpha(n, m)$-approximation algorithm for UPB$_{\leq T}$, 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 11, and we are only highlighting the changes. The process of the reduction is the same as Algorithm 1 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$_{\leq T}$ (line 8). The reduction process is formally shown in Algorithm 2.

Calculating the thresholds. For every phase i we choose a threshold T_i such that $T_i = \min\{T : \mathbf{Pr}\left[c_s > T\right] \leq \delta\}$ i.e. at most δ of the probability mass of the scenarios are not covered, again using binary search as in Algorithm 1. 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 15, we know that for remaining total probability mass $(1-c)(\delta+c)^i$, any threshold which satisfies

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

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

$$T_i = t_{(1-c)(\delta+c)^{i-1}/\beta\alpha} + \beta\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in\text{Int}_i}} c_s \frac{p_s}{(1-c)(\delta+c)^i}. \tag{4}$$

Following the proof of Theorem 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 $\text{PB}_{\leq T}$ is run in an interval of the form $\text{Int}_i = [t_{(1-c)(\delta+c)^i}, t_{(1-c)(\delta+c)^i/(\beta\alpha)}]$. Note also that these intervals are overlapping. Bounding the cost of the final policy $\pi_{\mathcal{I}}$ for all intervals we get

$$\begin{aligned}
\pi_{\mathcal{I}} &\leq \sum_{i=0}^{\infty}(1-c)(\delta+c)^i T_i \\
&= \sum_{i=0}^{\infty}\left((1-c)(\delta+c)^i t_{(1-c)(\delta+c)^{i-1}/\beta\alpha} + \beta\alpha \sum_{\substack{s\in\mathcal{S}\\ c_s\in\text{Int}_i}} c_s p_s\right) && \text{From equation (4)} \\
&\leq 2\cdot\beta\alpha\pi^*_{\mathcal{I}} + \beta\alpha\sum_{i=0}^{\infty}\sum_{\substack{s\in\mathcal{S}\\ c_s\in\text{Int}_i}} c_s p_s && \text{Using Lemma 14} \\
&\leq 2\beta\alpha\log\alpha\cdot\pi^*_{\mathcal{I}},
\end{aligned}$$

where the inequalities follow similarly to the proof of Theorem 11. Choosing $c=\delta=0.1$ and $\beta=20$ we get the theorem. ◀